Preserving Numerical Algorithms

Árpád Lukács,

in collaboration with **J.C. Nash**

Fosdem 2019

2[nd] February

Brussels

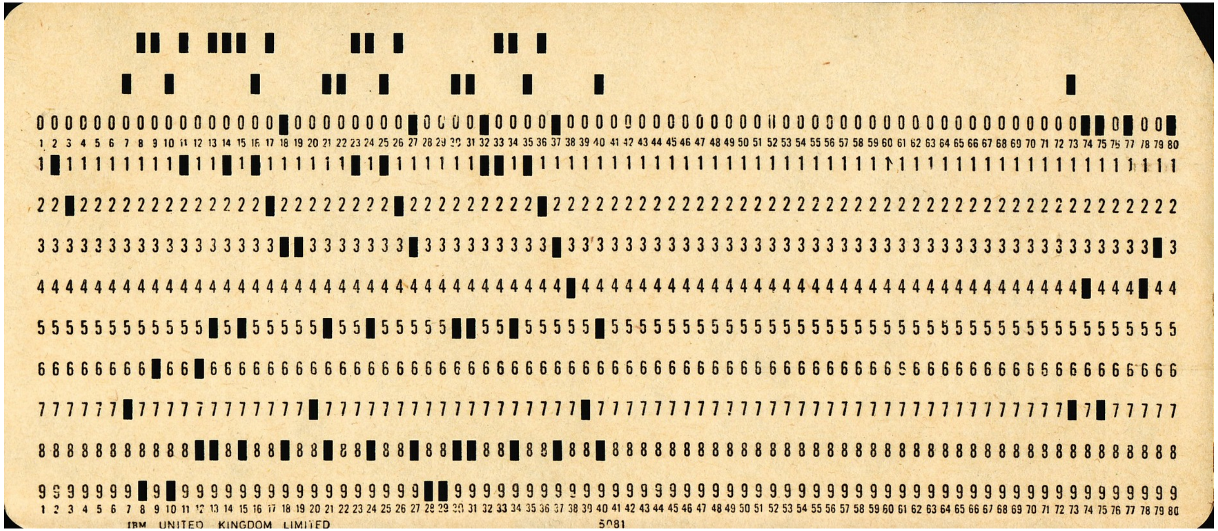

Image source: Wikipedia

TU Delft, The Netherlands & Wigner RCP, Budapest, Hungary

Support by the **histoRicalg** project of the R consortium

Overview

1. Introduction
2. Availability of programming languages
3. Subroutine libraries
4. Experience with running codes
5. Conclusions

Preservation of algorithmic knowledge

- Long history of numerical mathematics
  - mathematical background (calculus, approximation theory)
  - hand computation
  - tables of functions
  - many algorithms known (FFT: Gauss, Lanczos; Euler; Runge-Kutta)
- Description of algorithms aimed at human computers:
  - one can expect the computer to think (e.g., recognize exceptional cases)
  - go back if more digits needed (loss of significance)
- Computer age (Manchester Baby, 1948)
  - Communicate algorithms to humans & computers (machines)
  - For machines, nothing left to the reader
  - the most precise description of an algorithms: a computer program

histoRicalg

- Project PI: . John C. Nash, University of Ottawa
  - Goal: provenance of numerical algorithms used by the R interpreter
- Initial momentum: **`uncmin`**
  - Unconstrained minimization
  - Many (**Fortran**) versions online → Bug found in **R**'s version
- Was the bug present in the original version?
- Is the original code still available (is it machine readable)?
- Some algorithms have a long history:
  - **Fortran** → C by **f2c**, manual editing
  - **Algol 60** → **Fortran** (manually), C by **f2c**, ...

Programming languages

Early languages:
on one machine only

Assemblers,
also machine specific

1957: **Fortran**,
standardization: 1966, 1977, 1990, 1995 ...

1958: **Algol**
standardization: **Algol** 60

Later **Algol** variants (**Algol** 68, **Algol W**)

1964: **BASIC**
standardization: 1978, 1984, 1987, 1991

Pascal

APL, **PL/I**, ...

So many languages ...
Can we still run them on a PC?

! Important !

> Comparison with new implementations

> Are there bugs?

Additional difficulty:

> 1985: floating point standardization

Quick Look at Fortran

- John Backus, IBM 1957
  - Release of the first Fortran compiler
  - In a year, most computer centers > 50% of code in Fortran
  - SHARE users group
  - IBM SSP = Scientific subroutine package
- Development of the language
  - ASA Fortran standardized in 1966
  - Continued development since (latest 2018)
  - Newer versions (mostly) include older ones
- Almost perfect support
  - Compilers with good standard compliance (`gfortran`, `ifort`)
  - Some vendor extensions also supported (`REAL*8`)
  - Source to source translator available to `C` (`f2c`)

Quick Look at Algol 60

- History
  - Ideas from Algol 58 (IAL International Algorithmic Language)
  - Algol 60 report: a precise specification of the syntax (BNF Backus normal form)
  - Some later bug fixes (Revised report, 1964; Modified report, 1976)
- Some properties of the language
  - introduced block-structure
  - most presently used programming languages in the ”algol family”
  - does not include i/o (expected to change too much)
  - not supported by IBM
  - machine representations vary, rather different from printed form
  - often used as pseudo-code
- Almost perfect support
  - translators (to `C`) (`marst`, `jff-algol`), interpreter (`nase`) available

Subroutine libraries

- History
  - Idea of subroutines: John Mauchly, 1947
  - Paper: Goldstine and von Neumann, 1948
  - First published library: Wilkes, Wheeler, Gill, 1951 (EDSAC binary)
  - Personal collections
  - SHARE user's group
- Formal Collections
  - Journals:
    - Communications of the ACM
    - ACM Transactions on Mathematical Software
    - Numerische Mathematik
    - Mathematics of computation
    - Computer Physics Communications
  - Books:

Subroutine libraries 2

> Books on computational methods:

- Wilkes, Wheeler, and Gill,
  1951 EDSAC machine code
- Wilkinson and Reinsch,
  Handbook of automatic computation, 1971
  Algol 60 code, later translated into Fortran: EISPACK
- Shampine and Allen, Numerical computing,
  1973 Fortran
- Forsythe and Moler,
  Computer solution of linear algebraic systems,
  1967 Fortran, Algol 60, PL/1
- Forsythe, Malcolm, and Moler,
  Computer methods ... , 1977
  Fortran code, machine readable
- Nash, Compact numerical methods,
  1979, 1990
  Pascal code, Fortran translation also available
- Kahaner, Moler, Nash,
  Numerical methods and software, 1980
  Fortran code, available on diskette
- Watanabe etal., Mathematical software
  for the PC and workstations,
  1991 - Fortran code, available on diskette
- Press, Teukolsky, Vetterling, Flannery,
  Numerical recipes, 1986, ... , 2007
- Many versions and languages:
  Fortran, Pascal, C, C++

Subroutine libraries 3

- Collected algorithms of the ACM: CALGO
  - 1-100 in Algol 60, mostly numerical (tested 1-4 → compiler bugs)
  - later more languages: **`Fortran`**, **`PL/I`**, . . . , Matlab
  - only later ones in machine readable form
- Other journals
  - Mathematics of computation: Shrager's root finder
  - KFKI report (Q-D root finder in Algol 60)

Subroutine libraries 4

- Formal subroutine libraries
  - **SSP**, IBM
  - **PORT**, Bell Laboratories, TOMS 528 (1978, Fortran)
    Important ideas: portable floating point, error handling, strong in extrema
  - **SLATEC**, US National Laboratories, (1982, Fortran)
    Uses PORT FP & err, large collection
  - **CMLIB**. NIST (Fortran); large overlap with SLATEC, includes `uncmin`
  - **NSWC**, Naval Surface Warfare Center, (Fortran)
    overlap with SLATEC, strong in special functions
  - **NUMAL** (Matematisch Center, Amsterdam, 1974, `Algol` 60)
    large collection, hand-compiled, translation to `C` by Lau
  - commerical libraries (**NAG**, **IMSL**, in `Fortran`)

Codes Tested: Algol 60

- ACM algorithms 1-4:
  - quadrature, secant, Bairstow, bisection, 1960
  - printed representation (differs from machine one)
  - found bugs in `jff-algol` compiling Bairstow
- NUMAL (1985):
  - exponential integral Ei(x)
  - in machine representation
  - one bug in `jff-algol` found
- Gellai, Determination of roots of polynomials, KFKI report (1971)
  - contains Q-D for real roots and a modified version of ACM 3 Bairstow
  - already in a machine representation

Experience with Algol 60

- Algol 60 compilers in Linux have a small userbase
  - needed to build Ubuntu
  - packages from source found some bugs
- Printed and computer representations of Algol 60 differ
  - character set differences
  - quoted keywords
- I/O in some cases the recommended (Knuth 1964)
- support good enough for comparison with a new implementation

Experience with Fortran

- Good compiler support on Linux, Windows
- Some non-standard things
  - assumed shape arrays (`dimension x(1)` → `dimension x(*)`)
  - **common trick** with **common blocks**:
    array only dimensioned in main program:
    ```
    common /blockname/ arr(1)
    ```
    Tell modern compiler not to use array size to optimize loops:
    `gfortran` option `-funconstrained-commons`
  - C and `Fortran` interoperability: `gfortran` `REAL` is `float` (`f2c`: `double`)
- Good compiler support:
  - for code preservation, maybe fix standard violations
  - write Makefile

SLATEC

Sandia, **L**os Alamos, **A**ir Force Weapons Laboratory **T**echnical **E**xchange **C**omittee

> **Fortran** 77
>
> Large collection of numerical subroutines (1400 user callable routines)
>
> Includes: BLAS, EISPACK, LINPACK, FFTPACK, FISHPACK, FNLIB, PCHIP, QUADPACK, DEPAC, DASSL, SDRIVE, SLAP

SLATEC

Sandia, **L**os Alamos, **A**ir Force Weapons Laboratory **T**echnical **E**xchange **C**omittee

- `Fortran` 77
- Large collection of numerical subroutines (1400 user callable routines)
- Includes: `BLAS`, `EISPACK`, `LINPACK`, `FFTPACK`, `FISHPACK`, `FNLIB`, `PCHIP`, `QUADPACK`, `DEPAC`, `DASSL`, `SDRIVE`, `SLAP`

- Experience
  - Debian package source available, but old (does not compile)
  - Changes only needed for machine properties part (`f2c` → `gfortran`)
  - All tests run
  - Minimal work:
    implement double precision version where missing, Ubuntu package

PORT

Portable, **O**utstanding, **R**eliable and **T**ested, Bell Laboratories, 1978 - 199x

- `Fortran 77`
- Bell Labs website removed, public domain part on `netlib.org`
- Most of it saved by Prof. Nash before (w/o PD parts)
- Still missing:
  - Banded matrix subroutines (?)
    `bnddq`, `bndls`, `bndqd`, `bndqf`, `bndqr`, `bndqs`, `bndqr`, `bndqs`
  - Minimization iteration step `n2itr` (probably removed, `nl2sol` replaced)
  - Sparse matrix forward solve

PORT

Portable, **O**utstanding, **R**eliable and **T**ested, Bell Laboratories, 1978 - 199x

- `Fortran` 77
- Bell Labs website removed, public domain part on `netlib.org`
- Most of it saved by Prof. Nash before (w/o PD parts)
- Still missing:
  - Banded matrix subroutines (?)
    `bnddq`, `bndls`, `bndqd`, `bndqf`, `bndqr`, `bndqs`, `bndqr`, `bndqs`
  - Minimization iteration step `n2itr` (probably removed, `nl2sol` replaced)
  - Sparse matrix forward solve

- Experience
  - Changes only required to `Makefile`, all tests run.)

UNCMIN

UNCMIN = **unc**onstrained **min**imization

> `Fortran` 77

> Standalone version on the internet

> **Kahaner**, **Moler**, and **Nash** (1989) has a subset on companion diskette

> **NIST CMLIB** includes uncmin, one author at **NIST** (**Koontz**)

> R bug report: C version: converted with f2c, has been edited
bug identified in `choldc()` (Cholesky decomposition)

> Original Fortran `uncmin` converges for the example

> Backported R's new `choldc` to Fortran: same result, number of steps

> Bug not in the original

Conclusions

- **Legacy mathematical code**
  - **Essential** for preservation of **algorithmic knowledge**
  - Rather high quality
  - Old programming languages: `Fortran` (pre-77), `Algol` 60
- **Tasks**
  - Future-proof codes
  - Test toolchain (Algol 60 translator)
  - Preserve code
- **Bugs?**

Thank you for your attention

Want to contribute?

Visit histoRicalg:

https://gitlab.com/nashjc/histoRicalg

We thank the R-consortium for support.