Numerical Modelling in 
**FORTRAN** day 11

Paul Tackley, 2018
Today’s Goals


2. **Projects**: Define goals, identify information needed from me. Due date: 15 February 2018
1. Chosen topic, agreed upon with me (suggestions given, also ask the advisor of your MSc or PhD project).
   – Due end of Semesterprüfung (15 Feb 2018)
   – Decide topic by final lecture!
   – [http://jupiter.ethz.ch/~pjt/FORTRAN/FortranProject.html](http://jupiter.ethz.ch/~pjt/FORTRAN/FortranProject.html)
Project: general guidelines

• Choose something either
  – related to your research project and/or
  – that you are interested in
• Effort: 1 KP => 30 hours. About 4 days’ work.
• I can supply information about needed equations and numerical methods that we have not covered
Some ideas for a project

• Involving solving partial differential equations on a grid (like the convection program)
  – Wave propagation
  – Porous flow (groundwater or partial melt)
  – Variable-viscosity Stokes flow
  – Shallow-water equations
  – 3-D version of convection code

• Involving other techniques
  – Spectral analysis and/or filtering
  – Principle component analysis (multivariate data)
  – Inversion of data for model parameters
  – N-body gravitational interaction (orbits, formation of solar system, ...)
  – Interpolation of irregularly-sampled data onto a regular grid
Fortran Review

• History
  – 60 years since first compiler

• Variables
  – types: real, integer, logical, character, complex
  – Implicit types and implicit none
  – Initialisation
  – parameters
  – defined types
  – precision (32 vs 64 bit etc.)
Fortran Review (2)

- Arrays
  - fixed vs. automatic vs. allocatable vs. assumed shape
  - index ranges
  - data, reshape
  - built-in array algebra

- Input/Output
  - open, print, read, write, close
  - formats
  - iostat, rewind, status
  - namelist
  - binary vs. ascii, direct vs. sequential
Fortran Review (3)

• Flow control structures
  – do loops (simple, counting, while), exit, cycle
  – if...elseif...endif blocks
  – select case...
  – forall
  – where
Functions and subroutines (procedures)

- Intrinsic functions: mathematical, conversion, ...
- Internal (contains), external (& interface blocks), in modules
- Array functions
- Recursive functions and result statement
- Named arguments
- Optional arguments
- Generic procedures
- Overloading
- User-defined operators
- Save
Fortran Review (5)

• Modules
  – use
  – public and private variables and functions
  – =>

• Pointers
  – to variables, arrays, array sections, areas of memory, in defined types

• Optimization
  – maximize use of cache and pipelining
  – loops most important
  – 90/10 rule (focus on bottlenecks)
  – avoid branches, maximize data locality, unroll, tiling,…

• Libraries, makefiles
For a more detailed review & discussion, see:

New features of Fortran 2003

• For a detailed description see: https://wg5-fortran.org/N1601-N1650/N1648.pdf


• Enhancements to derived types
  – Parameterized derived types
  – Improved control of accessibility
  – Improved structure constructors
  – Finalizers
New features of Fortran 2003 (2)

• Object-oriented programming support
  – Type extension and inheritance
  – Polymorphism
  – Dynamic type allocation
  – Type-bound procedures

• Data manipulation enhancements
  – Allocatable components
  – Deferred type parameters
  – VOLATILE attribute
  – Explicit type specification in array constructors & allocate statements
  – Extended initialization expressions
  – Enhanced intrinsic procedures
New features of Fortran 2003 (3)

• Input/output enhancements
  – Asynchronous transfer
  – Stream access
  – User specified transfer ops for derived types
  – User specified control of rounding
  – Named constants for preconnected units
  – FLUSH
  – Regularization of keywords
  – Access to error messages

• Procedure pointers
New features of Fortran 2003 (4)

• Support for the exceptions of the IEEE Floating Point Standard
• Interoperability with the C programming language
• Support for international characters (ISO 10646 4-byte characters...)
• Enhanced integration with host operating system
  – Access to command line arguments, environment variables, processor error messages
• Plus numerous minor enhancements
Fortran 2008

• Minor update of Fortran 2003

• Brief summary (Wikipedia)
  – Submodules – additional structuring facilities for modules; supersedes ISO/IEC TR 19767:2005
  – Coarray Fortran – a parallel execution model
  – The DO CONCURRENT construct – for loop iterations with no interdependencies
  – The CONTIGUOUS attribute – to specify storage layout restrictions
  – The BLOCK construct – can contain declarations of objects with construct scope
  – Recursive allocatable components – as an alternative to recursive pointers in derived types

• For more details see ftp://ftp.nag.co.uk/sc22wg5/n1801-n1850/n1828.pdf
The oldest high-level programming language
Numerical methods review

• Approximations
  – Concept of discretization
  – Finite difference approximation
  – Treatment of boundary conditions

• Equations
  – diffusion equation
  – Poisson’s equation
  – advection-diffusion equation
  – Stokes equation and Navier-Stokes equation
  – streamfunction and streamfunction-vorticity formulation
  – Pr, Ra, Ek
  – initial value vs. boundary value problems
Numerical methods Review (2)

• Timestepping
  – stability (timestep, advection scheme)
  – explicit vs. implicit, semi-implicit
  – upwind advection

• Solvers
  – direct
  – iterative (relaxation), multigrid method
  – Jacobi vs. Gauss-Seidel vs. red-black

• Parallelisation
  – Domain decomposition
  – Message-passing and MPI
Discussion

• We have focussed on one application (constant viscosity convection)
• Can apply same techniques to other applications / physical problems
Example: Darcy Equation
(groundwater or partial melt flow)

\[ \vec{u} = -\frac{k}{\eta}\left(\vec{\nabla}P - g\rho\hat{y}\right) \]

\( \vec{u} = \) volume/area/time = “Darcy velocity” (really a flux)

\[ \vec{\nabla} \cdot \vec{u} = 0 \]

\( k = \) permeability (related to porosity and interconnectivity)
\( \eta = \) viscosity of fluid
\( \rho = \) density of fluid, \( P = \) pressure in fluid
Simplify and rearrange

Take divergence: velocity is eliminated
\[ \nabla \cdot \vec{u} = -\nabla \cdot \left( \frac{k}{\eta} \nabla P \right) + \nabla \cdot \left( \frac{k}{\eta} g \rho \hat{y} \right) = 0 \]

2nd order equation for pressure
\[ \nabla \cdot \left( \frac{k}{\eta} \nabla P \right) = g \frac{\partial}{\partial y} \left( \frac{\rho k}{\eta} \right) \]

Combine k and \( \eta \) into hydraulic resistance R
\[ \nabla \cdot \left( \frac{1}{R} \nabla P \right) = g \frac{\partial}{\partial y} \left( \frac{\rho}{R} \right) \]

A Poisson-like equation for pressure => easy to solve using existing methods
Example 2: Full elastic wave equation

Most convenient to solve as coupled first-order PDEs

\[ F = ma \text{ for a continuum:} \]
\[ \rho \frac{\partial \vec{v}}{\partial t} = \nabla \cdot \sigma \]

Hooke’s law for an isotropic material
\[ \sigma_{ij} = \lambda \delta_{ij} \epsilon_{kk} + 2\mu \epsilon_{ij} \]

Where \( \epsilon_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \)
\( u = \) displacement of point from equilibrium position

\( \mu \) = shear modulus. \( \lambda, \mu \) are known as Lamé constants

Sometimes written
\[ \sigma_{ij} = K \epsilon_{kk} \delta_{ij} + 2\mu (\epsilon_{ij} - \frac{1}{3} \epsilon_{kk} \delta_{ij}) \]

Using \( K = \) bulk modulus
Resulting equations to solve using finite differences

Variables: 3 velocity components and 6 stress components

On staggered grid (same as viscous flow modelling)

\[
\frac{\partial}{\partial t} v_x = \frac{1}{\rho} \left( \frac{\partial}{\partial x} \sigma_{xx} + \frac{\partial}{\partial y} \sigma_{xy} + \frac{\partial}{\partial z} \sigma_{xz} \right)
\]

\[
\frac{\partial}{\partial t} \sigma_{xx} = (\lambda + 2\mu) \frac{\partial v_x}{\partial x} + \lambda \left( \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z} \right)
\]

\[
\frac{\partial}{\partial t} v_y = \frac{1}{\rho} \left( \frac{\partial}{\partial x} \sigma_{xy} + \frac{\partial}{\partial y} \sigma_{yy} + \frac{\partial}{\partial z} \sigma_{zy} \right)
\]

\[
\frac{\partial}{\partial t} \sigma_{yy} = (\lambda + 2\mu) \frac{\partial v_y}{\partial y} + \lambda \left( \frac{\partial v_x}{\partial x} + \frac{\partial v_z}{\partial z} \right)
\]

\[
\frac{\partial}{\partial t} v_z = \frac{1}{\rho} \left( \frac{\partial}{\partial x} \sigma_{zx} + \frac{\partial}{\partial y} \sigma_{zy} + \frac{\partial}{\partial z} \sigma_{zz} \right)
\]

\[
\frac{\partial}{\partial t} \sigma_{zz} = (\lambda + 2\mu) \frac{\partial v_z}{\partial z} + \lambda \left( \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} \right)
\]

\[
\frac{\partial}{\partial t} \sigma_{xy} = \mu \left( \frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x} \right)
\]

\[
\frac{\partial}{\partial t} \sigma_{xz} = \mu \left( \frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x} \right)
\]

\[
\frac{\partial}{\partial t} \sigma_{yz} = \mu \left( \frac{\partial v_y}{\partial z} + \frac{\partial v_z}{\partial y} \right)
\]
Mathematical classification of 2nd-order PDEs: Elliptical, hyperbolic, parabolic

Elliptical
\[ \nabla^2 \phi = f \]
e.g., Poisson

Parabolic
\[ \frac{\partial \phi}{\partial t} \propto \nabla^2 \phi \]
e.g., diffusion

Hyperbolic
\[ \nabla^2 \phi \propto \frac{1}{V^2} \frac{\partial^2 \phi}{\partial t^2} \]
e.g., wave eqn.

We’ve done them all!
Waves: \( \frac{\partial^2 P}{\partial t^2} = v^2 \nabla^2 P \)

Fluids:
\[
\frac{1}{\Pr} \left( \frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} \right) = -\nabla P + \nabla \cdot \left( \eta \left( \frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right) + \frac{1}{E_k} \hat{\Omega} \times \vec{v} + Ra \cdot T \hat{g}
\]

\[
\frac{\partial T}{\partial t} + \vec{v} \cdot \nabla T = \nabla \cdot (k \nabla T)
\]

\( \nabla \cdot \vec{v} = 0 \)

Percolation:
\[
\nabla \cdot \left( \frac{k}{\eta} \nabla P \right) = g \frac{\partial}{\partial y} \left( \frac{\rho k}{\eta} \right)
\]

N-body:
\[
\frac{d^2 \vec{x}_i}{dt^2} = \vec{a}_i \quad \vec{a}_i = G \sum_{j=1}^{n, j \neq i} \frac{m_j}{|\vec{x}_j - \vec{x}_i|^2} \cdot \frac{\vec{x}_j - \vec{x}_i}{|\vec{x}_j - \vec{x}_i|}
\]

Note the many similar terms!
END! FORTRAN