Implementation of advanced numerical solvers in FLAC3D thermal and fluid implicit formulation

Andrey V. Pyatigorets, David B. Russell
**FLAC3D thermal and fluid logic**

- FLAC3D thermal / fluid logic adopt energy-balance / fluid mass-balance equations formulated at each gridpoint:

\[
\frac{dT_n}{dt} = -\frac{1}{\sum m_{th}^n} Q_{th}^n \\
\frac{dP_n}{dt} = -\frac{1}{\sum m_{fl}^n} Q_{fl}^n
\]

- FLAC3D thermal and fluid modules have two separate solvers: explicit and implicit

- Explicit solver uses forward difference scheme:  
  \[T_{new} = T_{old} + \Delta t \cdot f(T_{old})\]

  **Timestep \(\Delta t\)** must be small (often \(10^{-4} \leftrightarrow 10^{-6}\) sec)!

- Current implicit solver uses:
  - Central difference (Crank-Nicolson) scheme to obtain a system of linear equations for each gridpoint:  
    \[A_{nj} \Delta T_j(t) = b_{n(t)}\]
  - Jacobi method (iterative) to solve it

  **Jacobi solver easily diverges (fails) when** \(\Delta t_{implicit} \gtrsim [10 \leftrightarrow 100] \Delta t_{explicit}\)

Need to find an approach which will allow using much larger timestep in the implicit scheme!
New approach – use global matrix

• Global matrix is assembled for the whole model based on previously calculated local matrices:

\[
\begin{bmatrix}
Q_1^a \\
Q_4^a \\
Q_3^a
\end{bmatrix} =
\begin{bmatrix}
c_{11}^a & c_{14}^a & c_{13}^a \\
c_{44}^a & c_{43}^a & c_{33}^a
\end{bmatrix}
\begin{bmatrix}
\Delta T_1 \\
\Delta T_4
\end{bmatrix}
\]

\[
\begin{bmatrix}
Q_1^b \\
Q_2^b \\
Q_4^b
\end{bmatrix} =
\begin{bmatrix}
c_{11}^b & c_{12}^b & c_{14}^b \\
c_{22}^b & c_{23}^b & c_{24}^b \\
c_{44}^b
\end{bmatrix}
\begin{bmatrix}
\Delta T_1 \\
\Delta T_4
\end{bmatrix}
\]

\[
\begin{bmatrix}
Q_2^c \\
Q_3^c \\
Q_4^c
\end{bmatrix} =
\begin{bmatrix}
c_{22}^c & c_{23}^c & c_{24}^c \\
c_{33}^c & c_{34}^c \\
c_{44}^c
\end{bmatrix}
\begin{bmatrix}
\Delta T_2 \\
\Delta T_3 \\
\Delta T_4
\end{bmatrix}
\]

\[
\begin{bmatrix}
Q_1 \\
Q_2 \\
Q_3 \\
Q_4
\end{bmatrix} =
\begin{bmatrix}
Q_1^a + Q_1^b \\
Q_2^b + Q_2^c \\
Q_3^a + Q_3^c \\
Q_4^a + Q_4^b + Q_4^c
\end{bmatrix}
= \begin{bmatrix}
c_{11} + c_{11}^a & c_{12}^b & c_{13}^a & c_{14} \\
& c_{22}^b + c_{22}^c & c_{23}^a & c_{24}^a + c_{24}^c \\
& & c_{33}^c & c_{34}^c \\
& & & c_{44} + c_{44}^a + c_{44}^c
\end{bmatrix}
\begin{bmatrix}
\Delta T_1 \\
\Delta T_2 \\
\Delta T_3 \\
\Delta T_4
\end{bmatrix}
\]

• Real global matrix is very sparse, symmetric and positive definite!
New solvers

- Two new solvers from INTEL MKL are adopted:
  - **Preconditioned Conjugate Gradient (PCG) Solver** – iterative solver for very large sparse systems (e.g. millions of equations). Uses simple Jacobi (diagonal) preconditioner.
  - **Direct Sparse Solver (DSS)** – direct solver for smaller sparse systems (equivalent to matrix inversion; LU / Cholesky).

- New solvers operate on **global matrix** representing the whole model. The matrix must be assembled in compressed sparse format (CSR3) - done once when user calls new solvers.

- Global matrix must be symmetric and positive definite. To check this and to find matrix condition number $\kappa$, extremal eigenvalues are calculated (Krylov-Schur method). Limit on number of iteration in PCG:
  $$\kappa = \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}}, \quad n \leq \frac{1}{2} \kappa \log \frac{2}{\varepsilon}$$

- Solvers are unconditionally stable! Solution can always be found up to the specified accuracy (providing the time step is below the characteristic time of the model)!
Performance

- Simple diffusion problem: \( T_{\text{bot}} = 100\text{C}, \ T_{\text{top}} = 0. \)

Results of temperature distribution after 15sec:

<table>
<thead>
<tr>
<th>Scheme / Solver</th>
<th>Explicit</th>
<th>Implicit / Jacobi</th>
<th>Implicit / precond. CG</th>
<th>Implicit / direct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timestep</td>
<td>1.28856e-5</td>
<td>1e-4</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>Number of steps</td>
<td>1164094</td>
<td>150000</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>Error</td>
<td>2e-5</td>
<td>5e-4</td>
<td>9e-5</td>
<td>9e-5</td>
</tr>
<tr>
<td>Relative speed-up (runtime)</td>
<td>1X (11m02s)</td>
<td>2.1X (5m13s)</td>
<td>441X (1.5sec)</td>
<td>662X (1sec)</td>
</tr>
<tr>
<td>Time/Step</td>
<td>5.68e-4</td>
<td>2.09e-3 (3.7x)</td>
<td>5e-2 (88x)</td>
<td>3.33e-2 (58x)</td>
</tr>
</tbody>
</table>

Overall speed-up over the explicit logic is \( 10^2 - 10^6 \) (for thermal diffusion problems involving large time step and large calculation time)

New solvers have been tested on:
- heat conduction,
- thermo-mechanical coupling,
- constant and transient boundary conditions,
- heat sources/sinks,
- convective boundaries.
Challenges / limitations

• Currently there is an issue within thermal/fluid Attach logic which does not properly account for presence of interfaces within same zone (when multiple attach conditions have to be resolved).

• Possible oscillations (from C-N scheme) for large time steps (so small number of cycles)!

• The global matrix has to be re-assembled every time the model geometry or physical properties of gp/zones change. Thus, new solvers may be inefficient for:
  o Problems in large strain mode
  o Problems with moving boundaries
  o Problems requiring small time step

Explicit time step ~0.00001 sec
Implicit time step ~0.0001 sec
Stable new time step ~0.1-0.2sec
Applicability

- Choice of proper solver is done by user:
  - Explicit method
  - Implicit Jacobi solver
  - Implicit PCG solver
  - Implicit direct solver

- It depends on:
  - Physical parameters (i.e. time-dependent)
  - Boundary conditions (i.e. time-dependent)
  - Timestep
  - Model size

**Generic solvers applicability for best performance**
Command syntax

```plaintext
model configure thermal

model thermal active on

• zone thermal implicit on

• zone thermal implicit on solver-jacobi

• zone thermal implicit on solver-pcg [preserve]

• zone thermal implicit on solver-direct [preserve]

model thermal timestep fix ...

model cycle ... / solve time ...
```

FLAC3D 7: calls Jacobi solver only

FLAC3D 8: calls Jacobi solver; if diverges – automatically switches to PCG solver
NWMO project


Ruiping Guo
NWMO-TR-2016-03
NWMO project


Temperature simulation in a nuclear waste repository over 1 million years

- FLAC3D 5 runtime is \( \approx 1 \text{ month} \) with single-thread thermal logic
- FLAC3D 7 runtime \( \approx 2-4 \text{ weeks} \) with multi-threaded thermal logic
- FLAC3D 8 runtime is \( \approx 25-30 \text{ minutes} \) using combination of new solvers

- The largest stable timestep for Jacobi solver (old technique) \( \approx 30 \text{ days} \)
- The largest timestep used in new solvers \( \approx 1200 \text{ years} \) (limited artificially, solvers allow larger)
NWMO results

**FLAC3D 8.00**
©2019 Itasca Consulting Group, Inc.

Gridpoint temperature
1 Gridpoint Temperature at g-p 177
vs. 1 thermal total time

Results validation (10 000 years)

- FLAC3D 7.0 - multithreaded Jacobi solver (~ 43 hours)
- FLAC3D 8 - PCG/DSS solvers (~ 25min)

Temperature, C vs. Log time, sec

- PCG
- DSS
- ~25 min
- 2-3 min

**Total time, sec \times 10^{13}**
NWMO results

**FLAC3D 8.00**
©2019 Itasca Consulting Group, Inc.

**History**
implDt implicitDt (FISH)
7.0000e+01 <-> 3.7722e+10
vs. 1 thermal total time
8.0270e+03 <-> 3.2973e+13

**Evolution of implicit timestep (70sec - 1196years)**

Time step is dynamically adjusted via a FISH function