A proposal for parallel implicit time evolution

Andy Wathen
Oxford University, UK

joint work with
Elle McDonald
Heat equation:

\[
\frac{\partial u}{\partial t} - \nabla^2 u = f \quad \text{in} \quad \Omega \times (0, T], \quad \Omega \subset \mathbb{R}^2 \text{ or } \mathbb{R}^3
\]

\[u(x, 0) = u_0(x), \quad u \text{ given on } \partial \Omega\]

Finite elements in space \((x)\), \(\theta\) time stepping gives

\[
M \frac{u_k - u_{k-1}}{\tau} + K \left( \theta u_k + (1 - \theta)u_{k-1} \right) = f_k
\]

\(M \in \mathbb{R}^{n \times n}\): SPD mass matrix (identity operator, but same sparsity as \(K\))

\(K \in \mathbb{R}^{n \times n}\): SPD discrete (negative) Laplacian (stiffness matrix)
Rearranging:

\[
(M + \tau \theta K) u_k = (M - \tau (1 - \theta) K) u_{k-1} + \tau f_k,
\]

\[k = 1, 2, \ldots, N\]

\[N \tau = T\]

Recall for unconditional stability: \(\frac{1}{2} \leq \theta \leq 1\)

\(\theta = 1\): backwards Euler, \(\theta = \frac{1}{2}\): Crank-Nicolson

else need \(\tau = \mathcal{O}(h^2)\): very small time steps for explicit method
\[
(M + \tau \theta K) u_k = (M - \tau (1 - \theta) K) u_{k-1} + \tau f_k,
\]

\[k = 1, 2, \ldots, N\]

Standard solution method:

Solve the \(N\) separate \(n \times n\) linear systems \textit{sequentially} for \(k = 1, 2, \ldots, N\) e.g. by algebraic multigrid (we use HSL_MI20)

\[\Rightarrow r = 5\text{ V-cycles for solution of each linear system to a relative residual tolerance of } 10^{-6}\]

Hence if we (quite reasonably) regard 1 V-cycle as the main unit of work

\[\Rightarrow Nr\text{ V-cycles sequentially for the overall solution}\]
Alternative proposal for parallel computation:

Write all timesteps at one go (all-at-once method):

\[ \mathcal{A} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix} = r.h.s \]

where \( \mathcal{A} \) is the matrix

\[
\begin{bmatrix}
M + \tau \theta K & 0 & 0 & 0 \\
- M + \tau (1 - \theta) K & M + \tau \theta K & 0 & 0 \\
0 & \ddots & \ddots & 0 \\
0 & 0 & - M + \tau (1 - \theta) K & M + \tau \theta K \\
\end{bmatrix}
\]

and \( r.h.s. = [M - \tau (1 - \theta) K u_0 + \tau f_1, \tau f_2, \ldots, \tau f_N]^T \)
\[
\mathcal{A} = \begin{bmatrix}
M + \tau \theta K & 0 & 0 & 0 \\
-M + \tau (1 - \theta) K & M + \tau \theta K & 0 & 0 \\
0 & \ddots & \ddots & 0 \\
0 & 0 & -M + \tau (1 - \theta) K & M + \tau \theta K
\end{bmatrix}
\]

\[\mathcal{A} \in \mathbb{R}^{L \times L}, \ L = Nn\]

We propose to solve this huge linear system (for the solution at all time steps) by GMRES (or BICGSTAB) with block diagonal preconditioner

\[
\mathcal{P} = \begin{bmatrix}
(M + \tau \theta K)_{MG} & 0 & 0 & 0 \\
0 & (M + \tau \theta K)_{MG} & 0 & 0 \\
0 & \ddots & \ddots & 0 \\
0 & 0 & 0 & (M + \tau \theta K)_{MG}
\end{bmatrix}
\]

where \((M + \tau \theta K)_{MG}\) is one AMG V-cycle exactly as above.
Theory: If we used

$$P_{\text{exact}} = \begin{bmatrix}
(M+\tau\theta K) & 0 & 0 & 0 \\
0 & (M+\tau\theta K) & 0 & 0 \\
0 & \ddots & \ddots & 0 \\
0 & 0 & 0 & (M+\tau\theta K)
\end{bmatrix}$$

as preconditioner (no AMG approximation) then we would have

$$P_{\text{exact}}^{-1}A = \begin{bmatrix}
I & 0 & 0 & 0 \\
J & I & 0 & 0 \\
0 & \ddots & \ddots & 0 \\
0 & 0 & J & I
\end{bmatrix},$$

$$J = (M+\tau\theta K)^{-1}(-M+\tau(1-\theta)K)$$
For

\[ P_{\text{exact}}^{-1} A = \begin{bmatrix} I & 0 & 0 & 0 \\ J & I & 0 & 0 \\ 0 & \cdot & \cdot & \cdot \\ 0 & 0 & J & I \end{bmatrix}, \]

the minimum polynomial is \((1 - s)^N\), so GMRES would terminate (in exact arithmetic) in \(N\) iterations.

We observe that \((M + \tau \theta K)_M G\) is spectrally so close to \((M + \tau \theta K)\) that convergence to a tolerance much less than the discretization error is achieved in \(N\) iterations also with \(P\) as preconditioner.
For $N=5$:

\[ \lambda(P^{-1}A_{BE}) \]
\[ \lambda(P^{-1}A_{CN}) \]

Sparse Days, CERFACS, 2014 – p.9/20
\[ \lambda(P^{-1}A_{BE}) \]
\[ \lambda(P^{-1}A_{CN}) \]
For $N=40$: 

![Graph showing the normalised residual over iteration number for GMRES. The y-axis represents the normalised residual on a logarithmic scale, ranging from $10^{-14}$ to $10^0$, and the x-axis represents the iteration number from 0 to 45. The graph shows a significant decrease in the normalised residual as the iteration number increases, indicating the convergence of the GMRES method.]
Thus: $N$ V-cycles for each of $N$ GMRES iterations—hence $N^2 (> Nr)$ overall.

**BUT** with $N$ processors, solution with $P$ is (embarrassingly) parallel—block diagonal $\Rightarrow$ independent computation.

Thus parallel effort is $N < Nr$ (= sequential effort).
For another problem:
Large $N$ means increasing work (of Arnoldi orthogonalisation) with GMRES: use BiCGSTAB:

![Graph showing the normalised residual versus iteration number with BiCGSTAB line]

The graph illustrates the normalised residual against the iteration number for the BiCGSTAB method.
For another problem:
## Backwards Euler

<table>
<thead>
<tr>
<th>$h$</th>
<th>$\tau$</th>
<th>N</th>
<th>DoF</th>
<th>GMRES</th>
<th>BiCGStab</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^{-3}$</td>
<td>$2^{-3}$</td>
<td>40</td>
<td>3240</td>
<td>40</td>
<td>38</td>
</tr>
<tr>
<td>$2^{-4}$</td>
<td>$2^{-4}$</td>
<td>80</td>
<td>23120</td>
<td>80</td>
<td>78</td>
</tr>
<tr>
<td>$2^{-5}$</td>
<td>$2^{-5}$</td>
<td>160</td>
<td>174240</td>
<td>160</td>
<td>157</td>
</tr>
<tr>
<td>$2^{-3}$</td>
<td>$2^{-3}$</td>
<td>40</td>
<td>3240</td>
<td>40</td>
<td>38</td>
</tr>
<tr>
<td>$2^{-4}$</td>
<td>$2^{-4}$</td>
<td>40</td>
<td>11560</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>$2^{-5}$</td>
<td>$2^{-5}$</td>
<td>40</td>
<td>43560</td>
<td>40</td>
<td>43</td>
</tr>
<tr>
<td>$2^{-6}$</td>
<td>$2^{-6}$</td>
<td>40</td>
<td>169000</td>
<td>43</td>
<td>44</td>
</tr>
<tr>
<td>$2^{-7}$</td>
<td>$2^{-7}$</td>
<td>40</td>
<td>665640</td>
<td>45</td>
<td>45</td>
</tr>
<tr>
<td>$2^{-8}$</td>
<td>$2^{-8}$</td>
<td>40</td>
<td>2641960</td>
<td>46</td>
<td>45</td>
</tr>
</tbody>
</table>
# Crank-Nicholson

<table>
<thead>
<tr>
<th>$h$</th>
<th>$\tau$</th>
<th>$N$</th>
<th>DoF</th>
<th>GMRES</th>
<th>BiCGStab</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^{-3}$</td>
<td>$2^{-5}$</td>
<td>32</td>
<td>2592</td>
<td>33</td>
<td>35</td>
</tr>
<tr>
<td>$2^{-4}$</td>
<td>$2^{-6}$</td>
<td>64</td>
<td>18496</td>
<td>66</td>
<td>68</td>
</tr>
<tr>
<td>$2^{-5}$</td>
<td>$2^{-7}$</td>
<td>128</td>
<td>139392</td>
<td>132</td>
<td>138</td>
</tr>
<tr>
<td>$2^{-3}$</td>
<td>$2^{-5}$</td>
<td>32</td>
<td>2592</td>
<td>34</td>
<td>35</td>
</tr>
<tr>
<td>$2^{-4}$</td>
<td>$2^{-6}$</td>
<td>32</td>
<td>9248</td>
<td>35</td>
<td>34</td>
</tr>
<tr>
<td>$2^{-5}$</td>
<td>$2^{-7}$</td>
<td>32</td>
<td>34848</td>
<td>37</td>
<td>35</td>
</tr>
<tr>
<td>$2^{-6}$</td>
<td>$2^{-8}$</td>
<td>32</td>
<td>135200</td>
<td>39</td>
<td>36</td>
</tr>
<tr>
<td>$2^{-7}$</td>
<td>$2^{-9}$</td>
<td>32</td>
<td>532512</td>
<td>40</td>
<td>38</td>
</tr>
<tr>
<td>$2^{-8}$</td>
<td>$2^{-10}$</td>
<td>32</td>
<td>2113568</td>
<td>38</td>
<td>39</td>
</tr>
</tbody>
</table>
Summary

For a simple linear PDE problem our proposal should achieve

\[ N \text{ work on } N \text{ processors} \quad (N^2 \text{ work on 1 processor}) \]

compared to

\[ Nr \text{ work for the standard sequential algorithm} \]

\[ \frac{1}{2} Nr \text{ work (?) for Parareal (?)} \]
Acknowledgement

This work is partially supported by Award No. KUK-C1-013-04 made by King Abdullah University of Science and Technology (KAUST)