Fast Integral Equation Methods for BIOEM in the Petascale Era: Kilo Cores, Mega Codes, Giga Equations, Tera Bytes, and Peta Multiplications

Ali E. YILMAZ

Department of Electrical & Computer Engineering
University of Texas at Austin

University of Illinois at Urbana Champaign, Urbana, IL, Apr. 19, 2011
Outline

• Motivation
  - Body-Centric Wireless Systems
  - Historical Perspective: Computational Bioelectromagnetics
  - Computing Trends

• Background
  - FDTD vs. MOM
  - Pre-corrected FFT/Adaptive Integral Method
  - Computational Complexity

• Parallelization
  - General Limits of Parallelization
  - Traditional Scheme and Its Limits
  - Proposed Parallelization
  - Comparison

• BIOEM Problems
  - High-Resolution Simulations
  - AustinMan Antrophomorphic Human Model
Body-Centric Wireless Systems

- More functionality near-, on-, in-body
- Wide-spread use
- Marketing
- Health concerns

Images from news articles in newscientist.com, switched.com, pulse2.com
Computational BIOEM Requirements

- **Phantoms** ("tissue-equivalent" liquids)
  - Resolve wavelength/skin depth in material
    \[ \frac{\lambda}{10} \approx 5.4 \text{ mm}, \frac{\delta}{10} \approx 3.6 \text{ mm} \ (900 \text{ MHz}) \]

- **Anthropomorphic Human Models**
  - Resolve wavelength/skin depth in tissues

<table>
<thead>
<tr>
<th>Tissue Name</th>
<th>Conductivity [S/m]</th>
<th>Relative Permittivity</th>
<th>Loss Tangent</th>
<th>Wavelength [mm]</th>
<th>Skin Depth [mm]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CerebroSpinalFluid</td>
<td>2.4126</td>
<td>68.638</td>
<td>0.70202</td>
<td>38.147</td>
<td>19.215</td>
</tr>
<tr>
<td>BoneCortical</td>
<td>0.14331</td>
<td>12.454</td>
<td>0.22984</td>
<td>93.782</td>
<td>131.57</td>
</tr>
<tr>
<td>BoneMarrow</td>
<td>0.040208</td>
<td>5.5043</td>
<td>0.1459</td>
<td>141.61</td>
<td>310.58</td>
</tr>
<tr>
<td>BrainGrayMatter</td>
<td>0.94227</td>
<td>52.725</td>
<td>0.35694</td>
<td>45.182</td>
<td>41.536</td>
</tr>
<tr>
<td>Muscle</td>
<td>0.94294</td>
<td>55.032</td>
<td>0.34222</td>
<td>44.277</td>
<td>42.355</td>
</tr>
</tbody>
</table>

\[ \frac{\lambda}{10} \approx 3.8 \text{ mm}, \frac{\delta}{10} \approx 1.9 \text{ mm} \quad \text{Finite element size} \sim 10-100 \text{ mm}^3 \]

- Resolve tissue boundaries
  Finite element size \( \sim 0.1-1 \text{ mm}^3 \) \( N \sim 1-10 \times 10^6 \)

“Our numerical computation was performed with a CDC 6500 computer which has a maximum capacity of inverting a 120x120 matrix. This limits the maximum number of cells to 40...A cell size of 10 cm$^3$ seems appropriate.”


“A total of 180 cubical cells of various sizes was used to obtain a best fit of the contour on diagrams of the 50th percentile standard man...the matrix is 270 by 270...”

"It is planned to extend the FFT method to 3-D problems. Thus far, the only technique available...for models of man...is the MOM."

"An obvious difficulty...is the creation of the model...Another problem to be solved is the presentation of the results."


"...serious errors exist in solutions for the TE polarization...fictitious line charge sources...have been removed...these modifications prevent...the use of the FFT-CGM and thus require O(N^3) operations."

"In contrast, FD-TD...yields excellent solutions...FD-TD has great potential for solving the high-resolution models needed in bioelectromagnetics."

“FDTD is used to calculate the detailed SAR within the human body...Comparison...with the method of moments.”


“Different numerical head phantoms based on MRI scans of three different adults were used with voxel sizes down to 1 mm³...up to 13 tissue types. The numerical results are compared with those of measurements in a multitissue phantom...”


“...major efforts...to improve the numerical dosimetry...in particular the FDTD has been improved and adapted to this domain.”
Computational BIOEM
90s and 00s: FDTD Controversies


“Are children more vulnerable to the microwave radiation from cellular mobile telephones?...Numerical dosimetry...one of the most productive research areas during the past decade...Instead of providing a consensus, these SAR studies have spawned diverse opinions.”


“Comparing localized SAR values determined experimentally...with those obtained using the FDTD method demonstrate good agreement, except when the FDTD method calculates “hot or cold” spots (relative to whole-body average SAR).”


“...induced SAR...in the human head due to cellular phone radiation...for heterogeneous and homogeneous HUGO models. The discrepancy of the results, concerning the location of hot spots...necessity of using high-resolution anatomy models for realistic simulations.”


“CT data sets are transformed...FDTD-method is the applied on a cubic lattice (voxel model) or a tetrahedron grid (region-based segmentation)...For comparison, the VSIE method is performed on the same tetrahedron grid...power deposition patterns in the interior of the patient depended strongly on the models...Due to its shorter calculation time, the FDTD method is currently used in the clinic. Predictions...without prior corrections of tissue specifications are not always supported by clinical experiences.”


“Numerous numerical dosimetric studies have been published about the exposure of mobile phone users which concluded with conflicting results. However, many of these studies lack reproducibility due to shortcomings in the description of the phone position.”

“…while the moment-method, using pulse basis functions, gives good values for whole-body average SAR, the convergence of the solutions for SAR distributions is questionable…Pulse functions should be replaced by a better approximation.”


“Special basis functions are defined within tetrahedral volume elements to insure that the normal electric field satisfies the correct jump condition at interfaces between different dielectric media.”


“A weak form of the integral equation…is obtained by testing it with appropriate testing functions. In view of the partial derivatives…the volumetric rooftop functions are chosen as testing and expansion functions.”


“The use of pulse basis functions yielded slow convergence and poor results when dealing with materials with high dielectric contrast. Better formulations were later proposed…most use mixed-order basis functions and Galerkin’s testing.”

“…BCG-FFT method reduces the memory requirement to O(N) and computational complexity to O($N_{\text{iter}}^2 \log N$), making the problem solvable on a workstation. For lossy material…$N_{\text{iter}}$ is found to be small.”
Historical Perspective: Supercomputing

Top Supercomputers (top500.org)

1. Tian-He 1A
   - Intel X5670
   - 6 Core 2930 MHz

2. Jaguar
   - AMD x86_64 Opteron
   - 6 Core 2600 MHz

15. Ranger
   - AMD x86_64 Opteron
   - 4 Core 2300 MHz

Blue Waters (this year)
   - IBM POWER7 processor
   - 8 Core
Historical Perspective: Limits of Serial Computing

Trends in Computation

Data collected by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, C. Batten, and D. Wentzlaff
BACKGROUND

- FDTD vs. MOM
- Pre-corrected FFT/Adaptive Integral Method
- Computational Complexity
FDTD vs. MOM for BIOEM

- **FDTD**
  + Historical momentum/easy to code
  + Matrix free
  - Uniform grids/staircasing
  - Special grid truncation
  - Extended grids
  - Courant stability condition
  - Phase dispersion
  - Time-domain (transients)

- **MOM+FFT-based Algorithm**
  - Rarer/harder to code
  - Dense matrix (+ fast algorithm)
  + Arbitrary mesh
  + Exact radiation condition
  + Never mesh free space
  + Unknowns only in space
  + Incorrect Green function phase speed
  + Frequency-domain (dispersive materials)

Simulation Time: $O(N_i N')$
Memory: $O(N')$

Simulation Time: $O(N_{iter} N \log N)$
Memory: $O(N)$

For the same accuracy: $N' >> N$
For $N' = N$: $\text{err}(\text{FDTD}) >> \text{err}(\text{MOM})$
FDTD vs. MOM for BIOEM

- Controversial results
- Increased expectations
- New applications
- Curiosity
- Connroversial results
- Computational research
- Curiosity
- Vanilla FDTD

- Increases in Raw Computer Power
- Vanilla MOM

Error

Computational results

Time/Memory Cost
Volume Electric Field Integral Equation

- Time-harmonic VEFIE for Inhomogeneous and Lossy Dielectric Scatterer

\[ \varepsilon_0, \mu_0 \]
\[ \varepsilon(r), \sigma(r), \mu_0 \]
\[ \frac{1}{2} = \int J(r) \]
\[ \varepsilon_0, \mu_0 \]

\[ \varepsilon(r) = \varepsilon(r) + \frac{\sigma(r)}{j\omega} \]
\[ \tilde{D}(r) = \varepsilon(r)E(r) \]
\[ J(r) = \left( 1 - \frac{\varepsilon_0}{\varepsilon(r)} \right) j\omega \tilde{D}(r) \]

\[ \varepsilon_0, \mu_0 \]

\[ \varepsilon(r), \sigma(r), \mu_0 \]
\[ \frac{1}{2} = \int J(r) \]
\[ \varepsilon_0, \mu_0 \]

\[ E^{\text{inc}}(r) = E(r) - E^{\text{sc}}(r) = E(r) + j\omega A(r) + \nabla \phi(r) \]

\[ \begin{cases} A(r) \\ \phi(r) \end{cases} = \int \int \int_{V} G(r, r') \begin{pmatrix} \mu_0 J(r') \\ -\nabla' \cdot J(r') / (j\omega \varepsilon_0) \end{pmatrix} dv', \quad G(r, r') = \frac{e^{-jk_0|r-r'|}}{4\pi|r-r'|} \]

\[ E^{\text{inc}}(r) = \frac{\tilde{D}(r)}{\varepsilon(r)} + (j\omega)^2 \mu_0 \int \int \int_{V} G(r, r') \kappa(r')\tilde{D}(r')dv' - \nabla \int \int \int_{V} \frac{G(r, r')\nabla' \cdot \kappa(r')\tilde{D}(r')}{\varepsilon_0} dv' \quad r \in V \]
Method of Moments

• MOM

\[
E^{inc}(\mathbf{r}) = \frac{\tilde{D}(\mathbf{r})}{\tilde{\varepsilon}(\mathbf{r})} + (j\omega)^2 \mu_0 \int \int \int G(\mathbf{r}, \mathbf{r'}) \kappa(\mathbf{r'}) \tilde{D}(\mathbf{r'}) d\mathbf{r'} - \nabla \int \int \int \frac{G(\mathbf{r}, \mathbf{r'}) \nabla' \cdot \kappa(\mathbf{r'}) \tilde{D}(\mathbf{r'})}{\varepsilon_0} d\mathbf{r'} \quad \mathbf{r} \in V
\]

- Discretize \& Galerkin test

\[
\tilde{D}(\mathbf{r}) \approx \sum_{k'=1}^{N} I_{k'} V_{k'}(\mathbf{r}) \quad \mathbf{J}(\mathbf{r}) \approx j\omega \sum_{k'=1}^{N} \kappa_{k'} I_{k'} V_{k'}(\mathbf{r})
\]

\[
\int \int \int d\mathbf{r} \kappa_{k}(\mathbf{r}) V_{k}(\mathbf{r}) \cdot \text{VEFIE} \quad \text{for } k = 1, \ldots, N
\]

- Fill matrix \& solve

\[
Z_{N \times N} \mathbf{I} = \mathbf{F}^{inc}
\]

• Computational Cost

Matrix fill time: \(O(N^2)\)
Memory: \(O(N^2)\)
Time per iteration: \(O(N^2)\)

Pre-corrected FFT/Adaptive Integral Method

Embed $V$ in a regular grid with $N_C$ nodes

Step 1: Project/Anterpolate

Step 2: Propagate

Step 3: Interpolate

Step 4: Correct

$Z_I \approx Z_{\text{near}} I + \Lambda^T \text{IFFT}\{\text{FFT}[G] \cdot \text{FFT}[\Lambda I]\}$

Correction region size: $\gamma = 1$

- Computational Cost
  
  Matrix fill time: $O(N_{\text{near}} + N^C + pN)$

  Memory: $O(N_{\text{near}} + N^C + pN)$

  Time per iteration: $O(N_{\text{near}} + pN + N^C \log N^C + pN)$
## Benchmark Problems

<table>
<thead>
<tr>
<th>CAD Sphere Phantom</th>
<th>Voxel-Based Sphere Phantom</th>
<th>AustinMan Partial Body</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta V$ (mm$^3$)</td>
<td>$l$ (mm)</td>
<td>$N$</td>
</tr>
<tr>
<td>1030</td>
<td>21</td>
<td>$\sim 10^4$</td>
</tr>
<tr>
<td>15</td>
<td>5</td>
<td>$\sim 6 \times 10^5$</td>
</tr>
<tr>
<td>0.3</td>
<td>1.4</td>
<td>$\sim 3.2 \times 10^7$</td>
</tr>
<tr>
<td></td>
<td>$\zeta / l$</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>2.4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>9.8</td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>$\delta / l$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$N$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\sim 10^4$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\sim 6 \times 10^5$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\sim 3.2 \times 10^7$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\Delta V$ (mm$^3$)</th>
<th>$l$ (mm)</th>
<th>$N$</th>
</tr>
</thead>
<tbody>
<tr>
<td>683</td>
<td>20</td>
<td>$\sim 1.5 \times 10^4$</td>
</tr>
<tr>
<td>11</td>
<td>5</td>
<td>$\sim 9 \times 10^5$</td>
</tr>
<tr>
<td>0.2</td>
<td>1.3</td>
<td>$\sim 5.6 \times 10^7$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\delta / l$</th>
<th>$\lambda / l$</th>
<th>$N$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.7</td>
<td>2.5</td>
<td>$\sim 1.5 \times 10^4$</td>
</tr>
<tr>
<td>7.1</td>
<td>2.5</td>
<td>$\sim 9 \times 10^5$</td>
</tr>
<tr>
<td>29</td>
<td>1.3</td>
<td>$\sim 5.6 \times 10^7$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CAD Sphere Phantom</th>
<th>Voxel-Based Sphere Phantom</th>
<th>AustinMan Partial Body</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\epsilon_r = 41.5, \sigma = 0.97$</td>
<td>$\epsilon_r = 41.5, \sigma = 0.97$</td>
<td>$f = 900$ MHz</td>
</tr>
<tr>
<td>radius=104 mm</td>
<td>radius=104 mm</td>
<td>$500 \times 300 \times 350$ mm$^3$</td>
</tr>
</tbody>
</table>
Definitions for Post-Processing

Relative RCS Error

\[
\text{Relative RCS Error} = \sqrt{\int_0^{2\pi} \int_0^\pi \left( \sigma_{\theta\theta}^{\text{AIM}} - \sigma_{\theta\theta}^{\text{MIE}} \right)^2 \sin \theta d\theta d\phi}
\]

\[
\langle P_{\text{absorbed}}(r, f) \rangle = \frac{1}{2} \int_0^{1/f} \sigma(r) \mathbf{E}(r, f) \cdot \mathbf{E}(r, f) dt
\]

\[
= \frac{1}{2} \sigma(r) \mathbf{E}(r, f) \cdot \mathbf{E}^*(r, f)
\]

\[
\begin{align*}
\text{radius} &= 104 \text{ mm} \\
\varepsilon_r &= 41.5, \sigma = 0.97
\end{align*}
\]

\[
\begin{align*}
\text{radius} &= 104 \text{ mm} \\
\varepsilon_r &= 41.5, \sigma = 0.97
\end{align*}
\]

\[
\begin{align*}
\text{radius} &= 104 \text{ mm} \\
\varepsilon_r &= 41.5, \sigma = 0.97
\end{align*}
\]

\[
\begin{align*}
\text{radius} &= 104 \text{ mm} \\
\varepsilon_r &= 41.5, \sigma = 0.97
\end{align*}
\]

\[
f = 900 \text{ MHz}
\]

\[
500 \times 300 \times 350 \text{ mm}^3
\]
Accuracy

0.2 mm³

11 mm³

683 mm³

0.3 mm³

15 mm³

1030 mm³

\[ \langle P_{\text{absorbed}} \rangle \]

Relative RCS Error

Average Tetrahedron Volume (mm³)

\[ \langle P_{\text{absorbed}} \rangle \]
Computational Costs

\[ \Delta V = 683 \text{ mm}^3 \]

\[ \Delta V = 0.2 \text{ mm}^3 \]
Need for Parallelization:
Minimum $P$ for Different Constraints

Matrix Fill Time

Memory

Matrix Solve Time

Different Constraints
Parallelization

- General Limits
- Traditional Scheme and Its Limitations
- Proposed Scheme
- Comparison
General Limits of Parallelization

- **(Hard) Scalability Limitations**
  1. Sequential part of the code (Amdahl’s Law)
  2. Load imbalance/Can’t divide computations further (run out of parallelism)
  3. Communication latency/bandwidth
Traditional Parallelization Scheme

- **Single Workload Distribution Strategy** [1]
  1. Project/Anterpolate: 1-D Slab
  2. Propagate: 1-D Slab
  3. Interpolate: 1-D Slab
  4. Correct: 1-D Slab
  5. Communicate Near

\[
P = P^{xy} = 6 \\
p_{\text{max}}^{1D} = \min(N_{cx}, N_{cy})
\]

Traditional Parallelization Scheme

- Single Workload Distribution Strategy [1]
  1. Project/Anterpolate: 1-D Slab
  2. Propagate: 1-D Slab
  3. Interpolate: 1-D Slab
  4. Correct: 1-D Slab
  5. Communicate Near

\[
P = P^{xy} = 6 \\
P_{1D}^{\text{max}} = \min(N^{x}, N^{y})
\]

Traditional Parallelization Scheme

• Single Workload Distribution Strategy
  1. Project/Anterpolate: 1-D Slab
  2. Propagate: 1-D Slab
     a) \( \text{FFT}_{ZY} \) – Transpose – \( \text{FFT}_X \)
     b) Multiply
     c) \( \text{IFFT}_X \) – Transpose – \( \text{IFFT}_{YZ} \)
  3. Interpolate: 1-D Slab
  4. Correct: 1-D Slab
  5. Communicate Near

\[ ZI \simeq \Lambda I \]

Memory: \( O\left(\frac{pN}{??}\right) \)

CPU time per iteration: \( O\left(\frac{pN}{??}\right) \)

Comm time per iteration: \( O(??) \)
Traditional Parallelization Scheme

- **Single Workload Distribution Strategy**
  1. Project/Anterpolate: 1-D Slab
  2. Propagate: 1-D Slab
     a) $\text{FFT}_{ZY} - \text{Transpose} - \text{FFT}_X$
     b) Multiply
     c) $\text{IFFT}_X - \text{Transpose} - \text{IFFT}_{YZ}$
  3. Interpolate: 1-D Slab
  4. Correct: 1-D Slab
  5. Communicate Near

\[ Z^I \approx \text{FFT}[\Lambda I] \]

Memory: \( O(\frac{N^C}{P_{\text{max}}^{1D}} + \frac{pN}{??}) \)

CPU time per iteration: \( O(\frac{N^C \log N^C}{P_{\text{max}}^{1D}} + \frac{pN}{??}) \)

Comm time per iteration: \( O(??) \)
Traditional Parallelization Scheme

- Single Workload Distribution Strategy
  1. Project/Anterpolate: 1-D Slab
  2. Propagate: 1-D Slab
     a) $\text{FFT}_{ZY} - \text{Transpose} - \text{FFT}_{X}$
     b) Multiply
     c) $\text{IFFT}_{X} - \text{Transpose} - \text{IFFT}_{YZ}$
  3. Interpolate: 1-D Slab
  4. Correct: 1-D Slab
  5. Communicate Near

$$ZI \approx \text{FFT}[A1]$$

Memory: $O(\frac{N_C^C}{P_{1D\,\max}} + \frac{pN}{??})$

CPU time per iteration: $O(\frac{N_C^C \log N_C^C}{P_{1D\,\max}} + \frac{pN}{??})$

Comm time per iteration: $O(\frac{p_{1D}}{P_{\max}})$
Traditional Parallelization Scheme

- Single Workload Distribution Strategy
  1. Project/Anterpolate: 1-D Slab
  2. Propagate: 1-D Slab
     a) \( \text{FFT}_{ZY} \) – Transpose – \( \text{FFT}_X \)
     b) Multiply
     c) \( \text{IFFT}_X \) – Transpose – \( \text{IFFT}_{YZ} \)
  3. Interpolate: 1-D Slab
  4. Correct: 1-D Slab
  5. Communicate Near

\[ ZI \approx \text{FFT}[\Lambda I] \]

Memory: \( O\left(\frac{N^C}{P_{1D_{\max}}} + \frac{pN}{??}\right) \)

CPU time per iteration: \( O\left(\frac{N^C \log N^C}{P_{1D_{\max}}} + \frac{pN}{??}\right) \)

Comm time per iteration: \( O\left(\frac{P_{1D_{\max}}}{??}\right) \)
Traditional Parallelization Scheme

- **Single Workload Distribution Strategy**
  1. Project/Anterpolate: 1-D Slab
  2. Propagate: 1-D Slab
     a) $\text{FFT}_{ZY} – \text{Transpose} – \text{FFT}_X$
     b) Multiply
     c) $\text{IFFT}_X – \text{Transpose} – \text{IFFT}_{YZ}$
  3. Interpolate: 1-D Slab
  4. Correct: 1-D Slab
  5. Communicate Near

\[
\text{ZI} \approx \text{FFT}[G] \cdot \text{FFT}[AI]
\]

**Memory:**

\[
O\left(\frac{N^C}{P_{\text{max}}^{1D}} + \frac{pN}{??}\right)
\]

**CPU time per iteration:**

\[
O\left(\frac{N^C \log N^C}{P_{\text{max}}^{1D}} + \frac{pN}{??}\right)
\]

**Comm time per iteration:**

\[
O\left(\frac{P_{1D}}{\text{max}}\right)
\]
Traditional Parallelization Scheme

- **Single Workload Distribution Strategy**
  1. **Project/Anterpolate: 1-D Slab**
  2. **Propagate: 1-D Slab**
     a) $\text{FFT}_{ZY} - \text{Transpose} - \text{FFT}_X$
     b) Multiply
     c) $\text{IFFT}_X - \text{Transpose} - \text{IFFT}_{YZ}$
  3. **Interpolate: 1-D Slab**
  4. **Correct: 1-D Slab**
  5. **Communicate Near**

$$Z \mathbf{i} \approx \text{IFFT}\{\text{FFT}[\mathbf{G}] \cdot \text{FFT}[\mathbf{AI}]\}$$

**Memory:** $O(\frac{N^C}{P_{\text{max}}^{1D}} + pN)$

**CPU time per iteration:** $O(\frac{N^C \log N^C}{P_{\text{max}}^{1D}} + \frac{pN}{??})$

**Comm time per iteration:** $O(\frac{P_{\text{max}}^{1D}}{??})$
Traditional Parallelization Scheme

- Single Workload Distribution Strategy
  1. Project/Anterpolate: 1-D Slab
  2. Propagate: 1-D Slab
     a) $\text{FFT}_Z$ – Transpose – $\text{FFT}_X$
     b) Multiply
     c) $\text{IFFT}_X$ – Transpose – $\text{IFFT}_{YZ}$
  3. Interpolate: 1-D Slab
  4. Correct: 1-D Slab
  5. Communicate Near

$$Z_I \approx \text{IFFT}\{\text{FFT}[G] \cdot \text{FFT}[\Lambda I]\}$$

Memory: $O\left(\frac{N^C}{P^{1D}_{\text{max}}} + \frac{pN}{??}\right)$

CPU time per iteration: $O\left(\frac{N^C \log N^C}{P^{1D}_{\text{max}}} + \frac{pN}{??}\right)$

Comm time per iteration: $O\left(\frac{P^{1D}_{\text{max}}}{??}\right)$
Traditional Parallelization Scheme

- **Single Workload Distribution Strategy**
  1. **Project/Anterpolate:** 1-D Slab
  2. **Propagate:** 1-D Slab
     a) \( \text{FFT}_{ZY} \) – Transpose – \( \text{FFT}_X \)
     b) Multiply
     c) \( \text{IFFT}_X \) – Transpose – \( \text{IFFT}_{YZ} \)
  3. Interpolate: 1-D Slab
  4. Correct: 1-D Slab
  5. Communicate Near

\[
ZI \approx \text{IFFT}\{\text{FFT}[G] \cdot \text{FFT}[AI]\}
\]

**Memory:** \( O\left(\frac{N^C}{P_{\text{max}}^{1D}} + \frac{pN}{??}\right) \)

**CPU time per iteration:** \( O\left(\frac{N^C \log N^C}{P_{\text{max}}^{1D}} + \frac{pN}{??}\right) \)

**Comm time per iteration:** \( O\left(\frac{P_{\text{max}}^{1D}}{??}\right) \)
Traditional Parallelization Scheme

- Single Workload Distribution Strategy
  1. Project/Anterpolate: 1-D Slab
  2. Propagate: 1-D Slab
     a) $\text{FFT}_{ZY} - \text{Transpose} - \text{FFT}_X$
     b) Multiply
     c) $\text{IFFT}_X - \text{Transpose} - \text{IFFT}_{YZ}$
  3. Interpolate: 1-D Slab
  4. Correct: 1-D Slab
  5. Communicate Near

\[ Z I \approx \Lambda^T \text{IFFT} \{ \text{FFT} [G] \cdot \text{FFT} [_A \Lambda I] \} \]

Memory: \( O(\frac{pN}{??} + \frac{N^C}{P_{\text{max}}^{1D}} + \frac{pN}{??}) \)

CPU time per iteration: \( O(\frac{pN}{??} + \frac{N^C \log N^C}{P_{\text{max}}^{1D}} + \frac{pN}{??}) \)

Comm time per iteration: \( O(\frac{P_{\text{max}}^{1D}}{??}) \)
Traditional Parallelization Scheme

- Single Workload Distribution Strategy
  1. Project/Anterpolate: 1-D Slab
  2. Propagate: 1-D Slab
     a) FFT$_{ZY}$ – Transpose – FFT$_X$
     b) Multiply
     c) IFFT$_X$ – Transpose – IFFT$_{YZ}$
  3. Interpolate: 1-D Slab
  4. Correct: 1-D Slab
  5. Communicate Near

\[ Z_I \approx Z_{\text{near}} I + \Lambda^T \text{IFFT}\{\text{FFT}[G] \cdot \text{FFT}[A_I]\} \]

Memory: \( O(\frac{N_{\text{near}}}{??} + \frac{pN}{??} + \frac{N^C}{P_{\text{max}}^{1D}} + \frac{pN}{??}) \)

CPU time per iteration: \( O(\frac{N_{\text{near}}}{??} + \frac{pN}{??} + \frac{N^C \log N^C}{P_{\text{max}}^{1D}} + \frac{pN}{??}) \)

Comm time per iteration: \( O(\frac{P_{\text{max}}^{1D}}{??}) \)
Traditional Parallelization Scheme

- **Single Workload Distribution Strategy**
  1. **Project/Anterpolate**: 1-D Slab
  2. **Propagate**: 1-D Slab
     a) \( \text{FFT}_{ZY} \) – Transpose – \( \text{FFT}_X \)
     b) Multiply
     c) \( \text{IFFT}_X \) – Transpose – \( \text{IFFT}_{YZ} \)
  3. **Interpolate**: 1-D Slab
  4. **Correct**: 1-D Slab
  5. **Communicate Near**

\[
ZI \approx Z^{\text{near}} + \Lambda^T \text{IFFT}\{\text{FFT}[G] \cdot \text{FFT}[\Lambda I]\}
\]

Memory: \( O\left(\frac{N^{\text{near}}}{??} + \frac{pN}{??} + \frac{N^C}{P^{1D}_{\text{max}}} + \frac{pN}{??}\right) \)

CPU time per iteration: \( O\left(\frac{N^{\text{near}}}{??} + \frac{pN}{??} + \frac{N^C \log N^C}{P^{1D}_{\text{max}}} + \frac{pN}{??}\right) \)

Comm time per iteration: \( O(C^{\text{near}} + P^{1D}_{\text{max}}) \)
Limits of Traditional Parallelization Scheme

1. Anterpolation, Correction, Interpolation computations not balanced

2. “Grid limited”  
   \[ P_{\text{max}}^{1D} = \min(N^x, N^y) \]

![Graphs showing Wall Clock Time vs. P](image1)

![Graphs showing Memory/Core (MB) vs. P](image2)
General Limits of Parallelization

(Hard) Scalability Limitations

1. Sequential part of the code (Amdahl’s Law)

2. Load imbalance/Can’t divide computations further (run out of parallelism)

3. Communication latency/bandwidth
Limits of Traditional Parallelization

Maximum $P$ and Maximum $N$

Matrix Fill Time

Memory

Matrix Solve Time

radius = 104 mm

$\varepsilon_r = 41.5, \sigma = 0.97$
Parallelization

- General Limits
- Traditional Scheme and Its Limitations
- Proposed Scheme
- Comparison
Proposed Parallelization Scheme

• Two Workload Distribution Strategies [1]
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute

Proposed Parallelization Scheme

• Two Workload Distribution Strategies
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
     a) Shift Balance in X Dimension
     b) Shift Balance in Y Dimension
     c) Shift Balance in Z Dimension
  5. Communicate Near/Redistribute
Proposed Parallelization Scheme

- **Two Workload Distribution Strategies**
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
     a) Shift Balance in X Dimension
     b) Shift Balance in Y Dimension
     c) Shift Balance in Z Dimension
  5. Communicate Near/Redistribute
Proposed Parallelization Scheme

• **Two Workload Distribution Strategies**
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. **Correct: 3-D Block**
     a) Shift Balance in X Dimension
     b) Shift Balance in Y Dimension
     c) Shift Balance in Z Dimension
  5. Communicate Near/Redistribute
Proposed Parallelization Scheme

- **Two Workload Distribution Strategies**
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
     a) Shift Balance in X Dimension
     b) Shift Balance in Y Dimension
     c) Shift Balance in Z Dimension
  5. Communicate Near/Redistribute
Proposed Parallelization Scheme

- Two Workload Distribution Strategies
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute

\[ P = P^{xy} \times P^{yz} = 3 \times 2 \]
\[ P_{\text{max}}^{2D} = P_{\text{max}}^{xy} \times P_{\text{max}}^{yz} \]
\[ = \min(N_{\text{cx}}, N_{\text{cy}}) \times \min(N_{\text{cy}}, N_{\text{cz}}) \]
Proposed Parallelization Scheme

• Two Workload Distribution Strategies

1. Project/Anterpolate: 2-D Pencil
2. Propagate: 2-D Pencil
3. Interpolate: 2-D Pencil
3.5. Redistribute
4. Correct: 3-D Block
5. Communicate Near/Redistribute

\[
P = P^{xy} \times P^{yz} = 3 \times 2
\]

\[
P_{\text{max}}^{2D} = P_{\text{max}}^{xy} \times P_{\text{max}}^{yz}
= \min(N^{cx}, N^{cy}) \times \min(N^{cy}, N^{cz})
\]
Proposed Parallelization Scheme

- Two Workload Distribution Strategies
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute

\[ P = P^{xy} \times P^{yz} = 3 \times 2 \]
\[ P_{\text{max}}^{2D} = P_{\text{max}}^{xy} \times P_{\text{max}}^{yz} = \min(N_{cx}, N_{cy}) \times \min(N_{cy}, N_{cz}) \]
Proposed Parallelization Scheme

- Two Workload Distribution Strategies
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
  3. Interpolate: 2-D Pencil
  3.5. **Redistribute**
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute
Proposed Parallelization Scheme

- **Two Workload Distribution Strategies**
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
  3. Interpolate: 2-D Pencil
  3.5. **Redistribute**
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute
• **Two Workload Distribution Strategies**
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
  5. **Communicate Near/Redistribute**
Proposed Parallelization Scheme

- Two Workload Distribution Strategies
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
     a) $\text{FFT}_Z - ZY$ Transpose – $\text{FFT}_Y$
        – YX Transpose – $\text{FFT}_X$
     b) Multiply
     c) $\text{IFFT}_X – XY$ Transpose – $\text{IFFT}_Y$
        – YZ Transpose – $\text{IFFT}_Z$
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute

$$ZI \approx \Lambda I$$

Memory: $O\left(\frac{pN}{??}\right)$

CPU time per iteration: $O\left(\frac{pN}{??}\right)$

Comm time per iteration: $O\left(??\right)$
Proposed Parallelization Scheme

• Two Workload Distribution Strategies
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
     a) $\text{FFT}_z – ZY \text{ Transpose} – \text{FFT}_y$
        – $YX \text{ Transpose} – \text{FFT}_x$
     b) Multiply
     c) $\text{IFFT}_x – XY \text{ Transpose} – \text{IFFT}_y$
        – $YZ \text{ Transpose} – \text{IFFT}_z$
  3. Interpolate: 2-D Pencil
  3.5. Distribute
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute

$Z I \approx \text{FFT}[\Lambda I]$

Memory: $O\left( \frac{N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??} \right)$

CPU time per iteration: $O\left( \frac{N^C \log N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??} \right)$

Comm time per iteration: $O(??)$
Proposed Parallelization Scheme

- **Two Workload Distribution Strategies**

  1. **Project/Anterpolate**: 2-D Pencil
  2. **Propagate**: 2-D Pencil
     a) \( \text{FFT}_Z - ZY \text{Transpose} - \text{FFT}_Y \)
     b) \( \text{YX} \text{Transpose} - \text{FFT}_X \)
     c) \( \text{IFFT}_X - XY \text{Transpose} - \text{IFFT}_Y \)
     \( - YZ \text{Transpose} - \text{IFFT}_Z \)
  3. **Interpolate**: 2-D Pencil
  3.5. **Redistribute**
  4. **Correct**: 3-D Block
  5. **Communicate Near/Redistribute**

\[ ZI \approx \text{FFT}[\Lambda I] \]

**Memory:** \( O\left( \frac{N^C}{P^{xy}_{\text{max}} P^{yz}_{\text{max}}} + \frac{pN}{??} \right) \)

**CPU time per iteration:** \( O\left( \frac{N^C \log N^C}{P^{xy}_{\text{max}} P^{yz}_{\text{max}}} + \frac{pN}{??} \right) \)

**Comm time per iteration:** \( O\left( \frac{P}{P^{xy}_{\text{max}}} \right) \)
Proposed Parallelization Scheme

• Two Workload Distribution Strategies
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
     a) FFT\(_z\) – ZY Transpose – FFT\(_y\)
        – YX Transpose – FFT\(_x\)
     b) Multiply
     c) IFFT\(_x\) – XY Transpose – IFFT\(_y\)
        – YZ Transpose – IFFT\(_z\)
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute

\[
\mathbf{ZI} \approx \text{FFT}[\Lambda \mathbf{I}]
\]

Memory: \(O\left( \frac{N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??} \right)\)

CPU time per iteration: \(O\left( \frac{N^C \log N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??} \right)\)

Comm time per iteration: \(O\left( \frac{P}{P_{\text{max}}^{xy}} \right)\)
Proposed Parallelization Scheme

• Two Workload Distribution Strategies
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
     a) $\text{FFT}_Z \rightarrow \text{ZY Transpose} \rightarrow \text{FFT}_Y$
        
        – $\text{YX Transpose} \rightarrow \text{FFT}_X$
     b) Multiply
     c) $\text{IFFT}_X \rightarrow \text{XY Transpose} \rightarrow \text{IFFT}_Y$
        
        – $\text{YZ Transpose} \rightarrow \text{IFFT}_Z$
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute

$ZI \approx FFT[AI]$

Memory: $O\left(\frac{N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??}\right)$

CPU time per iteration: $O\left(\frac{N^C \log N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??}\right)$

Comm time per iteration: $O\left(\frac{P}{P_{\text{max}}^{yz}} + \frac{P}{P_{\text{max}}^{xy}}\right)$
Proposed Parallelization Scheme

- **Two Workload Distribution Strategies**
  1. **Project/Anterpolate: 2-D Pencil**
  2. **Propagate: 2-D Pencil**
     a) $\text{FFT}_Z$ – ZY Transpose – $\text{FFT}_Y$
     b) Multiply
     c) $\text{IFFT}_X$ – XY Transpose – $\text{IFFT}_Y$
  3. **Interpolate: 2-D Pencil**
  3.5. **Redistribute**
  4. **Correct: 3-D Block**
  5. **Communicate Near/Redistribute**

\[ ZI \approx \text{FFT}[\Delta I] \]

Memory: $O\left( \frac{N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??} \right)$

CPU time per iteration: $O\left( \frac{N^C \log N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??} \right)$

Comm time per iteration: $O\left( \frac{P}{P_{\text{max}}^{yz}} + \frac{P}{P_{\text{max}}^{xy}} \right)$
Proposed Parallelization Scheme

- **Two Workload Distribution Strategies**
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
    a) FFT\(_Z\) – ZY Transpose – FFT\(_Y\)
    - YX Transpose – FFT\(_X\)
    b) Multiply
    c) IFFT\(_X\) – XY Transpose – IFFT\(_Y\)
    - YZ Transpose – IFFT\(_Z\)
  3. Interpolate: 2-D Pencil
  3.5. Distribute
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute

\[ \text{Memory: } O( \frac{N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??}) \]

\[ \text{CPU time per iteration: } O( \frac{N^C \log N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??}) \]

\[ \text{Comm time per iteration: } O( \frac{P}{P_{\text{max}}^{yz}} + \frac{P}{P_{\text{max}}^{xy}}) \]
Proposed Parallelization Scheme

• Two Workload Distribution Strategies
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
    a) FFT\(_Z\) – ZY Transpose – FFT\(_Y\)
    – YX Transpose – FFT\(_X\)
    b) Multiply
    c) IFFT\(_X\) – XY Transpose – IFFT\(_Y\)
    – YZ Transpose – IFFT\(_Z\)
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute

\[ ZI \approx \text{IFFT} \{ \text{FFT}[G] \cdot \text{FFT}[AI] \} \]

Memory: \( O(\frac{N^C}{P_{max}^{xy} P_{max}^{yz}} + \frac{pN}{??}) \)

CPU time per iteration: \( O(\frac{N^C \log N^C}{P_{max}^{xy} P_{max}^{yz}} + \frac{pN}{??}) \)

Comm time per iteration: \( O(\frac{P}{P_{max}^{yz}} + \frac{P}{P_{max}^{xy}}) \)
Proposed Parallelization Scheme

- **Two Workload Distribution Strategies**
  1. **Project/Interpolate**: 2-D Pencil
  2. **Propagate**: 2-D Pencil
     - a) $\text{FFT}_Z - ZY \text{ Transpose} - \text{FFT}_Y$
       - YX Transpose – $\text{FFT}_X$
     - b) Multiply
     - c) $\text{IFFT}_X - XY \text{ Transpose} - \text{IFFT}_Y$
       - YZ Transpose – $\text{IFFT}_Z$
  3. **Interpolate**: 2-D Pencil
  3.5. **Redistribute**
  4. **Correct**: 3-D Block
  5. **Communicate Near/Redistribute**

\[
\mathbf{ZI} \approx \text{IFFT}\{\text{FFT}[\mathbf{G}] \cdot \text{FFT}[\mathbf{AI}]\}
\]

Memory: \( O(\frac{N^C}{P_{\text{max}}^{xy}} \frac{N^C}{P_{\text{max}}^{yz}} + \frac{pN}{??}) \)

CPU time per iteration: \( O(\frac{N^C \log N^C}{P_{\text{max}}^{xy}} \frac{N^C \log N^C}{P_{\text{max}}^{yz}} + \frac{pN}{??}) \)

Comm time per iteration: \( O(\frac{P}{P_{\text{max}}^{yz}} + \frac{P}{P_{\text{max}}^{xy}}) \)
Proposed Parallelization Scheme

• Two Workload Distribution Strategies
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
     a) $\text{FFT}_Z – \text{ZY Transpose} – \text{FFT}_Y$
        – $\text{YX Transpose} – \text{FFT}_X$
     b) Multiply
     c) $\text{IFFT}_X – \text{XY Transpose} – \text{IFFT}_Y$
        – $\text{YZ Transpose} – \text{IFFT}_Z$
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute

$\mathbf{ZI} \approx \text{IFFT}\{\text{FFT}[\mathbf{G}] \cdot \text{FFT}[\mathbf{A}I]\}$

Memory: $O\left( \frac{N^C}{P^x_{\max} P^y_{\max}} + \frac{pN}{??} \right)$

CPU time per iteration: $O\left( \frac{N^C \log N^C}{P^x_{\max} P^y_{\max}} + \frac{pN}{??} \right)$

Comm time per iteration: $O\left( \frac{P}{P^y_{\max}} + \frac{P}{P^x_{\max}} \right)$
Proposed Parallelization Scheme

- Two Workload Distribution Strategies
  1. Project/Anterpolate: 2-D Pencil
  2. Propagate: 2-D Pencil
     a) $\text{FFT}_Z - \text{ZY Transpose} - \text{FFT}_Y$
     b) Multiply
     c) $\text{IFFT}_X - \text{XY Transpose} - \text{IFFT}_Y$
  3. Interpolate: 2-D Pencil
  3.5. Redistribute
  4. Correct: 3-D Block
  5. Communicate Near/Redistribute

\[
\begin{align*}
\text{ZI} & \approx \text{IFFT}\{\text{FFT}[^G] \cdot \text{FFT}[^A]\} \\
\end{align*}
\]

- Memory: $O(\frac{N^C}{P_{\text{max}}^{xy}} + \frac{p N}{??})$
- CPU time per iteration: $O(\frac{N^C \log N^C}{P_{\text{max}}^{xy}} + \frac{p N}{??})$
- Comm time per iteration: $O(\frac{P}{P_{\text{max}}^{yz}} + \frac{P}{P_{\text{max}}^{xy}})$
Proposed Parallelization Scheme

- **Two Workload Distribution Strategies**
  1. **Project/Anterpolate: 2-D Pencil**
  2. **Propagate: 2-D Pencil**
     - a) $\text{FFT}_Z$ – ZY Transpose – $\text{FFT}_Y$
       - YX Transpose – $\text{FFT}_X$
     - b) Multiply
     - c) $\text{IFFT}_X$ – XY Transpose – $\text{IFFT}_Y$
       - YZ Transpose – $\text{IFFT}_Z$
  3. **Interpolate: 2-D Pencil**
  3.5. **Redistribute**
  4. **Correct: 3-D Block**
  5. **Communicate Near/Redistribute**

$$ZI \approx \text{IFFT}\{\text{FFT}[G] \cdot \text{FFT}[AI]\}$$

**Memory:** $O\left(\frac{N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??}\right)$

**CPU time per iteration:** $O\left(\frac{N^C \log N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??}\right)$

**Comm time per iteration:** $O\left(\frac{P}{P_{\text{max}}^{yz}} + \frac{P}{P_{\text{max}}^{xy}}\right)$
Proposed Parallelization Scheme

- **Two Workload Distribution Strategies**
  1. **Project/Anterpolate: 2-D Pencil**
  2. **Propagate: 2-D Pencil**
     a) $\text{FFT}_Z - \text{ZY Transpose} - \text{FFT}_Y$
     b) Multiply
     c) $\text{IFFT}_X - \text{XY Transpose} - \text{IFFT}_Y$
  3. **Interpolate: 2-D Pencil**
  3.5. **Redistribute**
  4. **Correct: 3-D Block**
  5. **Communicate Near/Redistribute**

\[
\Lambda^T \text{IFFT}\{\text{FFT}[G] \cdot \text{FFT}[A I]\} \\
\]

**Memory:** $O(\frac{pN}{??} + \frac{N^C}{P_{\max}^{xy} P_{\max}^{yz}} + \frac{pN}{??})$

**CPU time per iteration:** $O(\frac{pN}{??} + \frac{N^C \log N^C}{P_{\max}^{xy} P_{\max}^{yz}} + \frac{pN}{??})$

**Comm time per iteration:** $O(\frac{P}{P_{\max}^{yz}} + \frac{P}{P_{\max}^{xy}})$
Proposed Parallelization Scheme

• Two Workload Distribution Strategies

1. Project/Anterpolate: 2-D Pencil
2. Propagate: 2-D Pencil
   a) $\text{FFT}_Z$ – ZY Transpose – $\text{FFT}_Y$
      – YX Transpose – $\text{FFT}_X$
   b) Multiply
   c) $\text{IFFT}_X$ – XY Transpose – $\text{IFFT}_Y$
      – YZ Transpose – $\text{IFFT}_Z$
3. Interpolate: 2-D Pencil
3.5. Redistribute
4. Correct: 3-D Block
5. Communicate Near/Redistribute

\[
\mathbf{ZI} \approx \Lambda^T \text{IFFT} \{ \text{FFT}[\mathbf{G}] \cdot \text{FFT}[\mathbf{A}] \}
\]

Memory: $O\left( \frac{pN}{??} + \frac{N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??} \right)$

CPU time per iteration: $O\left( \frac{pN}{??} + \frac{N^C \log N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??} \right)$

Comm time per iteration: $O\left( C^{rd} + \frac{P}{P_{\text{max}}^{yz}} + \frac{P}{P_{\text{max}}^{xy}} \right)$
Proposed Parallelization Scheme

• Two Workload Distribution Strategies

1. **Project/Anterpolate: 2-D Pencil**
2. **Propagate: 2-D Pencil**
   a) $\text{FFT}_Z - ZY$ Transpose – $\text{FFT}_Y$
   b) $\text{YX}$ Transpose – $\text{FFT}_X$
   b) Multiply
   c) $\text{IFFT}_X - XY$ Transpose – $\text{IFFT}_Y$
   d) $\text{YZ}$ Transpose – $\text{IFFT}_Z$
3. **Interpolate: 2-D Pencil**
3.5. **Redistribute**
4. **Correct: 3-D Block**
5. **Communicate Near/Redistribute**

\[ ZI \approx Z_{\text{near}} I + \Lambda^T \text{IFFT}\{\text{FFT}[G] \cdot \text{FFT}[A I]\} \]

Memory: \( O \left( \frac{N_{\text{near}}}{P^{k}} + \frac{pN}{??} + \frac{N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??} \right) \)

CPU time per iteration: \( O \left( \frac{N_{\text{near}}}{P^{k}} + \frac{pN}{??} + \frac{N^C \log N^C}{P_{\text{max}}^{xy} P_{\text{max}}^{yz}} + \frac{pN}{??} \right) \)

Comm time per iteration: \( O \left( C_{\text{rd}} + \frac{P}{P_{\text{max}}^{yz}} + \frac{P}{P_{\text{max}}^{xy}} \right) \)
Proposed Parallelization Scheme

- **Two Workload Distribution Strategies**
  1. **Project/Anterpolate**: 2-D Pencil
  2. **Propagate**: 2-D Pencil
     a) $\text{FFT}_Z - \text{ZY Transpose} - \text{FFT}_Y$
     - $\text{YX Transpose} - \text{FFT}_X$
     b) Multiply
     c) $\text{IFFT}_X - \text{XY Transpose} - \text{IFFT}_Y$
     - $\text{YZ Transpose} - \text{IFFT}_Z$
  3. **Interpolate**: 2-D Pencil
  3.5. **Redistribute**
  4. **Correct**: 3-D Block
  5. **Communicate Near/Redistribute**

\[
\mathbf{ZI} \approx \mathbf{Z}_{\text{near}} \mathbf{I} + \Lambda^T \text{IFFT}\{\text{FFT}[^\mathbf{G}] \cdot \text{FFT}[^\mathbf{A}I]\}
\]

**Memory:**
\[
O\left(\frac{N_{\text{near}}}{P} + \frac{pN}{??} + \frac{N^C}{\max P_{xy} P_{yz}} + \frac{pN}{??}\right)
\]

**CPU time per iteration:**
\[
O\left(\frac{N_{\text{near}}}{P} + \frac{pN}{??} + \frac{N^C \log N^C}{\max P_{xy} P_{yz}} + \frac{pN}{??}\right)
\]

**Comm time per iteration:**
\[
O(C_{\text{near}} + C_{\text{rd}} + \frac{P}{\max P_{xy} P_{yz}} + \frac{P}{\max P_{xy}})
\]
Parallelization

- General Limits
- Traditional Scheme and Its Limitations
- Proposed Scheme
- Comparison
Traditional vs. Proposed Scheme

1. Correction computations better balanced

\[ P_{2D}^{\text{max}} = \min(N_{cx}, N_{cy}) \min(N_{cy}, N_{cz}) \]

2. Grid limitation extended

[Graphs showing wall clock time and memory/core comparison for different schemes and grid limitations.]
Traditional vs. Proposed
Maximum $P$ and Maximum $N$

Matrix Fill Time

Matrix Solve Time

Memory

radius=104 mm

$\varepsilon_r = 41.5, \sigma = 0.97$
$P$ vs. $N$ (Relative Efficiency=90%)

Matrix Fill Time (eff=90%)

Matrix Solve Time (eff=90%)

Memory (eff=90%)

radius=104 mm

$\epsilon_r = 41.5, \sigma = 0.97$
$P$ vs. $N$ (Relative Efficiency=50%)

Matrix Fill Time (eff=50%)

Matrix Solve Time (eff=50%)

Memory (eff=50%)

$N^{0.43}$

$N^{0.66}$

radius=104 mm

$\varepsilon_r = 41.5, \sigma = 0.97$

Ranger
General Limits of Parallelization

- **(Hard) Scalability Limitations**
  1. Sequential part of the code (Amdahl’s Law)
     - No size $N$ arrays or operations
  2. Load imbalance/Can’t divide computations further (run out of parallelism)
     - Two workload distribution strategies
     - 2D pencil decomposition based 3D FFTs \[^1\]
  3. Communication latency/bandwidth
     - Hybrid MPI/OpenMP \[^2\]

BIOEM Problems

- High-Resolution Simulations
- AustinMan Antropohomorphic Model
High-Resolution Simulations

1030 mm$^3$

683 mm$^3$

683 mm$^3$

$P_{\text{absorbed}}$

15 mm$^3$

11 mm$^3$

11 mm$^3$

0.3 mm$^3$

0.2 mm$^3$

0.2 mm$^3$
Sample Tissues: 11 mm³ vs. 0.2 mm³

Skin Dry + Fat

Muscle + Tendon

Bone Cortical + Bone Marrow
Sample Tissues: 11 mm$^3$ vs. 0.2 mm$^3$

1. White+Gray Matter
2. Teeth
3. Tongue
4. Gland
5. Spinal Chord
6. Mucous Membrane
7. Vitreous Humor
8. Blood Vessel
Sample Tissues: 11 mm³ vs. 0.2 mm³

1. White+Gray Matter
2. Teeth
3. Tongue
4. Gland
5. Spinal Chord
6. Mucous Membrane
7. Vitreous Humor
8. Blood Vessel
AustinMan Anthropomorphic Model

- **Semi-automated**
  - Automated: Color based identification
  - Manual: Knowledge of anatomy

- **Boundary&Region Identification Software Kit (BRISkit)**
  - Create masks
  - Region growing
  - MATLAB graphical user interface
    - Identify regions
    - Edit boundaries
AustinMan Anthropomorphic Model

Sub-mm pixel resolution (1/3 mm)
AustinMan Anthropomorphic Model

Download at: web2.corral.tacc.utexas.edu/AustinManEMVoxels/

AustinMan Electromagnetic Voxels

**AustinMan Electromagnetic Voxels V1.1**

For AustinMan v1.1, we updated the AustinMan v1.0 model by improving the continuity of some materials across slices, correcting the known errors in v1.0, and using the 3-D coarsening method. AustinMan v1.1 has been available for download since March 21, 2011. A list of materials identified in each slice can be found here. A log of changes from AustinMan v1.0 to AustinMan v1.1 for each slice can be found here.

<table>
<thead>
<tr>
<th>Model Type</th>
<th>Voxel Resolution</th>
<th>Number of Voxels</th>
<th>File Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partial Body (Slices 1002-1354)</td>
<td>$1 \times 1 \times 1 \text{ mm}^3$</td>
<td>13,315,164</td>
<td>1.79 GB</td>
</tr>
<tr>
<td>Partial Body (Slices 1002-1354)</td>
<td>$2 \times 2 \times 2 \text{ mm}^3$</td>
<td>1,675,948</td>
<td>218 MB</td>
</tr>
<tr>
<td>Partial Body (Slices 1002-1354)</td>
<td>$4 \times 4 \times 4 \text{ mm}^3$</td>
<td>212,511</td>
<td>26 MB</td>
</tr>
</tbody>
</table>

AustinMan Electromagnetic Voxels by J. W. Massey, C. S. Geyik, M. Ball, P. Findell, and A. E. Yilmaz is licensed under a Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License.

Based on a work at http://web2.corral.tacc.utexas.edu/AustinManEMVoxels/. Permissions beyond the scope of this license may be available at http://web2.corral.tacc.utexas.edu/AustinManEMVoxels/index.htm.
Acknowledgements

Fangzhou Wei
Madison Ball
Jackson Massey
Cemil Geyik
Dr. Tahir Malas
Dr. Victor Eijkhout
Prof. Paul Findell
Prof. John Pearce
Prof. Leszek Demkowicz
Prof. Robert van de Geijn

Normalized E-Field

15 mm³
11 mm³
11 mm³
0.3 mm³
0.2 mm³
0.2 mm³