LARGE-SCALE 3D CSEM MODELING
WITH A BLR MUMPS SOLVER

Sebastien de la Kethulle de Ryhove, Piyoosh Jaysaval & Daniil V. Shantsev
EMGS R&D, Norway

MUMPS Users Days, Grenoble, France
June 1-2, 2017
Our Story

• 2013 MUMPS Users Days (guests)
• 2014-2015 MUMPS – EMGS research project
• 2017 Geophys. J. Intl. article
• 2017 MUMPS Users Days again
• 2014 - … Use of MUMPS in EMGS for research / production
Outline

• Marine Controlled-source EM (CSEM) method
• CSEM forward & inverse problems
• MUMPS-BLR for CSEM problems
• Air effects
• Conclusions
Marine CSEM survey

Horizontal electric dipole source:
- current up to 7200 Amp
- frequency up to ~20 Hz
- tow-distance ~1,000 km i.e. ~10,000 source positions

Seabed EM receiver:
- measures $E_x$, $E_y$, $H_x$, $H_y$
- 2D grid, typically ~100 receivers

3D subsurface resistivity distribution:
- inverted from the measured data

High-resistive anomaly indicating a possible oil/gas accumulation: impact on drilling decision
EMGS

- 2 – 4 vessels
- ~ 200 employees
- ~1000 surveys
- ~ 100,000 km² data library (10-20% area of France)
- R&D group: ~10 people
CSEM Forward & Inverse problems

Spot the difference.
Forward problem

- EMGS forward modelling
  - Time-domain
    - Frequency-domain
      - Direct Solver (MUMPS)
      - Iterative Solver
  - Equation to solve: \( \nabla \times \nabla \times \mathbf{E} - i\omega \mu_0 \sigma \mathbf{E} = i\omega \mu_0 \mathbf{J}_{\text{source}} \)

- This study:
  - VTI
  - Finite-difference based on Yee grid
  - Unknowns: \( E_x, E_y, E_z \)
  - 13 non-zero elements in each row of \( A \)
  - Symmetric \( A \)

Can be the preferred choice for very large number of RHSs
CSEM data

Number of receivers: \( N_R \sim 100 \)
Number of source positions: \( N_S \sim 10,000 \)

Data from one receiver at one frequency

- \( \sim 200 \) datapoints per line (for 100 m sampling, 20 km line)
- \( \sim 5 \) source lines for each receiver
- Amplitude + phase
- \( 3 - 5 \) frequencies
- \( 2 - 4 \) field components (\( E_x, E_y \))
- \( \sim 100 \) receivers

- Total: a few millions of data samples
Inversion algorithm

Survey data $d$

Initial model $m^0$

Regularization

Modeling

Reg. cost $\epsilon_{\text{reg}}(m)$

Data cost $\epsilon_{\text{dat}}(d, A(m))$

Model update $m^{i+1}$

OK

No

Yes

Stop

Inversion example

Survey data $d$

Unknown subsurface

Electric mag. [V/(Am²)]

Phase [deg]

Offset [km]

Position [km]
Inversion algorithm

Survey data $d$

Initial model $m^0$

Regularization

Data cost $\varepsilon_{dat}(d, A(m))$

Model update $m^{i+1}$

Inversion algorithm

Reg. cost $\varepsilon_{reg}(m)$

Modeling

Inversion example

Resistivity [Ωm]

Depth [km]

Position [km]

Electric mag. [V/(Am²)]

Phase [deg]
Inversion algorithm

\[ \text{Inversion algorithm} \]

\[ \sim N_r \] RHSs (reciprocity 😊)

\[ \sim N_r \] RHSs for quasi-Newton update

\[ \sim N_s + N_r \] RHSs for Gauss-Newton update

\[ \text{Survey data } \mathbf{d} \]

\[ \text{Initial model } \mathbf{m}^0 \]

\[ \text{Regularization} \]

\[ \text{Modeling} \]

\[ \text{Reg. cost } \varepsilon_{\text{reg}}(\mathbf{m}) \]

\[ \text{Data cost } \varepsilon_{\text{dat}}(\mathbf{d}, A(\mathbf{m})) \]

\[ \text{Model update } \mathbf{m}^{i+1} \]

\[ \text{OK} \]

\[ \text{Yes} \]

\[ \text{Stop} \]

\[ \text{Inversion example} \]

\[ \text{Resistivity [Ωm]} \]

\[ \text{Position [km]} \]

\[ \text{Depth [km]} \]

\[ \varepsilon_{\text{reg}}(\mathbf{m}) \]

\[ \text{Offset [km]} \]

\[ \text{Electric mag. [V/(Am)]} \]

\[ \text{Phase [deg]} \]

\[ \text{RHSs} \]

\[ \sim N_r \]

\[ \sim N_s + N_r \]
Quasi-Newton and Gauss-Newton inversion

- Cost function to be minimized:
  \[ \varepsilon(m) = \varepsilon_{\text{data}}(m) + \lambda \varepsilon_{\text{reg}}(m) = \sum_{k=1}^{N} r_k r_k^* + \lambda \varepsilon_{\text{reg}}(m) \]
- Model update \( \Delta m \) is found from:
  \[ (H_{\text{data}} + \lambda H_{\text{reg}}) \Delta m = -g \]
- The Hessian matrix is \( H_{\text{data}} \approx J^+ J + \text{c. c.} \)

Gradient vector
\[ g = \begin{pmatrix} \frac{\partial \varepsilon}{\partial \sigma_1} \\ \vdots \\ \frac{\partial \varepsilon}{\partial \sigma_M} \end{pmatrix} \]

Jacobian (sensitivity) matrix
\[ J = \begin{pmatrix} \frac{\partial r_1}{\partial \sigma_1} & \cdots & \frac{\partial r_1}{\partial \sigma_M} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_N}{\partial \sigma_1} & \cdots & \frac{\partial r_N}{\partial \sigma_M} \end{pmatrix} \]

Number of model parameters

<table>
<thead>
<tr>
<th>Method</th>
<th>Hessian ( H )</th>
<th>Number of RHSs in forward problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quasi-Newton (e.g. BFGS)</td>
<td>approximated using successive gradients ( g )</td>
<td>( \sim N_r ), ( \sim 100 )</td>
</tr>
<tr>
<td>Gauss-Newton</td>
<td>computed from Jacobian ( H_{\text{data}} \approx J^+ J + \text{c. c.} )</td>
<td>( \sim N_s + N_r ), ( \sim 10,000 )</td>
</tr>
</tbody>
</table>
Gauss-Newton is better

- Gauss - Newton is more expensive, but a much powerful method
- It will take over in the future:
  - 2008: Launch BFGS
  - 2016: Launch Gauss-Newton

SEG 2016:
MUMPS for CSEM
MUMPS for CSEM: previous studies

<table>
<thead>
<tr>
<th></th>
<th>Number of unknowns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Streich</td>
<td>Geophysics 2009</td>
</tr>
<tr>
<td>da Silva et al.</td>
<td>Geophysics 2012</td>
</tr>
</tbody>
</table>

0.9 millions
4.2 millions
7.8 millions

Goals of the present study:
- Use **BLR** for factorization of CSEM matrices
- Test problems with **>20 millions** unknowns
- Compare MUMPS vs Iterative solver
## Models and Matrices

### Half-space + Target H-model

<table>
<thead>
<tr>
<th>Grid</th>
<th>Matrix</th>
<th>dx = dy</th>
<th>dz</th>
<th>Nx = Ny</th>
<th>Nz</th>
<th>Number of unknowns</th>
<th>Number of non-zero elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>G1</td>
<td>H1</td>
<td>400</td>
<td>200</td>
<td>64</td>
<td>74</td>
<td>909,312</td>
<td>11,658,644</td>
</tr>
<tr>
<td>G2</td>
<td>H3 / D3</td>
<td>200</td>
<td>200</td>
<td>114</td>
<td>74</td>
<td>2,885,112</td>
<td>37,148,644</td>
</tr>
<tr>
<td>G3</td>
<td>H17</td>
<td>100</td>
<td>100</td>
<td>214</td>
<td>127</td>
<td>17,448,276</td>
<td>225,626,874</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Grid</th>
<th>Matrix</th>
<th>dx = dy</th>
<th>dz</th>
<th>Nx</th>
<th>Ny</th>
<th>Nz</th>
<th>Number of unknowns</th>
<th>Number of non-zero elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>G4</td>
<td>S3</td>
<td>480</td>
<td>80</td>
<td>98</td>
<td>87</td>
<td>130</td>
<td>3,325,140</td>
<td>42,836,538</td>
</tr>
<tr>
<td>G5</td>
<td>S21</td>
<td>240</td>
<td>40</td>
<td>181</td>
<td>160</td>
<td>237</td>
<td>20,590,560</td>
<td>266,361,112</td>
</tr>
</tbody>
</table>

### SEAM S-model

- **SEAM model:**
  - Created by SEG Advanced Modelling
  - Salt body
  - Representative for Gulf of Mexico
Block-low-rank (BLR) algorithm

Input: a symmetric matrix $A$ of $p \times p$ blocks
Output: the factors matrices $L, D$

for $k = 1$ to $p$ do
  Factor: $L_{kk}D_{kk}L_{kk}^T = A_{kk}$
  for $i = k + 1$ to $p$ do
    Solve: $L_{ik} = A_{ik}L_{kk}^T D_{kk}^{-1}$
    Compress: $L_{ik} \approx Y_{ik}Z_{ik}^T$
    for $j = k + 1$ to $i$ do
      Update: $A_{ij} = A_{ij} - L_{ik}D_{kk}L_{jk}^T$
      $\approx A_{ij} - Y_{ik}(Z_{ik}^T D_{kk} Z_{jk}) Y_{jk}^T$
    end for
  end for
end for

• BLR format is used to compress fronts
• The compression accuracy is controlled by the BLR threshold $\varepsilon$ that varied from $10^{-4}$ to $10^{-16}$
• Block size: 256 (or 416 for largest matrix, S21)

BLR-compressed matrix structure. Block darkness = Compression rate
Figures from Amestoy et al. 2015
BLR threshold & Solution accuracy

Relative residual L2 norm: \[ \delta = \frac{||s - Mx^\varepsilon||}{||s||} < 10^{-6} \]

3D cubes of relative difference between BLR and FR solutions

Optimal BLR threshold: \( \varepsilon = 10^{-7} \)
**BLR savings**

**BLR factor storage (% of FR)**

- $10^{-11}$ to $10^{-05}$

**BLR factorization flops (% of FR)**

- $10^{-11}$ to $10^{-05}$

**Weak dependence:**
- *Choice of the BLR threshold is not critical*
- *Large gains even for strict accuracy requirements*
BLR savings

Hardware:
- EOS supercomputer
- 90 MPI tasks × 10 threads

Hardware details:
CALMIP supercomputer EOS – a BULLx DLC system, 612 nodes, each composed of two Intel Ivybridge processors with 10 cores (total 12 240 cores) running at 2.8 GHz per node and 64 GB/node, [https://www.calmip.univ-toulouse.fr/](https://www.calmip.univ-toulouse.fr/)

NB:
Memory reduction due to storage savings has not yet been implemented for these tests, hence, the potential gains in run-time are even larger.
Scalability

**Scalability Graph**

**BLR factorization time (% of FR)**

- **LR with $\epsilon = 10^{-7}$**
  - H17
  - S21

**Number of cores**

- $90 \times 10$
- $128 \times 10$
- $192 \times 10$

Robust BLR-gains independent of the number of cores
Air effects

Spot the difference.
Air-wave

In marine CSEM variations of properties (resistivity is extreme)

- Air has almost infinite resistivity
- EM waves propagate there fast (speed of light) and without attenuation
- Air effectively connects distant parts of the model

Figure from Amestoy et al. 2015

BLR-compressed matrix structure.
Shallow-water & Deep-water models

The number of cells is the same for both models, i.e. the matrices have identical structure.
BLR & Full Rank (FR) flops

<table>
<thead>
<tr>
<th>Layer</th>
<th>FR - Shallow</th>
<th>FR - Deep</th>
<th>BLR - Shallow</th>
<th>BLR - Deep</th>
</tr>
</thead>
<tbody>
<tr>
<td>Air 10^6 Ωm</td>
<td>162 \cdot 10^{12}</td>
<td>162 \cdot 10^{12}</td>
<td>24 \cdot 10^{12}</td>
<td>14 \cdot 10^{12}</td>
</tr>
<tr>
<td>Water 25 Ωm</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Formation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 - 100 Ωm</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Matrix size: 4.9 \cdot 10^6
Flops complexity

Number of flops for matrix factorization \( N_{\text{flops}} \propto N^m \)

Matrix size (number of unknowns) \( N \)

Consistent with (or better than):

- Theory: \( m = 1.7 \)

- 3D Seismic: \( m = 1.78 \)
Factor storage complexity

Memory needed to store factors

Matrix size (number of unknowns)

\[ N_{\text{bytes}} \propto N^m \]

Consistent with:

- **Theory**:
  - \( m = 1.33 \) (FR), \( m = 1.17 \) (BLR)

- **3D Seismic**:
  - \( m = 1.36 \) (FR), \( m = 1.19 \) (BLR)
Number of RHS estimates

Example CSEM survey over the SEAM model
• \( N_r = 11 \times 11 = 121 \) receiver
• 22 towlines
• each towline has 150 shot points (30 km / 200 m)
• Source positions in total: \( N_s = 22 \times 150 = 3300 \)
• Field components: \( N_{fields} = 4 \) (Ex, Ey, Hx, Hy)

BFGS inversion:

\[
N_{RHS} = N_r \times N_{fields} \times 2 = 986
\]

Gauss-Newton inversion:

\[
N_{RHS} = N_s + N_r \times N_{fields} = 3784
\]
MUMPS-BLR vs Iterative solver

<table>
<thead>
<tr>
<th>Inversion</th>
<th>Number of RHS</th>
<th>FR solver times (sec)</th>
<th>BLR solver times (sec)</th>
<th>Iterative solver</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Analysis</td>
<td>Factoriz</td>
<td>Solution</td>
</tr>
<tr>
<td>BFGS</td>
<td>968</td>
<td>87</td>
<td>2803</td>
<td>965</td>
</tr>
<tr>
<td>Gauss-Newton</td>
<td>3784</td>
<td>87</td>
<td>2803</td>
<td>3772</td>
</tr>
</tbody>
</table>

- Iterative solver always wins for BFGS inversion (<1000 RHSs)
- Iterative solver always wins for full-rank MUMPS
- Gauss-Newton: due to BLR factorization became faster than iterative solver! 😊
- MUMPS solution time (1 sec / RHS) is currently slower than iterative solver 😞
- BLR can also be applied to the solution phase, and MUMPS may win at the end 😊

S21 matrix with $21 \times 10^6$ unknowns
SPOT THE DIFFERENCE.

Thank you