

3D CSEM modelling in the frequency and Laplace domains

One story of success and one of failure

Dieter Werthmüller

5th December 2019

Thursday Talk – Applied Geophysics & Petrophysics



Faculty of Civil Engineering and Geosciences > Department of Geoscience & Engineering > Section of Applied Geophysics & Petrophysics

The Project

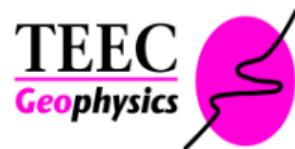


GitaroJIM.eu



MarTERA.eu

Project duration: 3 years (June 2018 – May 2021)

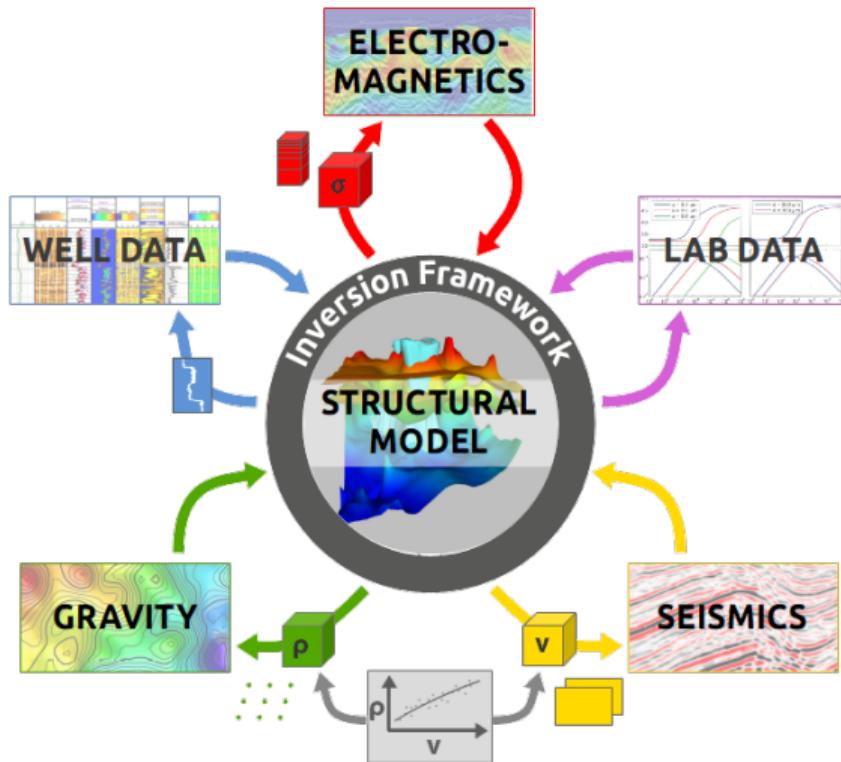


TERRASYS
GEOPHYSICS

Supported by University of Southampton and Equinor

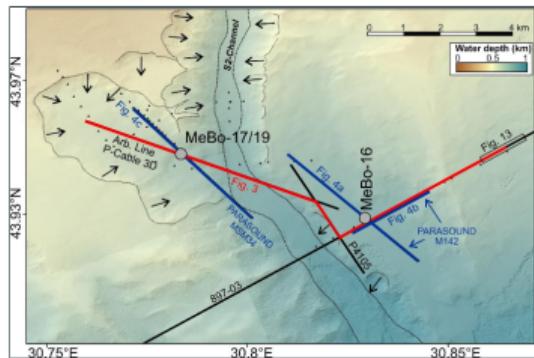
Member of the Delphi Consortium

Gitaro.JIM – Objectives

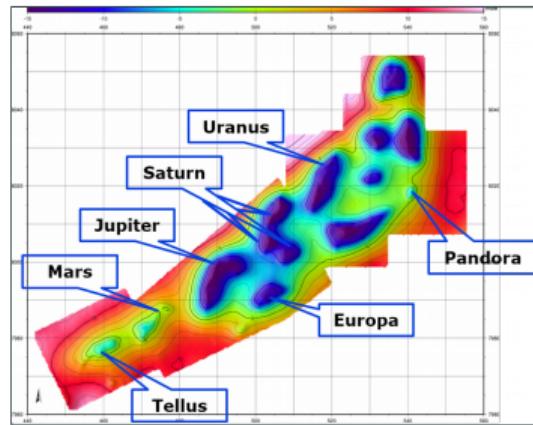


Gitaro.JIM – Data

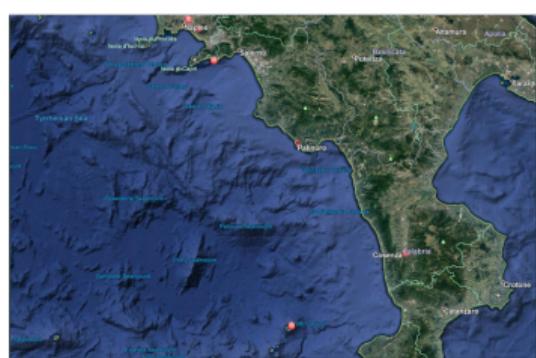
Methane Hydrates
Danube Paleo Delta
Black Sea



Oil & Gas
Nordkapp Basin
Barents Sea



Seafloor Massive Sulfides
Palinuro
Tyrrhenian Sea



- 3D CSEM modelling; frequency domain
- Comparison to existing code; frequency domain and time domain
- Inversion
- Anisotropy
- Topography

The Code

W.A. Mulder, T.B. Jönsthövel, M. Wirianto, E.C. Slob, R.-E. Plessix, . . .

$$-s\sigma\hat{\mathbf{E}} - \nabla \times \mu^{-1} \nabla \times \hat{\mathbf{E}} = s\hat{\mathbf{J}}$$

- Matlab/C
- Multigrid
- Finite Volumes
- VTI resistivity

Geophysical Prospecting, 2006, 54, 633–649

A multigrid solver for 3D electromagnetic diffusion

W.A. Mulder*

Shell International Exploration and Production, PO Box 60, 2280 AB Rijswijk, The Netherlands

Received March 2005, revision accepted March 2006

ABSTRACT

The performance of a multigrid solver for the time-harmonic in geophysical settings is investigated. The frequencies are so travelling at the speed of light to be negligible, so that a direct discretization of the governing equations is obtained



- Python/Numba
- Apache v2.0
- empymod.github.io

Installation

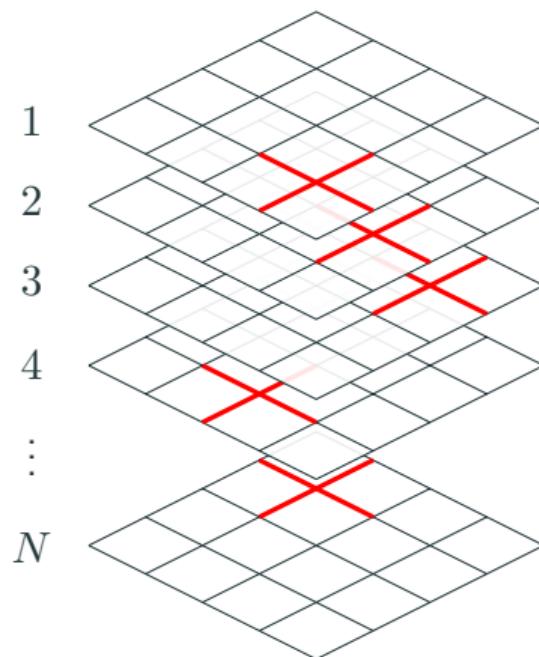
conda: `conda install -c conda-forge emg3d`

pip: `pip install emg3d`

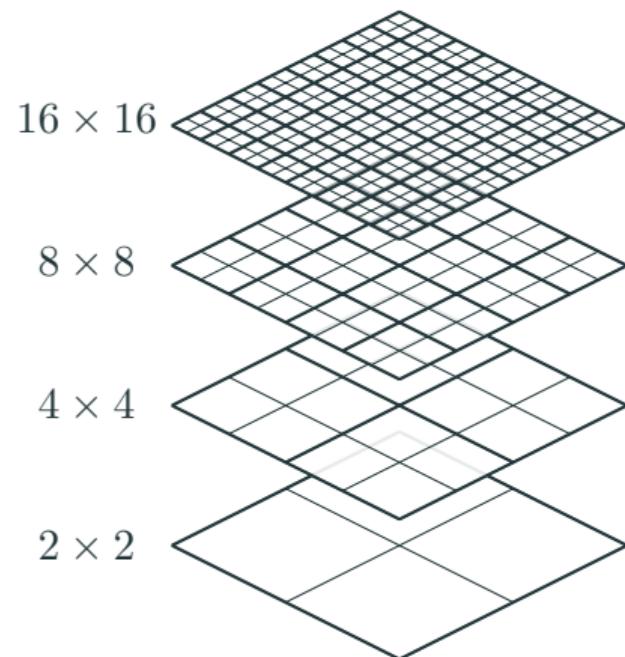
Requirements: NumPy, SciPy, Numba, empymod; Python 3.7+

Multigrid method

Matrix-free solver

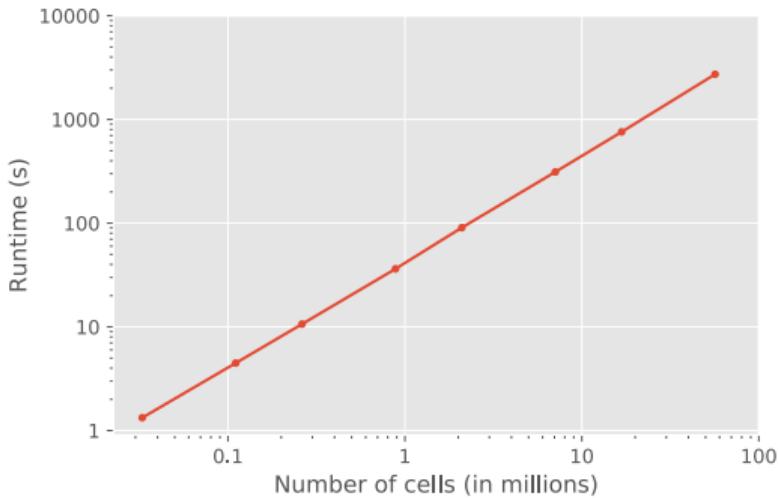


Restriction and prolongation

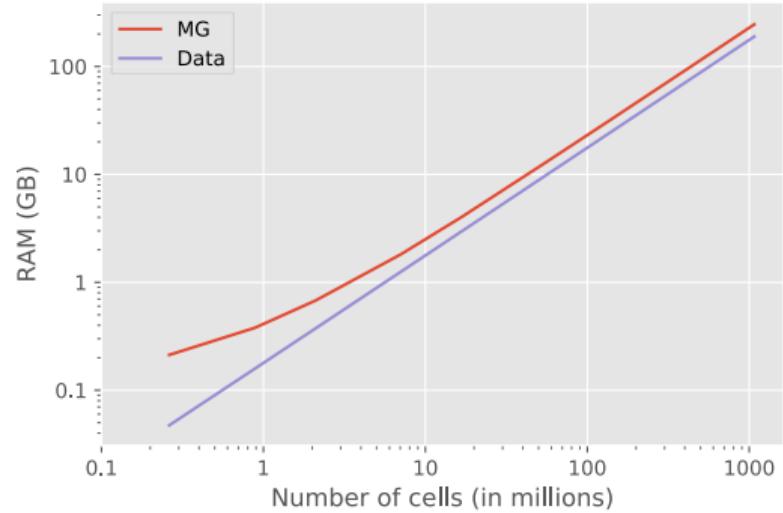


Multigrid – CPU & RAM

CPU



RAM



Measured on texel, single thread

Frequency Domain

emg3d – Minimal Example

```
import emg3d
import discretize

grid = discretize.TensorMesh(
    [[(25, 48)], [(50, 32)], [(30, 32)]],
    x0='CCC')

model = emg3d.utils.Model(
    grid, res_x=1.5, res_y=1.8, res_z=3.3)

sfield = emg3d.utils.get_source_field(
    grid, src=[0, 0, 0, 0, 0], freq=10)

efield = emg3d.solver.solver(
    grid, model, sfield, verb=3)
```

emg3d – Minimal Example

```
import emg3d
import discretize

grid = discretize.TensorMesh(
    [(25, 48)], [(50, 32)], [(30, 32)],
    x0='CCC')

model = emg3d.utils.Model(
    grid, res_x=1.5, res_y=1.8, res_z=3.3)

sfield = emg3d.utils.get_source_field(
    grid, src=[0, 0, 0, 0, 0], freq=10)

efield = emg3d.solver.solver(
    grid, model, sfield, verb=3)
```

```
:: emg3d START :: 10:28:27 :: v0.9.1
```

```
MG-cycle      : 'F'          ssolsolver : False
semicoarsening : False [0]    tol       : 1e-06
linerelaxation : False [0]    maxit     : 50
nu_{i,1,c,2}   : 0, 2, 1, 2  verb       : 3
Original grid  : 48 x 32 x 32 => 49,152 cells
Coarsest grid  : 3 x 2 x 2   => 12 cells
Coarsest level : 4 ; 4 ; 4
```

[hh:mm:ss]	rel. error	[abs. error, last/prev]	l s
------------	------------	-------------------------	-----



[10:28:27]	2.623e-02	after	1 F-cycles	[1.464e-06, 0.026]	0 0
[10:28:28]	2.253e-03	after	2 F-cycles	[1.258e-07, 0.086]	0 0
[10:28:28]	3.051e-04	after	3 F-cycles	[1.704e-08, 0.135]	0 0
[10:28:29]	5.501e-05	after	4 F-cycles	[3.071e-09, 0.180]	0 0
[10:28:29]	1.170e-05	after	5 F-cycles	[6.532e-10, 0.213]	0 0
[10:28:30]	2.745e-06	after	6 F-cycles	[1.533e-10, 0.235]	0 0
[10:28:30]	6.874e-07	after	7 F-cycles	[3.838e-11, 0.250]	0 0

```
> CONVERGED
> MG cycles      : 7
> Final rel. error : 6.874e-07
```

```
:: emg3d END   :: 10:28:30 :: runtime = 0:00:03
```

emg3d – Minimal Example

```
import emg3d
import discretize

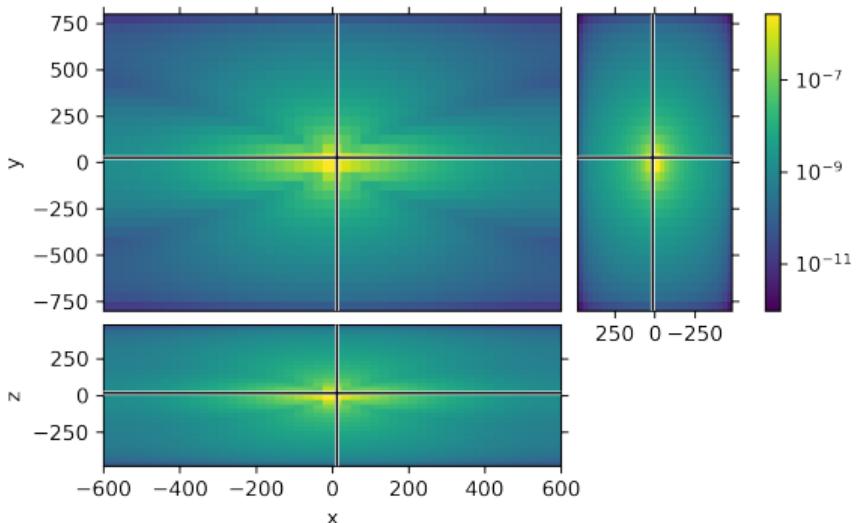
grid = discretize.TensorMesh(
    [[(25, 48)], [(50, 32)], [(30, 32)]],
    x0='CCC')

model = emg3d.utils.Model(
    grid, res_x=1.5, res_y=1.8, res_z=3.3)

sfield = emg3d.utils.get_source_field(
    grid, src=[0, 0, 0, 0, 0], freq=10)

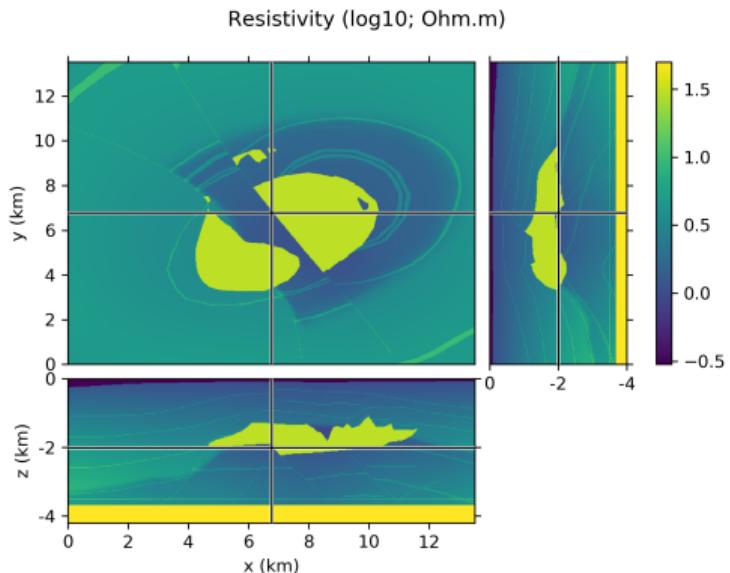
efield = emg3d.solver.solver(
    grid, model, sfield, verb=3)

grid.plot_3d_slicer(
    efield.fx.ravel('F'), view='abs', vType='Ex',
    pcolorOpts={'norm': LogNorm()})
```

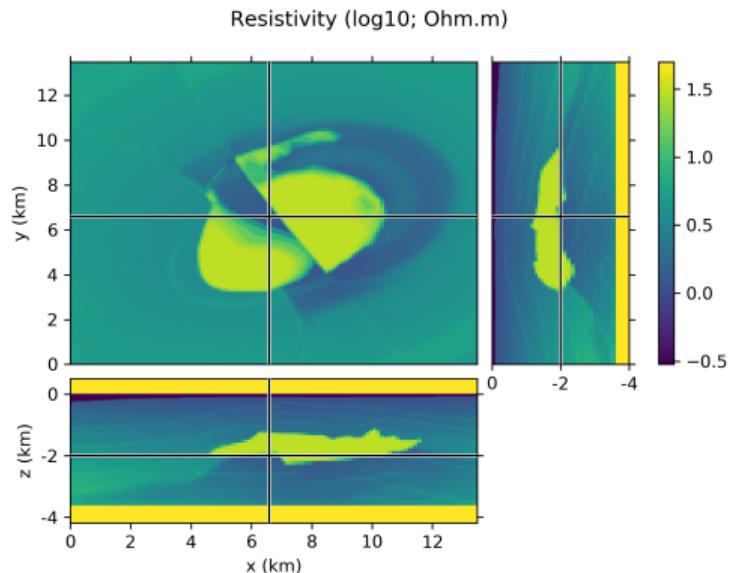


emg3d – SEG-EAGE salt model[†]

≈ 96 million cells (only model)
 $(20\text{m} \times 20\text{m} \times 20\text{m})$

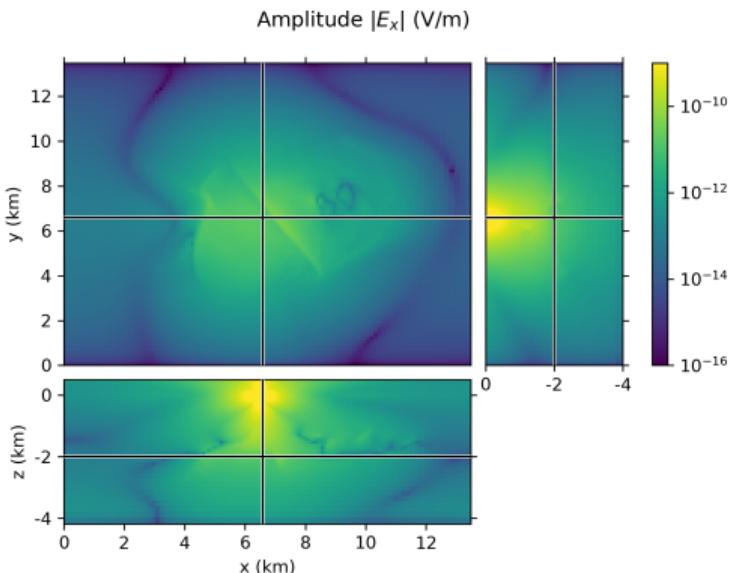


≈ 2.1 million cells (+ air & boundary)
 $(x/y: 55 - 182\text{ m}; z: 20 - 517\text{ m})$



[†] SEG-EAGE salt model: Aminzadeh et al. (1997); Velocity-to-resistivity conversion: Mulder (2007).

emg3d – SEG-EAGE salt model[†]



:: emg3d START :: 10:36:50 :: v0.9.1

```
MG-cycle      : 'F'          sslsolver : 'bicgstab'
semicoarsening : False [0]    tol      : 1e-06
linerelaxation : False [0]    maxit    : 50 (1)
nu_{i,i,c,2}   : 0, 2, 1, 2  verb      : 3
Original grid  : 128 x 128 x 128 => 2,097,152 cells
Coarsest grid  : 2 x 2 x 2    => 8 cells
Coarsest level : 6 ; 6 ; 6
```

[hh:mm:ss]	rel. error	solver	MG	l s
[10:37:03]	1.591e-01	after		1 F-cycles 0 0
[10:37:15]	3.299e-02	after		2 F-cycles 0 0
[10:37:16]	2.291e-02	after	1 bicgstab-cycles	3 F-cycles 0 0
[10:37:26]	6.756e-03	after		4 F-cycles 0 0
[10:37:37]	1.035e-03	after		5 F-cycles 0 0
[10:37:38]	8.113e-04	after	2 bicgstab-cycles	6 F-cycles 0 0
[10:37:51]	3.055e-04	after		7 F-cycles 0 0
[10:38:01]	6.347e-05	after		8 F-cycles 0 0
[10:38:02]	4.581e-05	after	3 bicgstab-cycles	9 F-cycles 0 0
[10:38:14]	1.753e-05	after		10 F-cycles 0 0
[10:38:27]	4.368e-06	after		11 F-cycles 0 0
[10:38:28]	3.969e-06	after	4 bicgstab-cycles	
[10:38:43]	2.619e-06	after		
[10:38:55]	1.626e-06	after		
[10:38:56]	1.127e-06	after	5 bicgstab-cycles	
[10:39:07]	8.366e-07	after		
[10:39:08]	7.448e-07	after	6 bicgstab-cycles	

```
> CONVERGED
> Solver steps     : 6
> MG prec. steps   : 11
> Final rel. error : 7.448e-07
```

:: emg3d END :: 10:39:08 :: runtime = 0:02:18

Time Domain

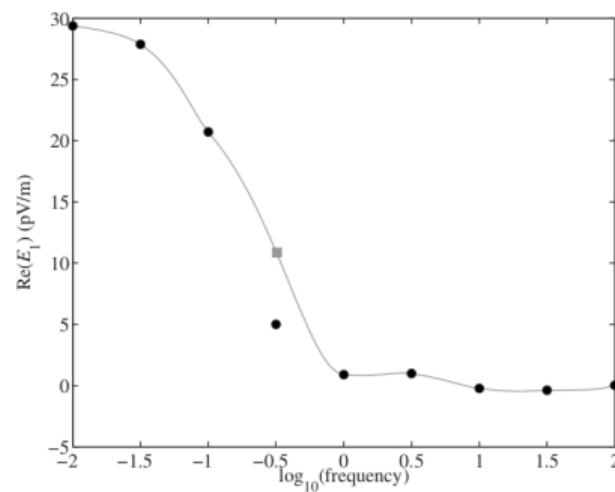
Automatic gridding

$$\delta = \sqrt{\frac{2}{\omega \mu \sigma}} \approx 503.3 / \sqrt{f \sigma},$$

- Minimum cell width $\text{func}(\delta_s, L_s, \text{pps})$
- Domains: $\text{pps} \cdot \delta_{\text{ave}}$
 - Core domain: $\text{pps} \approx 4 - 8$ ($3 - 8$)
 - Boundary domain: $\text{pps} \approx 4$ (5)
 - Air (upwards z): 50 km
- Mapping by volume averaging
- **Find optimal α for a fixed #cells**

Adaptive frequency selection

- Start: $f_{\min} - f_{\max}$, $1/\text{dec}$
- Add more frequencies left and right of f_i if $|E_{\text{int}}(f_i) - E_{\text{calc}}(f_i)| > \text{tol}$



Changes to existing schemes

(a) code

- Flexible #cell, in each direction
- Mult. with any low prime, $\{2, 3, 5\} \cdot 2^n$
e.g., **64, 80, 96, 128, 160, 192, 256**

(c) f -selection and Fourier transform

- Regular spacing (log-scale)
- A logarithmic Fourier transform
 - Digital Linear Filters (DLF)[†]
 - FFTLog[‡]

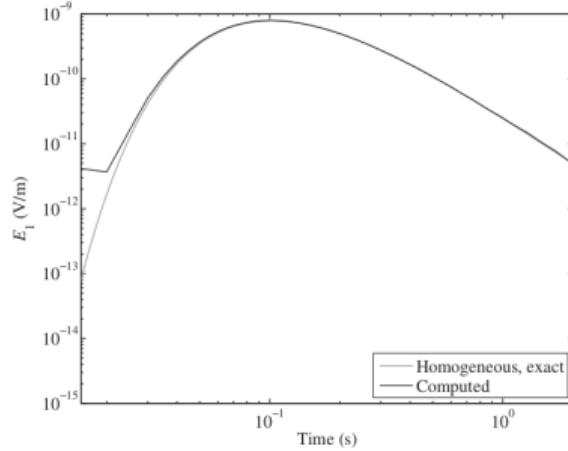
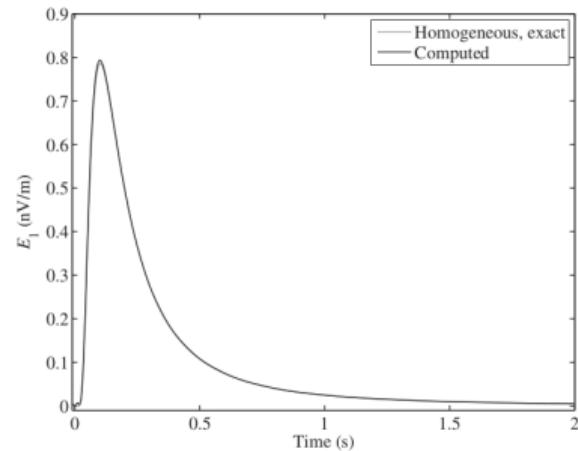
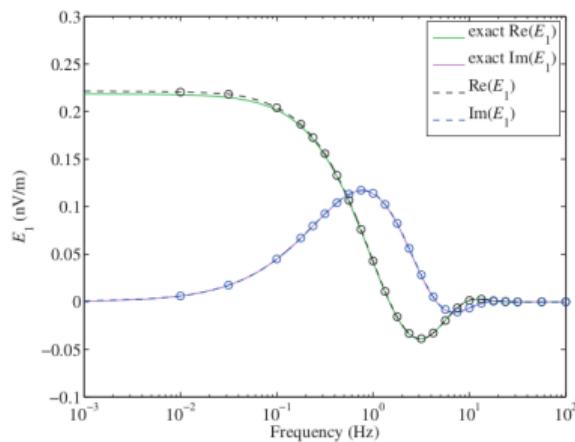
(b) automatic gridding

- Define core domain
- **Search for minimal number of cells**
 - α -range for core domain
 - α -range for boundary domain
- Careful regarding the air layer

[†] Ghosh (1970); [‡] Hamilton (2000)

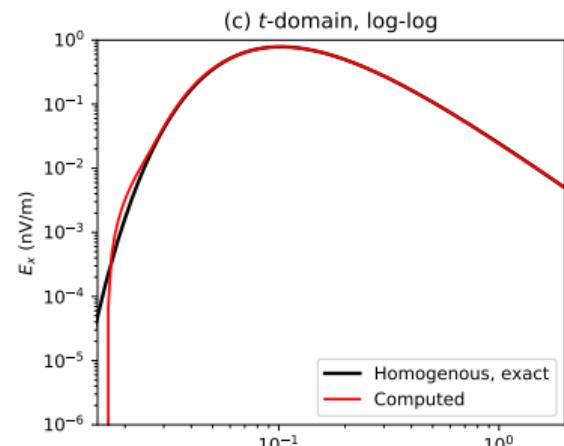
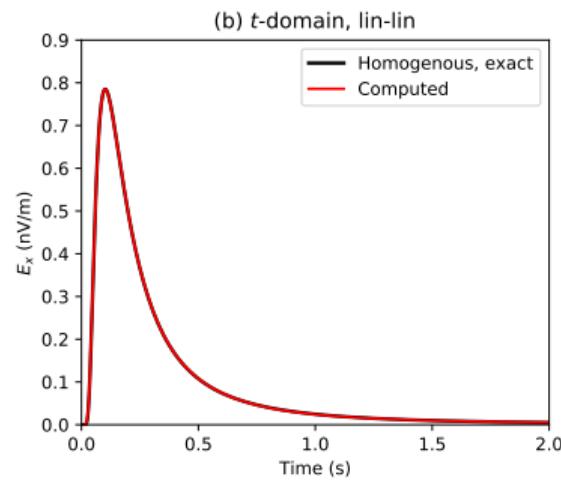
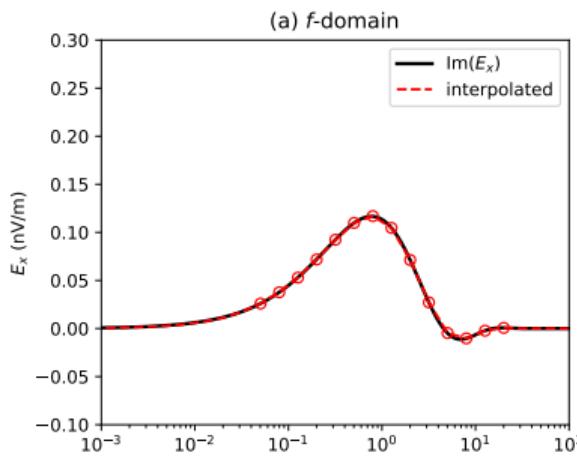
Mulder et al., 2008; Figures 2-4; homogeneous fullspace

- 26 frequencies, 0.01 Hz–100 Hz
- Runtime: 3h 47 min 12 s
- The peak value has an error of about 1 %



Our result, 2019

- 14 frequencies, 0.05 Hz–20 Hz
- Runtime: 1 min 40 s
- The peak value has an error of about 0.01 %



Analysis – Overview and gridding

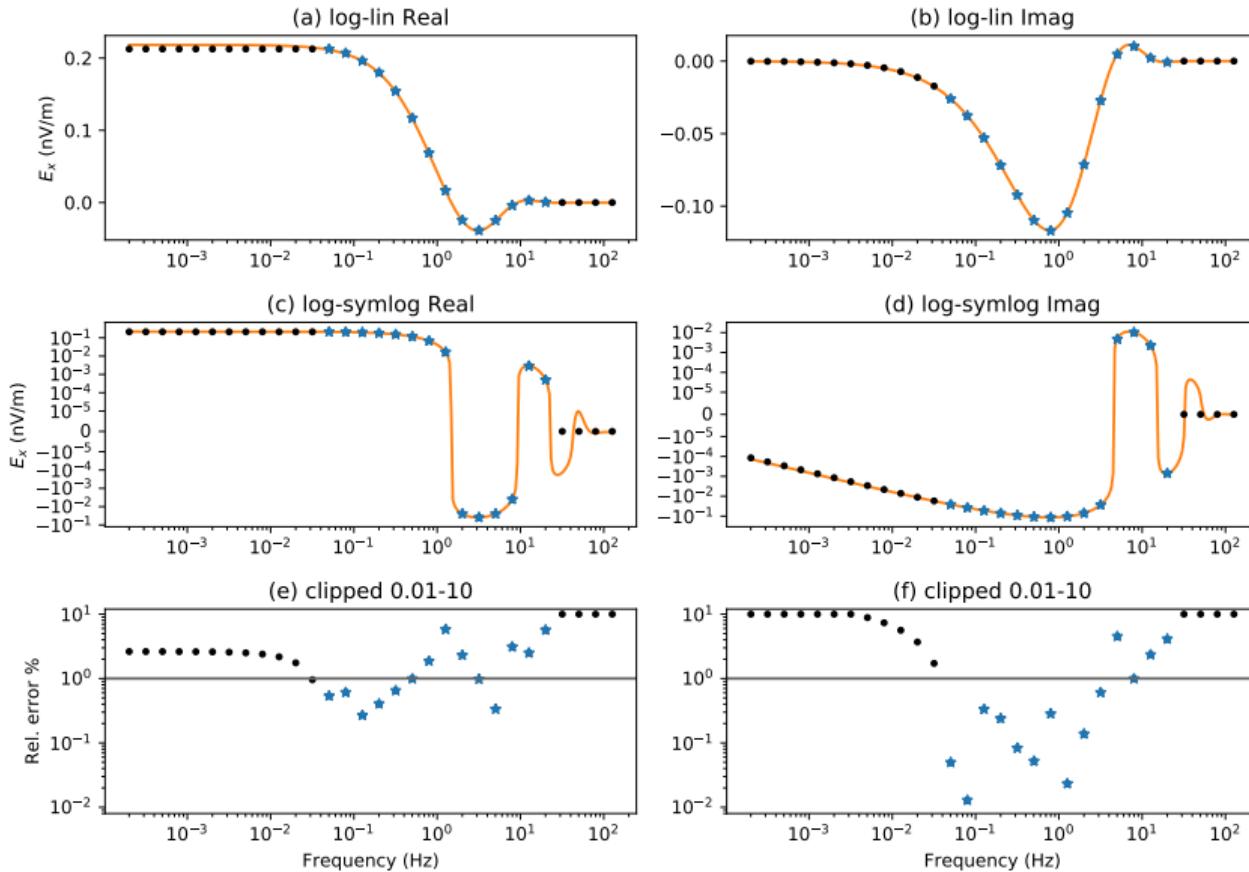
Main reasons for speed-up of 136x:

- ++ Automatic gridding; transform; cell sizes
- + Computers faster in the last 11 years
- ≈ Different implementation (Matlab/C vs Python/Numba)

**** TOTAL RUNTIME :: 0:01:40 ****

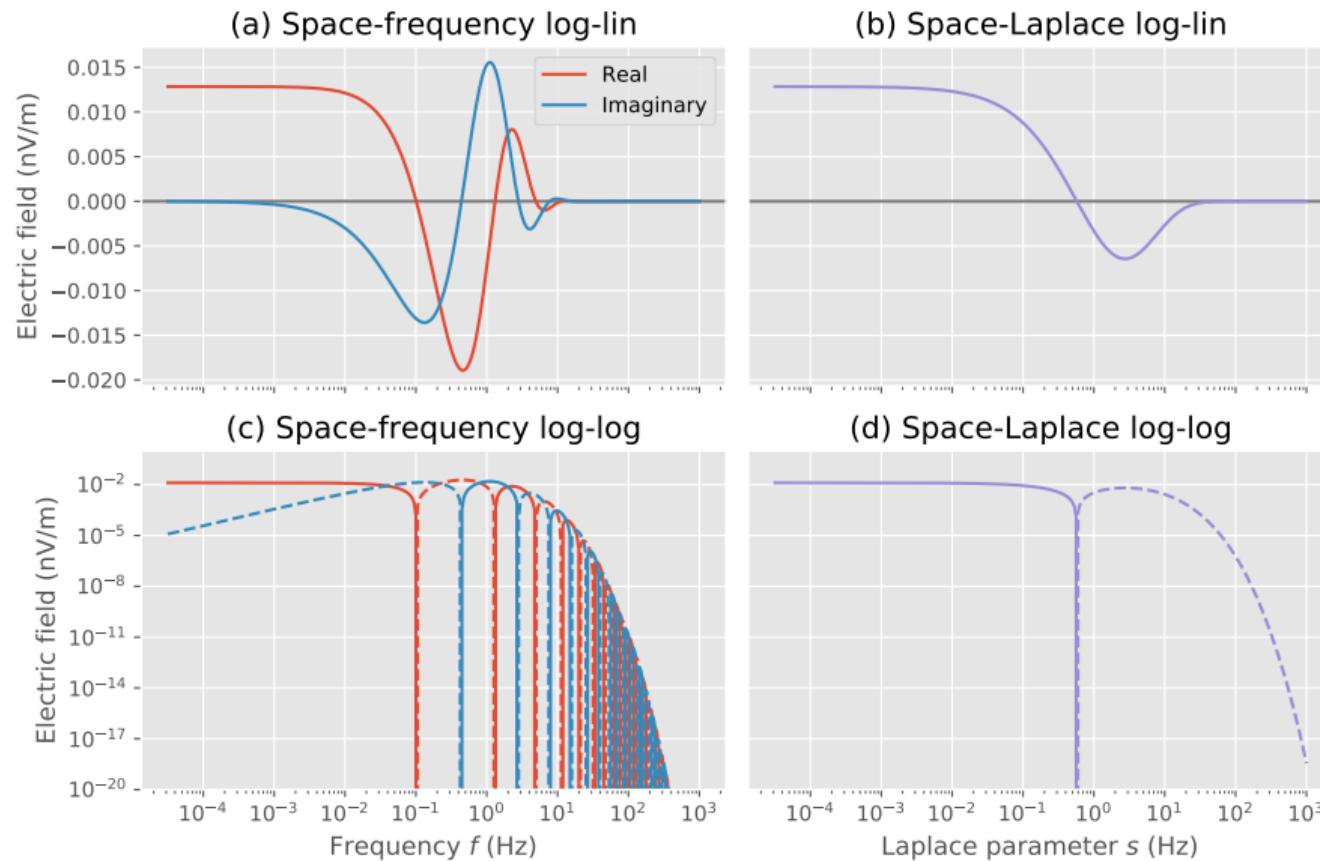
```
20.036 Hz: 4/1 it; 4 s; a: 1.00 / 1.18; n: 80 x 24 x 24; d: 20 / 138
12.642 Hz: 7/1 it; 6 s; a: 1.00 / 1.22; n: 80 x 24 x 24; d: 20 / 192
 7.977 Hz: 7/1 it; 6 s; a: 1.00 / 1.25; n: 80 x 24 x 24; d: 20 / 265
 5.033 Hz: 7/1 it; 6 s; a: 1.00 / 1.29; n: 80 x 24 x 24; d: 20 / 364
 3.176 Hz: 7/1 it; 5 s; a: 1.00 / 1.30; n: 80 x 24 x 24; d: 24 / 422
 2.004 Hz: 7/1 it; 4 s; a: 1.00 / 1.30; n: 64 x 24 x 24; d: 30 / 531
 1.264 Hz: 7/1 it; 4 s; a: 1.00 / 1.30; n: 64 x 24 x 24; d: 37 / 669
 0.798 Hz: 7/1 it; 7 s; a: 1.00 / 1.20; n: 64 x 24 x 24; d: 40 / 616
 0.503 Hz: 7/1 it; 7 s; a: 1.00 / 1.23; n: 64 x 24 x 24; d: 40 / 893
 0.318 Hz: 7/1 it; 9 s; a: 1.00 / 1.25; n: 64 x 24 x 24; d: 40 / 1137
 0.200 Hz: 7/1 it; 7 s; a: 1.00 / 1.28; n: 64 x 24 x 24; d: 40 / 1623
 0.126 Hz: 7/1 it; 7 s; a: 1.00 / 1.30; n: 64 x 24 x 24; d: 40 / 2047
 0.080 Hz: 7/1 it; 13 s; a: 1.00 / 1.25; n: 64 x 32 x 32; d: 40 / 2220
 0.050 Hz: 7/1 it; 12 s; a: 1.00 / 1.27; n: 64 x 32 x 32; d: 40 / 2955
```

Analysis – Frequency Calculation



Laplace Domain

Laplace s : $-s\sigma\hat{\mathbf{E}} - \nabla \times \mu^{-1}\nabla \times \hat{\mathbf{E}} = s\hat{\mathbf{J}}$ \Rightarrow $s = i\omega = 2i\pi f$ or s



Laplace – faster calculation, faster convergence

Frequency domain

- 9–10 s per F-cycle
- 5 BiCGSTAB cycles; 9 MG cycles
- 1 min 32 s

```
[22:03:37] 1.565e-01 after      1 F-cycles 0 0
[22:03:47] 2.998e-02 after      2 F-cycles 0 0
[22:03:47] 2.047e-02 after 1 bicgstab-cycles
[22:03:57] 5.506e-03 after      3 F-cycles 0 0
[22:04:07] 6.357e-04 after      4 F-cycles 0 0
[22:04:08] 5.097e-04 after 2 bicgstab-cycles
[22:04:18] 1.757e-04 after      5 F-cycles 0 0
[22:04:28] 2.832e-05 after      6 F-cycles 0 0
[22:04:28] 1.956e-05 after 3 bicgstab-cycles
[22:04:38] 6.460e-06 after      7 F-cycles 0 0
[22:04:48] 1.568e-06 after      8 F-cycles 0 0
[22:04:49] 1.375e-06 after 4 bicgstab-cycles
[22:04:58] 8.652e-07 after      9 F-cycles 0 0
[22:04:59] 7.268e-07 after 5 bicgstab-cycles

> CONVERGED
> Solver steps : 5
> MG prec. steps : 9
> Final rel. error : 7.268e-07

:: emg3d END :: 22:04:59 :: runtime = 0:01:32
```

Laplace domain

- 7–8 s per F-cycle
- 4 BiCGSTAB cycles; 8 MG cycles
- 1 min 2 s ⇒ 1/3 faster!

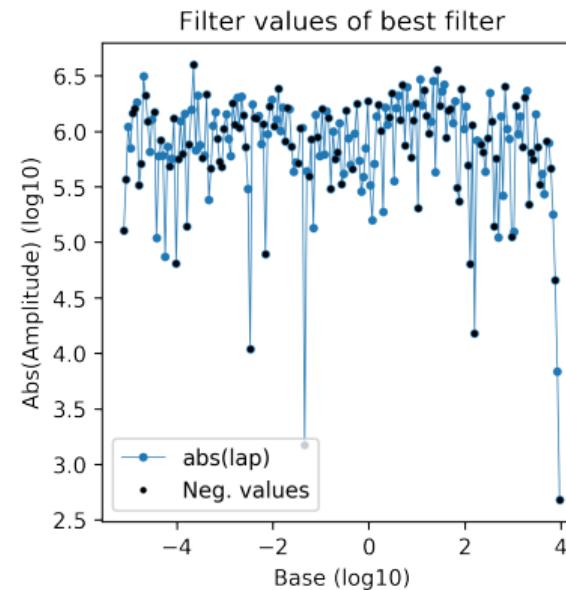
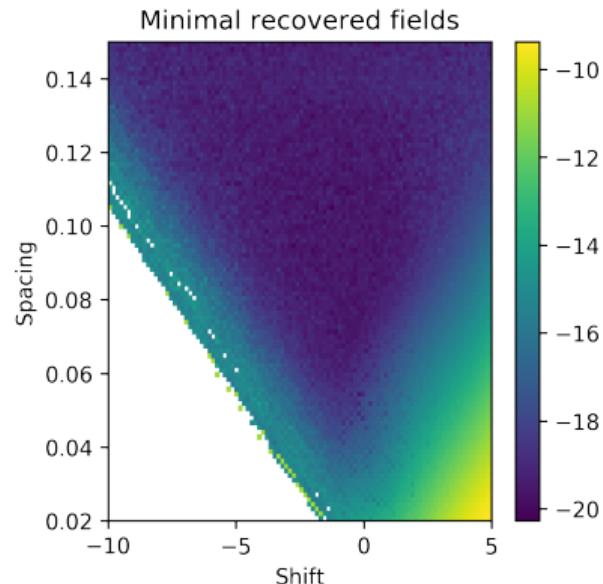
```
[22:00:48] 1.508e-01 after      1 F-cycles 0 0
[22:00:56] 2.805e-02 after      2 F-cycles 0 0
[22:00:56] 1.900e-02 after 1 bicgstab-cycles
[22:01:04] 4.924e-03 after      3 F-cycles 0 0
[22:01:11] 5.519e-04 after      4 F-cycles 0 0
[22:01:12] 4.401e-04 after 2 bicgstab-cycles
[22:01:19] 1.468e-04 after      5 F-cycles 0 0
[22:01:27] 2.277e-05 after      6 F-cycles 0 0
[22:01:27] 1.551e-05 after 3 bicgstab-cycles
[22:01:35] 4.829e-06 after      7 F-cycles 0 0
[22:01:42] 8.981e-07 after      8 F-cycles 0 0
[22:01:43] 7.426e-07 after 4 bicgstab-cycles

> CONVERGED
> Solver steps : 4
> MG prec. steps : 8
> Final rel. error : 7.426e-07

:: emg3d END :: 22:01:43 :: runtime = 0:01:02
```

DLF Laplace-to-time

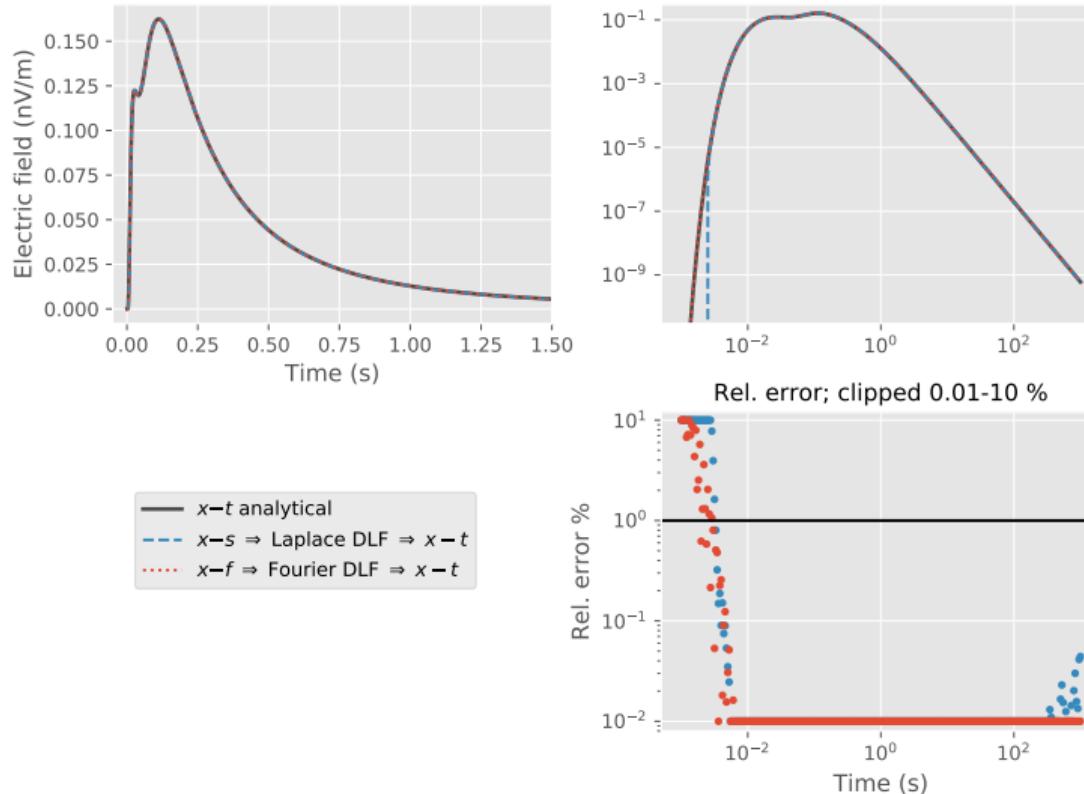
$$F(r) = \int_0^\infty f(l)K(lr)dl \quad r = e^x, l = e^{-y} \rightarrow \quad F(r) \approx \sum_{n=1}^N \frac{f(b_n/r)h_n}{r}$$



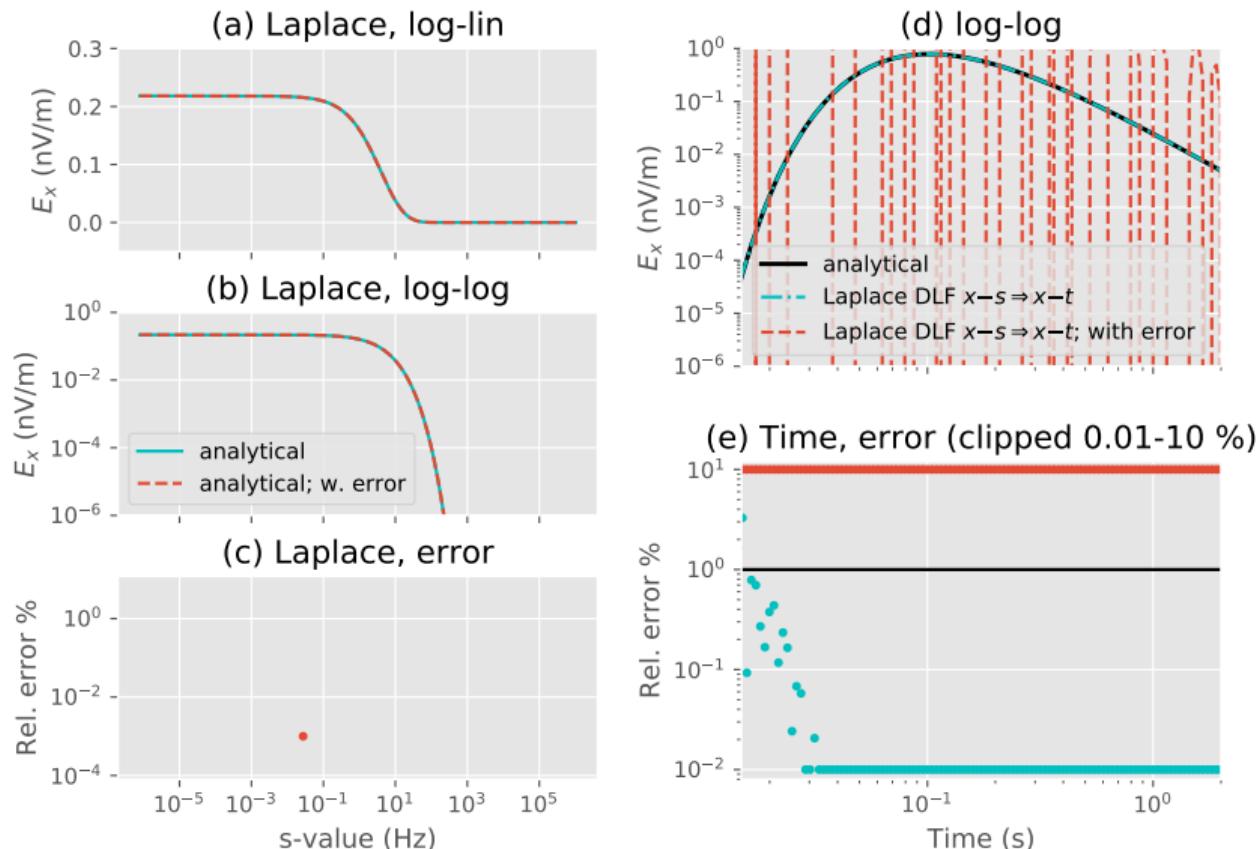
† DLF-method: Ghosh (1970); Used tool: Werthmüller et al. (2019)

Test using empymod

Half-space; $z_s = 150$ m, $z_r = 200$ m, $r = 2.0$ km.



Laplace – It doesn't work



Summary

Summary

- **emg3d**
- **Frequency-domain** calculation
- **Laplace-domain** calculation
- **Time-domain** calculation
 - Significant speed-up using the f -domain
 - Not working using the s -domain

Slides: github.com/empymod ⇒ presentations ⇒ ThursdayTalk2019

References

- Aminzadeh, F., Brac, J., and Kunz, T., 1997,** SEG/EAGE 3-D Salt and Overthrust Models: SEG, Tulsa, Oklahoma; wiki.seg.org/wiki/SEG/EAGE_Salt_and_Overthrust_Models.
- Ghosh, D. P., 1970,** The application of linear filter theory to the direct interpretation of geoelectrical resistivity measurements: Ph.D. Thesis, TU Delft; uuid: 88a568bb-ebee-4d7b-92df-6639b42da2b2.
- Hamilton, A. J. S., 2000,** Uncorrelated modes of the non-linear power spectrum: Monthly Notices of the Royal Astronomical Society, 312, pages 257–284; doi: 10.1046/j.1365-8711.2000.03071.x.
- Jönsthövel, T. B., C. W. Oosterlee, and W. A. Mulder, 2006,** Improving multigrid for 3-D electro-magnetic diffusion on stretched grids: European Conference on Computational Fluid Dynamics; UUID: df65da5c-e43f-47ab-b80d-2f8ee7f35464.
- Mulder, W. A., 2006,** A multigrid solver for 3D electromagnetic diffusion: Geophysical Prospecting, 54, 633–649; doi: 10.1111/j.1365-2478.2006.00558.x.
- Mulder, W. A., 2007,** A robust solver for CSEM modelling on stretched grids: EAGE Technical Program Expanded Abstracts, D036; doi: 10.3997/2214-4609.201401567.

- Mulder, W. A., M. Wirianto, and E. Slob, 2008,** Time-domain modeling of electromagnetic diffusion with a frequency-domain code: Geophysics, 73, F1–F8; doi: 10.1190/1.2799093.
- Plessix, R.-E., M. Darnet, and W. A. Mulder, 2007,** An approach for 3D multisource, multifrequency CSEM modeling: Geophysics, 72, SM177–SM184; doi: 10.1190/1.2744234.
- Werthmüller, D., 2017,** An open-source full 3D electromagnetic modeler for 1D VTI media in Python: `empymod`: Geophysics, 82(6), WB9–WB19; doi: 10.1190/geo2016-0626.1.
- Werthmüller, D., K. Key, and E. C. Slob, 2019,** A tool for designing digital filters for the Hankel and Fourier transforms in potential, diffusive, and wavefield modeling: Geophysics, 84(2), F47–F56; doi: 10.1190/geo2018-0069.1.
- Werthmüller, D., W. A. Mulder, and E. C. Slob, 2019,** `emg3d`: A multigrid solver for 3D electromagnetic diffusion: Journal of Open Source Software, 4(39), 1463; doi: 10.21105/joss.01463.

Swung (software underground)



Hackathon EAGE 2020: 06/07 June 2020 (#amstel-hack-2020)

Appendix

3D CSEM modelling in the frequency and Laplace domains

One story of success and one of failure

Dieter Werthmüller

5th December 2019

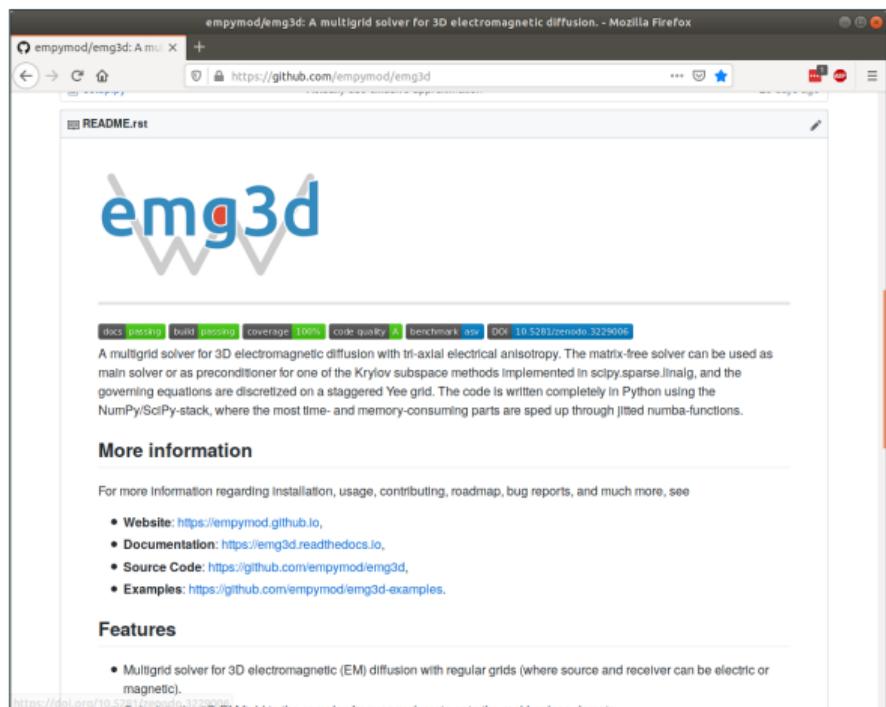
Thursday Talk – Applied Geophysics & Petrophysics



Faculty of Civil Engineering and Geosciences > Department of Geoscience & Engineering > Section of Applied Geophysics & Petrophysics

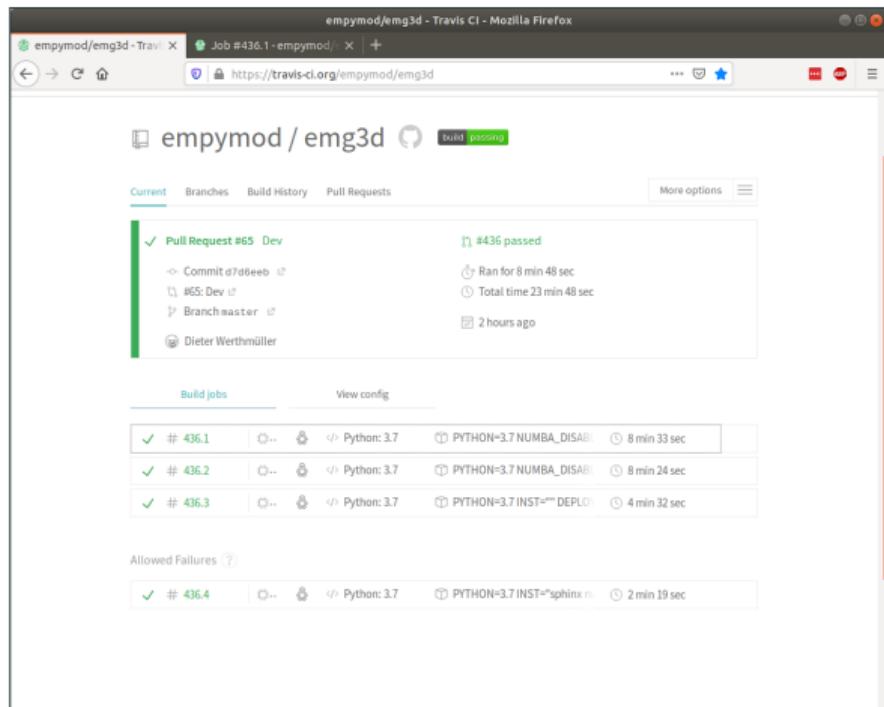
emg3d – CI workflow

- Commit to GitHub
- Travis CI - run all tests
 - Report to coveralls
 - Check all hyperlinks
 - Deploy to PyPi (*if tag*)
 - Deploy to conda-forge (*if tag*)
- Codacy
- ReadTheDocs and Gallery
- Zenodo (*if tag*)
- Benchmarks (asv, *manual*)



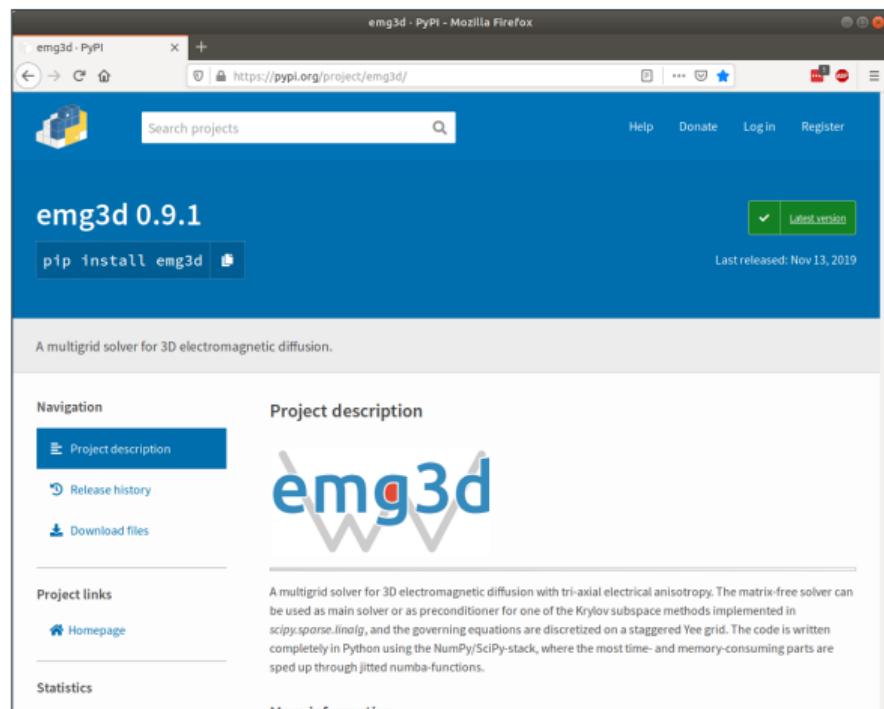
emg3d – CI workflow

- Commit to GitHub
- Travis CI - run all tests
 - Report to coveralls
 - Check all hyperlinks
 - Deploy to PyPi (*if tag*)
 - Deploy to conda-forge (*if tag*)
- Codacy
- ReadTheDocs and Gallery
- Zenodo (*if tag*)
- Benchmarks (asv, *manual*)



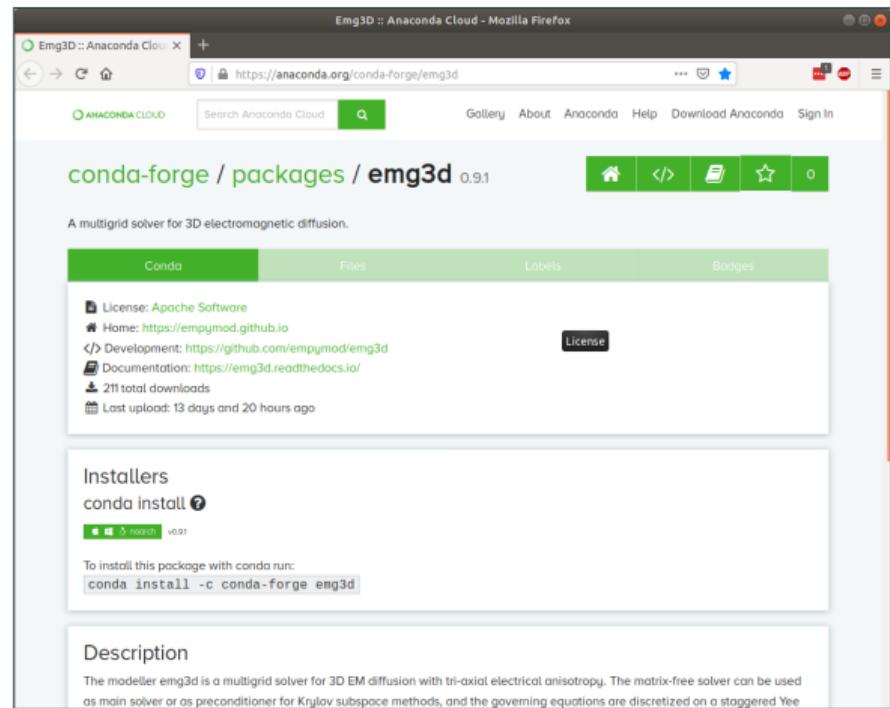
emg3d – CI workflow

- Commit to GitHub
- Travis CI - run all tests
 - Report to coveralls
 - Check all hyperlinks
 - Deploy to PyPi (*if tag*)
 - Deploy to conda-forge (*if tag*)
- Codacy
- ReadTheDocs and Gallery
- Zenodo (*if tag*)
- Benchmarks (*asv, manual*)



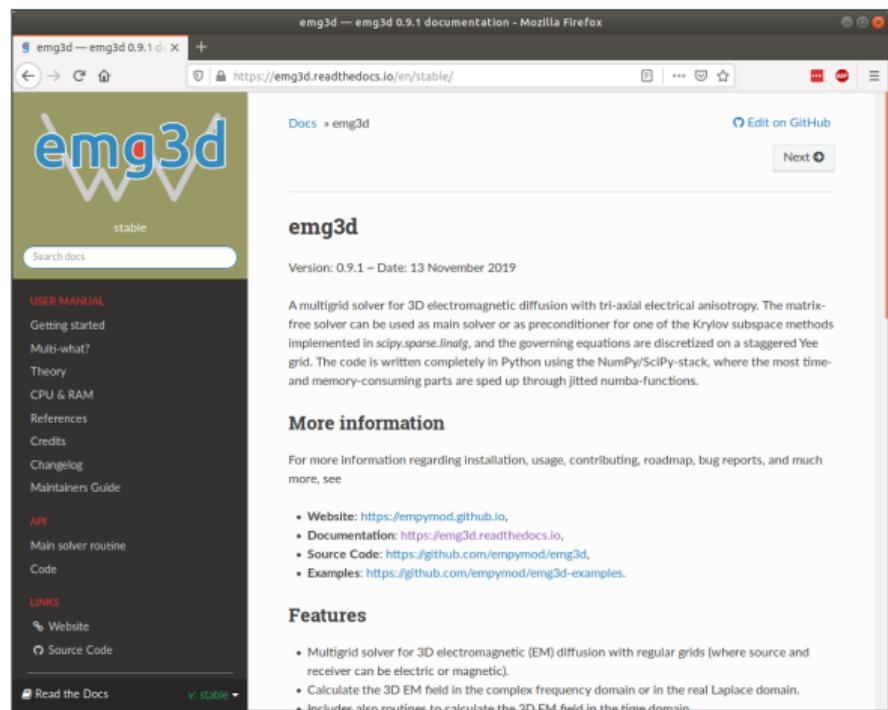
emg3d – CI workflow

- Commit to GitHub
- Travis CI - run all tests
 - Report to coveralls
 - Check all hyperlinks
 - Deploy to PyPi (*if tag*)
 - Deploy to conda-forge (*if tag*)
- Codacy
- ReadTheDocs and Gallery
- Zenodo (*if tag*)
- Benchmarks (asv, *manual*)



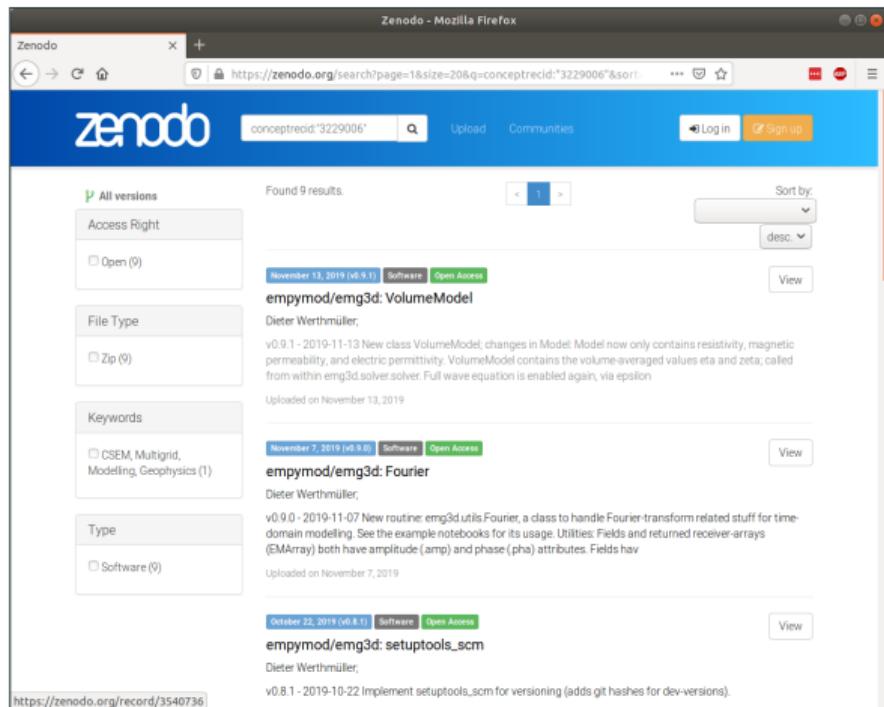
emg3d – CI workflow

- Commit to GitHub
- Travis CI - run all tests
 - Report to coveralls
 - Check all hyperlinks
 - Deploy to PyPi (*if tag*)
 - Deploy to conda-forge (*if tag*)
- Codacy
- ReadTheDocs and Gallery
- Zenodo (*if tag*)
- Benchmarks (asv, *manual*)



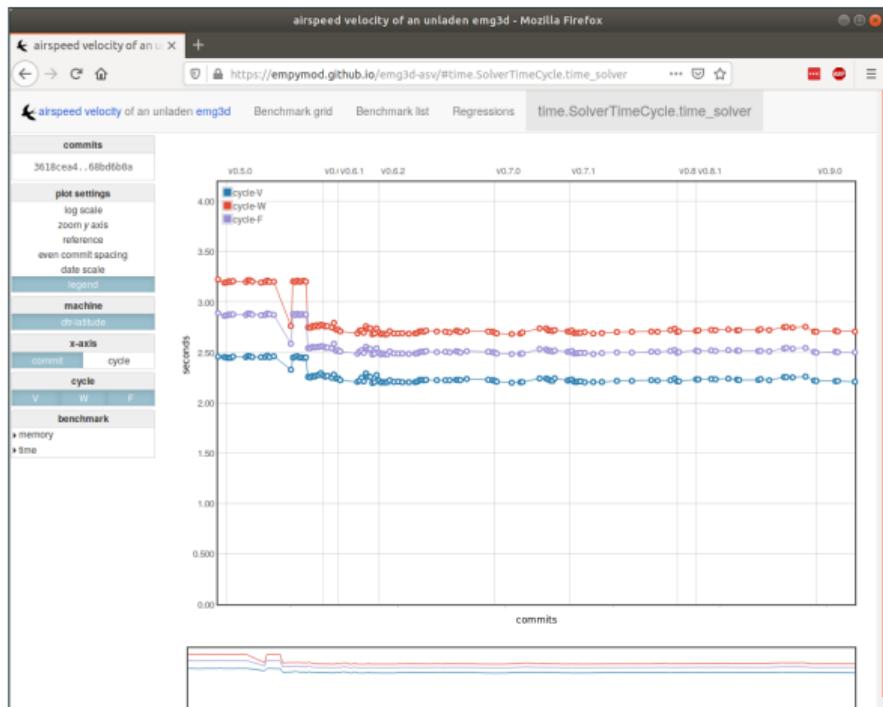
emg3d – CI workflow

- Commit to GitHub
- Travis CI - run all tests
 - Report to coveralls
 - Check all hyperlinks
 - Deploy to PyPi (*if tag*)
 - Deploy to conda-forge (*if tag*)
- Codacy
- ReadTheDocs and Gallery
- Zenodo (*if tag*)
- Benchmarks (asv, *manual*)



emg3d – CI workflow

- Commit to GitHub
- Travis CI - run all tests
 - Report to coveralls
 - Check all hyperlinks
 - Deploy to PyPi (*if tag*)
 - Deploy to conda-forge (*if tag*)
- Codacy
- ReadTheDocs and Gallery
- Zenodo (*if tag*)
- Benchmarks (asv, *manual*)



emg3d – Links

Website: empymod.github.io

Code: github.com/empymod/emg3d

Examples: github.com/empymod/emg3d-examples

Docs: emg3d.rtfd.io

Travis: travis-ci.org/empymod/emg3d

Coveralls: coveralls.io/github/empymod/emg3d

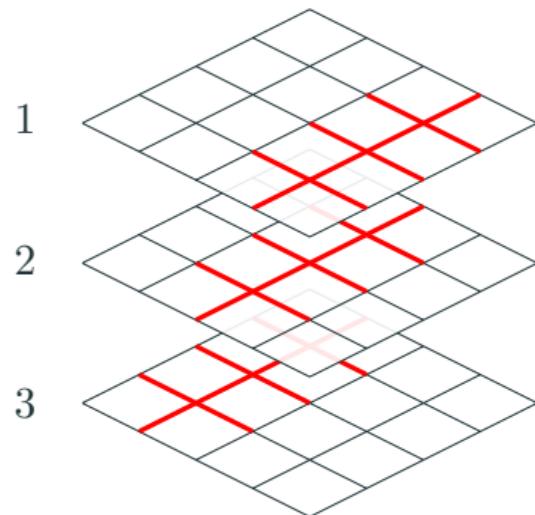
Codacy: app.codacy.com/project/prisae/emg3d/dashboard

Benchmarks: empymod.github.io/emg3d-asv

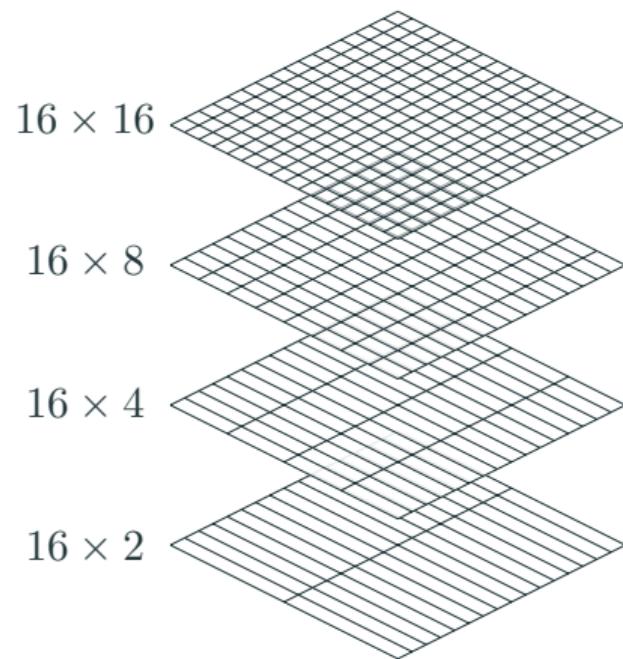
Zenodo: doi.org/10.5281/zenodo.3229006

Multigrid – Line relaxation & Semicoarsening

Line relaxation



Semicoarsening



Analysis – Result

