Modelling with $2\frac{1}{2}$D Approximations

Alistair Boyle

Carleton University

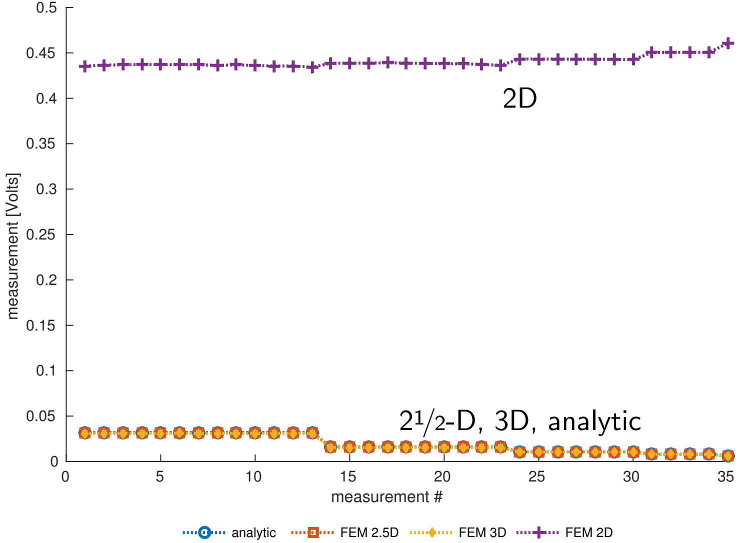
ICEBI & EIT Stockholm
June 19–23, 2016

Motivation

2D Models are Wrong

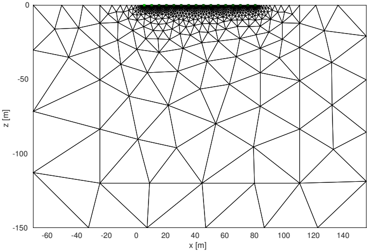
(some of the time)

Motivation

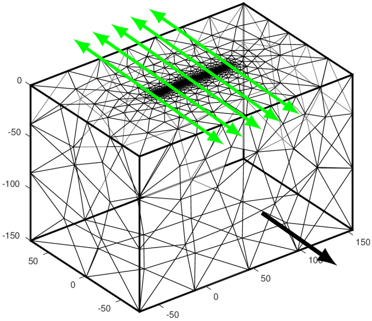


0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
measurement [Volts]
0
5
10
15
20
25
30
35
measurement #
2D
2½-D, 3D, analytic
analytic
FEM 2.5D
FEM 3D
FEM 2D

Motivation



=

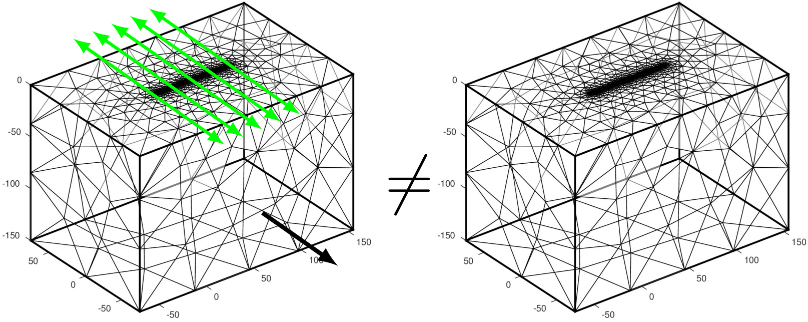


2D Models are Still 3D

with a uniform field in *z*

and *infinite* electrodes

Motivation



Electrodes are Finite

but detailed 3d models are expensive

What If

conductivity is uniform in z...

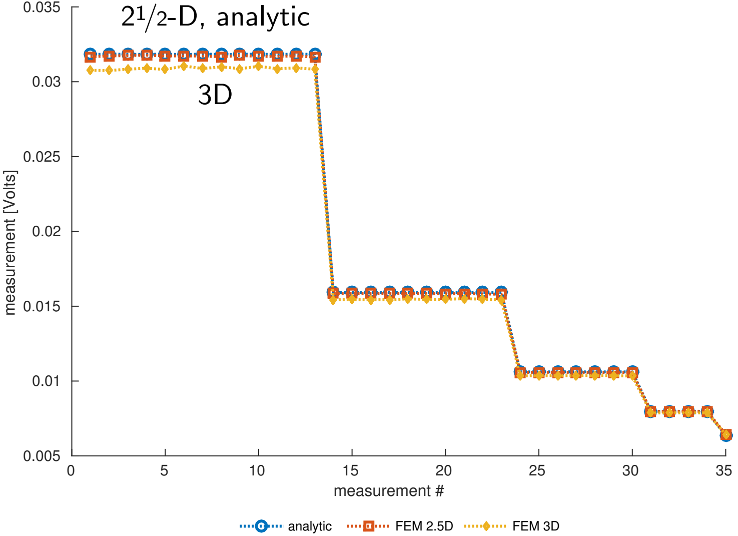
then we can make use of Fourier transforms

$$\tilde{\phi}_{xy\tilde{k}} = \int_0^\infty \phi_{xyz} \cos(\tilde{k}z) dz \qquad \mathcal{F} \qquad (1)$$

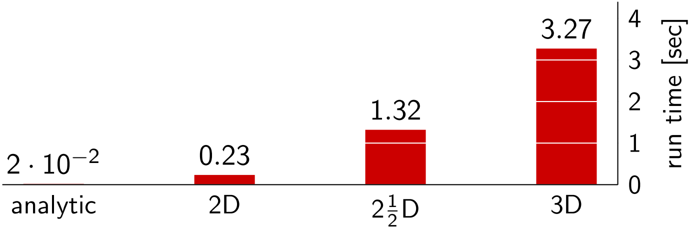
$$-\nabla \cdot (\sigma_{xy} \nabla \tilde{\phi}_{xy\tilde{k}}) + \tilde{\mathbf{k}}^2 \boldsymbol{\sigma}_{\mathbf{xy}} \tilde{\boldsymbol{\phi}}_{\mathbf{xy}\tilde{\mathbf{k}}} = \tilde{\mathbf{Q}} \delta_{xy} \qquad (2)$$

$$\phi_{xyz} = \frac{\mathbf{2}}{\boldsymbol{\pi}} \int_{\mathbf{0}}^\infty \tilde{\boldsymbol{\phi}}_{\mathbf{xy}\tilde{\mathbf{k}}} \cos(\tilde{\mathbf{k}}\mathbf{z}) \mathbf{d}\tilde{\mathbf{k}} \qquad \mathcal{F}^{-1} \qquad (3)$$

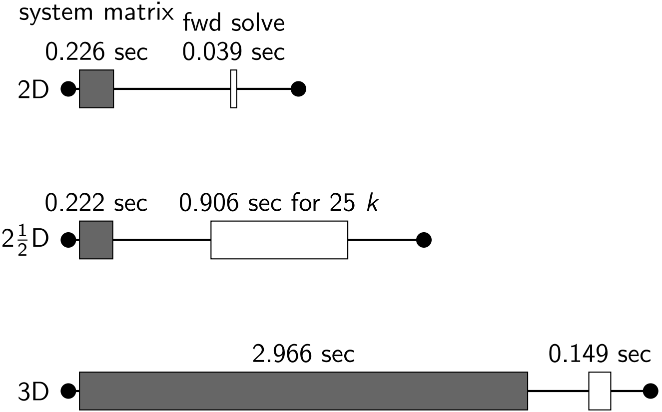
Fourier $2\frac{1}{2}$D



Compute Time

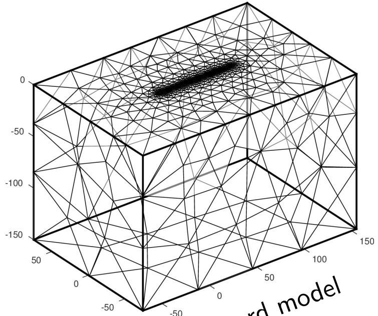


Compute Time



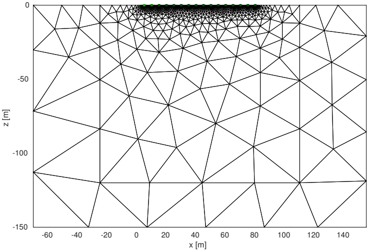
Another $2\frac{1}{2}$D Method

with a dual-mesh



forward model

+

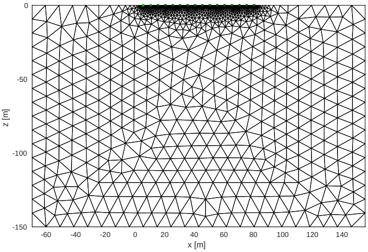


inverse model

same geometry but expensive forward solution

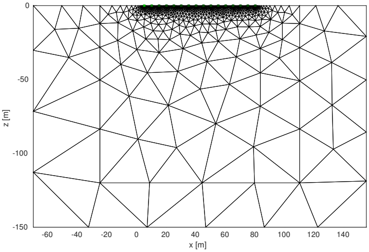
Another 2D Method

with a dual-mesh



forward model

+



inverse model

same geometry but *in*expensive forward solution

Now Available in EIDORS

```
% fwd_solve:
img.fwd_model.solve = @fwd_solve_2p5d_1st_order;
img.fwd_solve_2p5d_1st_order.k = [ a .. b ];  % optional
img.fwd_solve_2p5d_1st_order.method = 'name'; % optional
% k as integration range, default: [0 Inf]
% method as 'trapz' 'quadv' (default) or 'integral'

% inv_solve:
imdl.fwd_model.solve = @fwd_solve_2p5d_1st_order;
imdl.fwd_model.jacobian = @jacobian_adjoint_2p5d_1st_order;
imdl.fwd_model.system_mat = @system_mat_2p5d_1st_order;
```