joint inversions with the SimPEG framework

J. Capriotti[1,2], L. Heagy[1], S. Soler[1], T. Astic[3]

[1]University of British Columbia Geophysical Inversion Facility
[2]Colorado School of Mines
[3]Kobold Metals

IMAGE '23

motivation: joint inversion methods share enough similarities that a common framework is feasible and useful simpeg

what SimPEG solves…

inversion as optimization

$$\min_{\mathbf{m}} \phi(\mathbf{m}) = \phi_d(\mathbf{m}) + \beta\phi_m(\mathbf{m})$$

$$\text{s.t. } \phi_d \leq \phi_d^* \quad \mathbf{m}_L \leq \mathbf{m} \leq \mathbf{m}_U$$

requires:

- numerical simulation
- computation of sensitivities
- definition of regularization functional
- optimization machinery

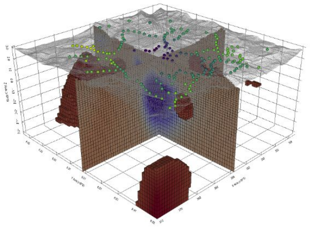
Simulation and Parameter Estimation in Geophysics

An open source python package for simulation and gradient based parameter estimation in geophysical applications.

Geophysical Methods

Contribute to a growing community of geoscientists building an open foundation for geophysics. SimPEG provides a collection of geophysical simulation and inversion tools that are built in a consistent framework.

- Gravity
- Magnetics
- Direct current resistivity
- Induced polarization
- Electromagnetics
  - Time domain
  - Frequency domain
  - Natural source (e.g. Magnetotellurics)
  - Viscous remanent magnetization
- Richards Equation



joint inversion

$$\begin{aligned}\phi(m_1, m_2, \ldots) = &\chi_1 \phi_{d,1}(\mathbf{m}) + \chi_2 \phi_{d,2}(\mathbf{m}) + \ldots \\ &+ \beta_1 \phi_{m,1}(m_1) + \beta_2 \phi_{m,2}(m_2) + \ldots \\ &+ \lambda \phi_{sim}(\mathbf{m})\end{aligned}$$

joint inversion

$$\phi(m_1, m_2, \ldots) = \boxed{\chi_1 \phi_{d,1}(\mathbf{m}) + \chi_2 \phi_{d,2}(\mathbf{m})} + \ldots$$
$$+ \beta_1 \phi_{m,1}(m_1) + \beta_2 \phi_{m,2}(m_2) + \ldots$$
$$+ \lambda \phi_{sim}(\mathbf{m})$$

Multiple data misfits

joint inversion

$$\begin{aligned}\phi(m_1, m_2, \ldots) =& \chi_1 \phi_{d,1}(\mathbf{m}) + \chi_2 \phi_{d,2}(\mathbf{m}) + \ldots \\ & \boxed{+\beta_1 \phi_{m,1}(m_1) + \beta_2 \phi_{m,2}(m_2)} + \ldots \\ & + \lambda \phi_{sim}(\mathbf{m})\end{aligned}$$

Multiple regularization functions

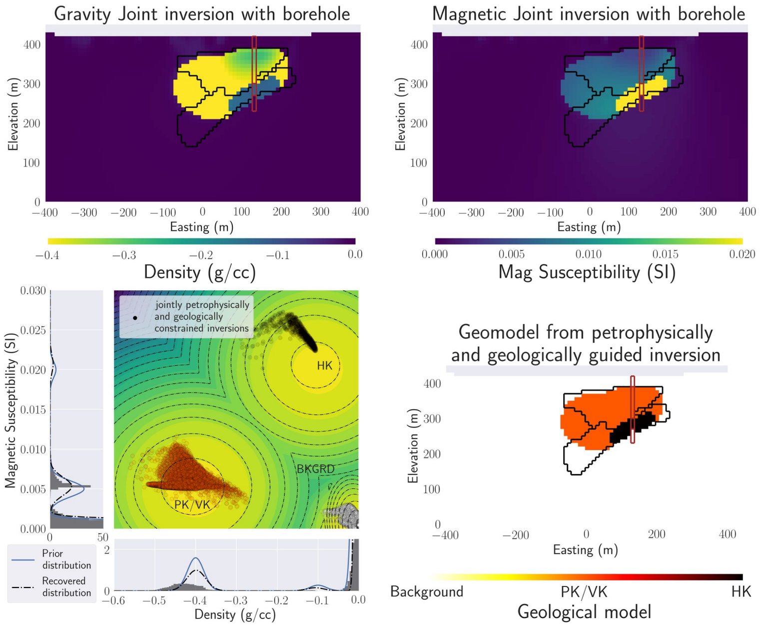
joint inversion

$$\begin{aligned}\phi(m_1, m_2, \ldots) = & \chi_1 \phi_{d,1}(\mathbf{m}) + \chi_2 \phi_{d,2}(\mathbf{m}) + \ldots \\ & + \beta_1 \phi_{m,1}(m_1) + \beta_2 \phi_{m,2}(m_2) + \ldots \\ & \boxed{+ \lambda \phi_{sim}(\mathbf{m})}\end{aligned}$$

Similarity measure

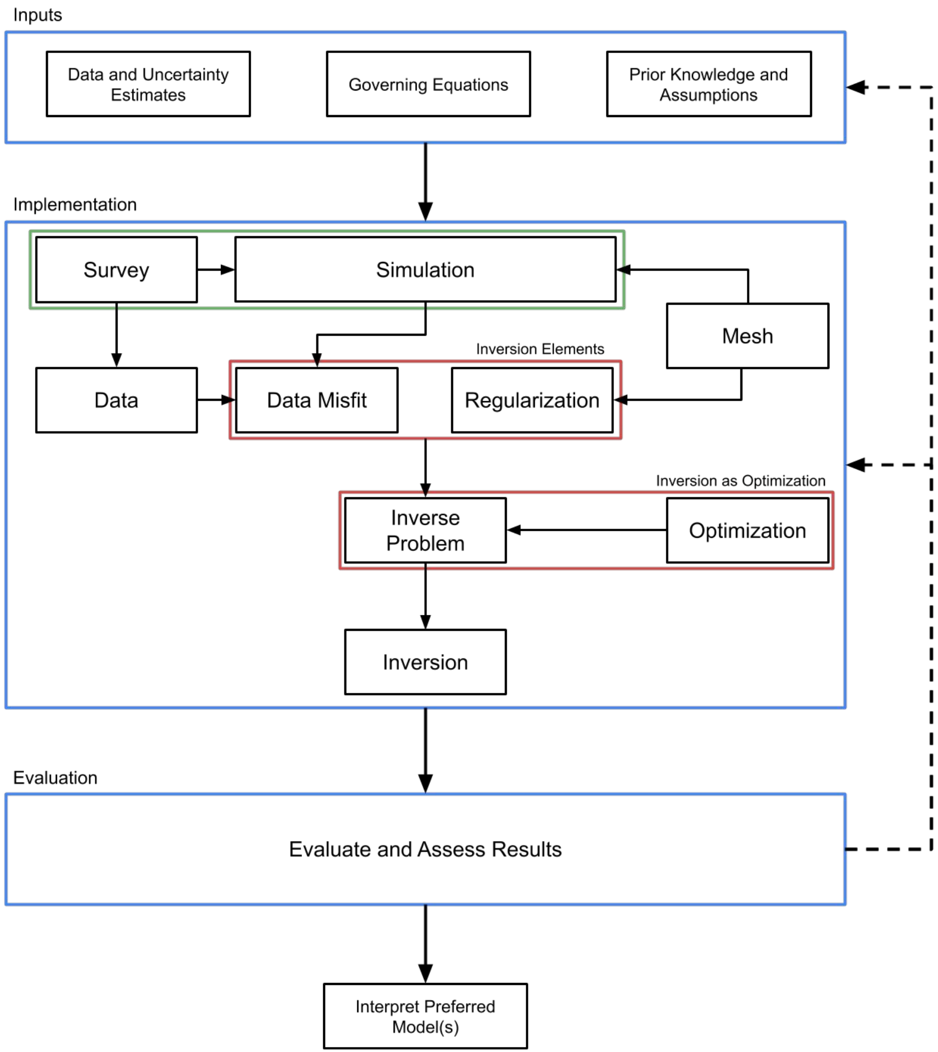
joint inversion methodologies

- structural approaches
  - structural similarity - Haber and Oldenburg 1997
  - cross-gradient - Gallardo and Meju 2003
  - structural gramian - Zhdanov 2012
  - joint total variation - Haber and Gazit 2013
  - …
- physical property based approaches
  - gramian - Zhdanov 2012
  - correspondence mappings - Haber and Gazit 2013
  - mutual information - Pluim et. al. 1999
  - fuzzy c-means - Lelièvre et. al. 2012, Sun and Li, 2015
  - petrophysically guided inversion - Astic et. al. 2021
  - …



[(Astic & Oldenburg, 2019)]

SimPEG framework



simulations

All simulations share a common calling convention

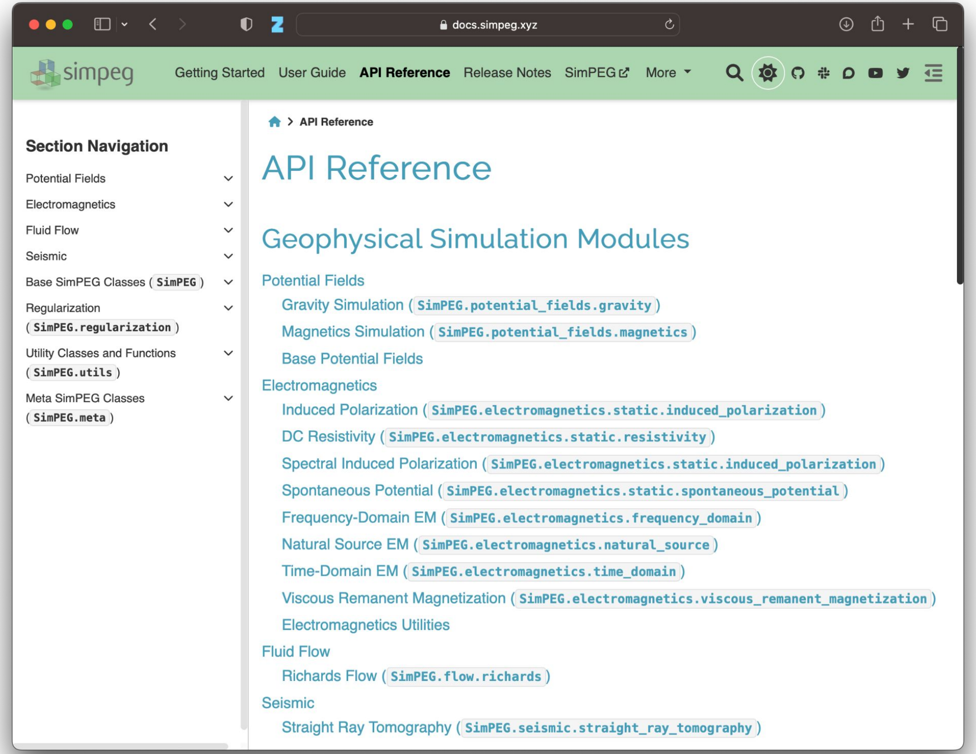
- forward modeling:

$$\phi_d(\mathbf{m}) = |W_d(\boxed{F(\mathbf{m})} - \mathbf{d})|^2$$

- jacobian vector operations

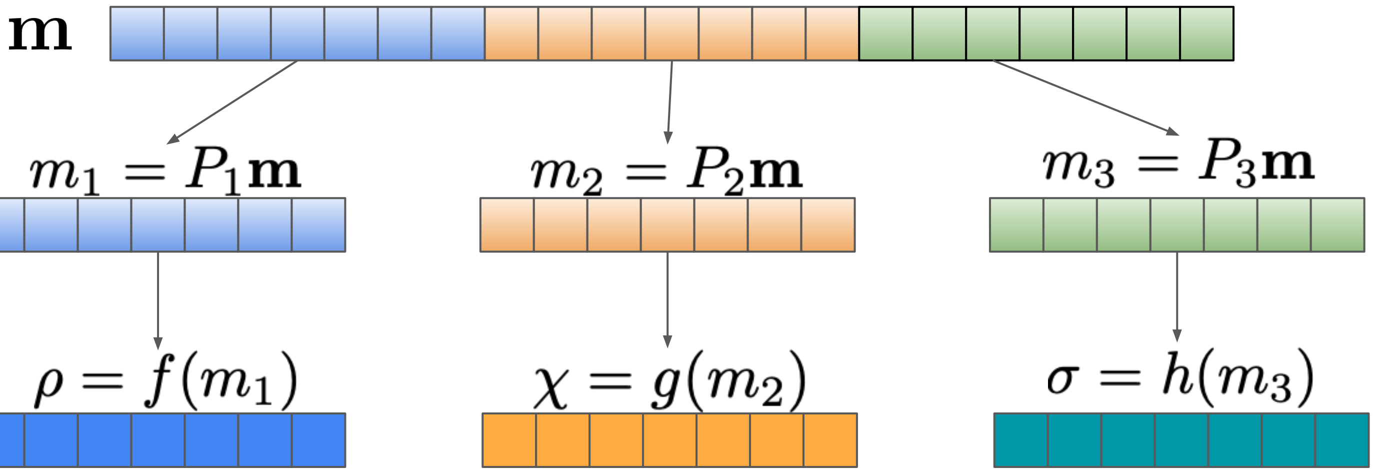
$$J(\mathbf{m})\mathbf{v}$$

$$J_{ij} = \frac{\partial d_i}{\partial m_j}$$



mappings

- transforms the inversion model to physical properties.
- automated chain rule derivatives
- joint inversions make use of `Projections` to select pieces of the model




```
sigma_map = ExpMap() * project_3
```

Composable!

objective functions

Composable objective functions

- Allows use of arbitrary minimizers (but most commonly use Gauss-Newton)
- Construct total objective function just like the math

$$
\begin{aligned}
\phi(m_1, m_2, \ldots) = &\chi_1 \phi_{d,1}(\mathbf{m}) + \chi_2 \phi_{d,2}(\mathbf{m}) + \ldots \\
&+ \beta_1 \phi_{m,1}(m_1) + \beta_2 \phi_{m,2}(m_2) + \ldots \\
&+ \lambda \phi_{sim}(\mathbf{m})
\end{aligned}
$$

```
obj = (
    chi_1 * data_misfit_1 + chi_2 * data_misfit_2
    + beta_1 * reg_1 + beta_2 * reg_2
    + lamb * cross_grad
)
```

directives

A list of rules on how to modify parameters during the inversion

Directive: Balance the multiple data misfits

$$
\begin{aligned}
\phi(m_1, m_2, \ldots) = & \boxed{\chi_1} \phi_{d,1}(\mathbf{m}) + \boxed{\chi_2} \phi_{d,2}(\mathbf{m}) + \ldots \\
& + \beta_1 \phi_{m,1}(m_1) + \beta_2 \phi_{m,2}(m_2) + \ldots \\
& + \lambda \phi_{sim}(\mathbf{m})
\end{aligned}
$$

directives

A list of rules on how to modify parameters during the inversion

Directive: Cool regularization parameters until target misfit is achieved

$$
\begin{aligned}
\phi(m_1, m_2, \dots) = & \chi_1 \phi_{d,1}(\mathbf{m}) + \chi_2 \phi_{d,2}(\mathbf{m}) + \dots \\
& + \boxed{\beta_1} \phi_{m,1}(m_1) + \boxed{\beta_2} \phi_{m,2}(m_2) + \dots \\
& + \lambda \phi_{sim}(\mathbf{m})
\end{aligned}
$$

directives

A list of rules on how to modify parameters during the inversion

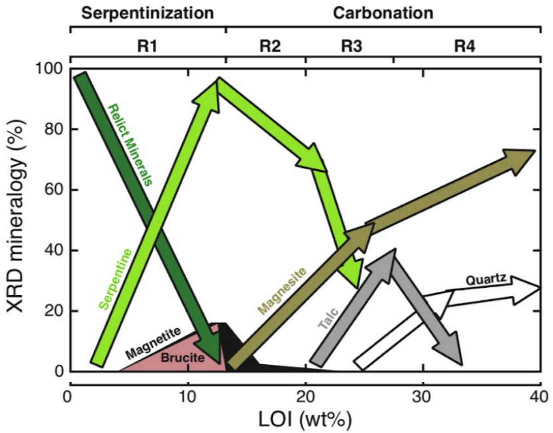
$$
\begin{aligned}
\phi(m_1, m_2, \ldots) = &\chi_1 \phi_{d,1}(\mathbf{m}) + \chi_2 \phi_{d,2}(\mathbf{m}) + \ldots \\
&+ \beta_1 \phi_{m,1}(m_1) + \beta_2 \phi_{m,2}(m_2) + \ldots \\
&+ \boxed{\lambda} \phi_{sim}(\mathbf{m})
\end{aligned}
$$

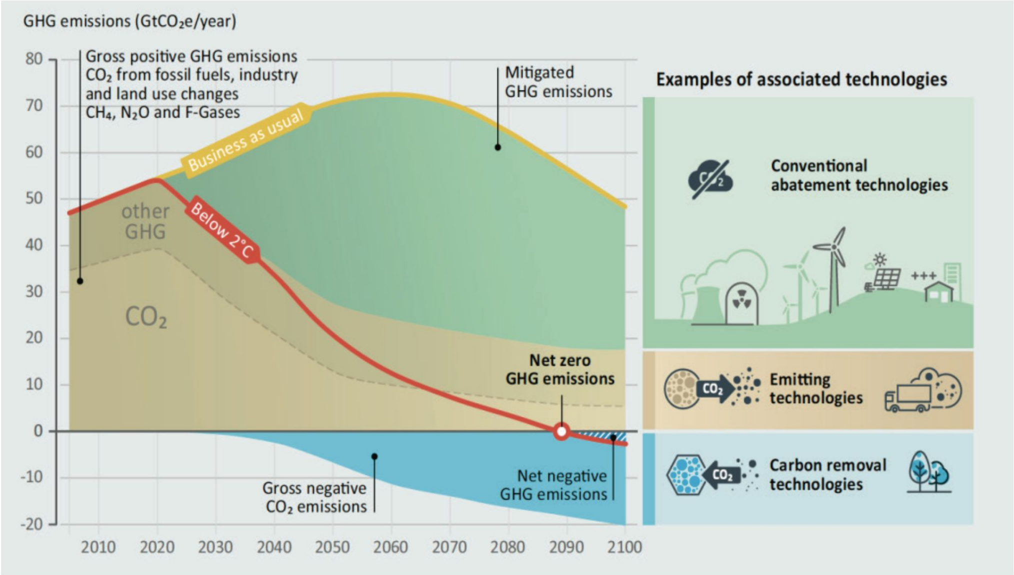
Directive: iteratively increase the similarity measure weight while still able to hit target misfits

geologic storage of CO_2

sedimentary settings: saline aquifers, depleted O&G reservoirs

carbon mineralization: reaction of CO_2 with mafic, ultramafic minerals

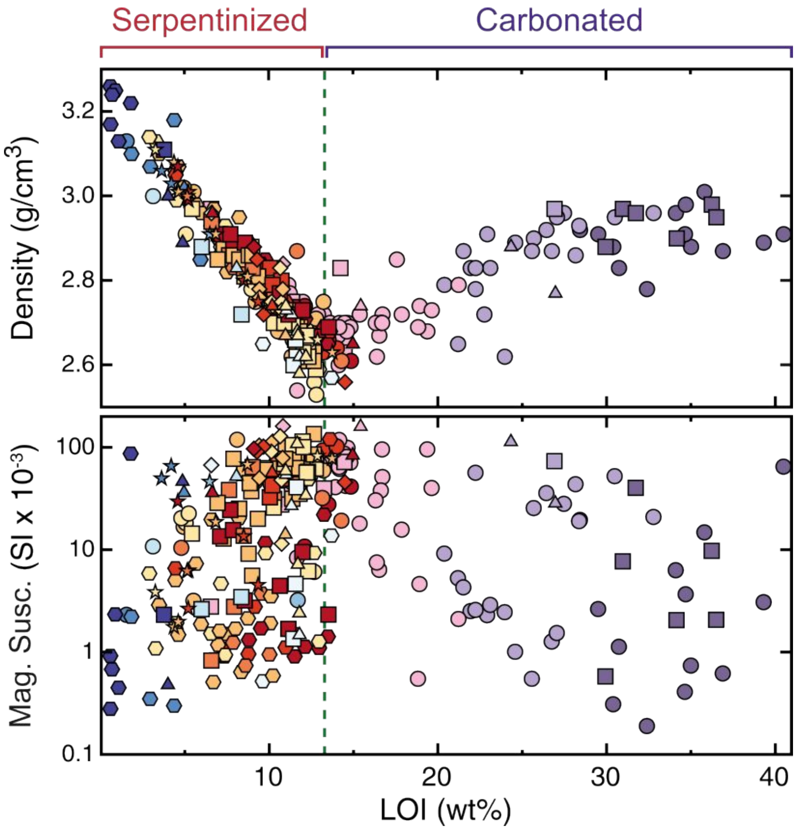




<u>Conclusion 4:</u> If the goals for climate and economic growth are to be achieved, negative emissions technologies will likely need to play a large role in mitigating climate change by removing ~10 Gt/y CO_2 globally by midcentury and ~20 Gt/y CO_2 globally by the century's end.

National academies report, 2019

physical properties



Loss of Ignition (LOI)

- **Proxy** variable for **alteration**
- 5%-13%: **high** carbonation potential
- **Density** and **susceptibility** change with LOI

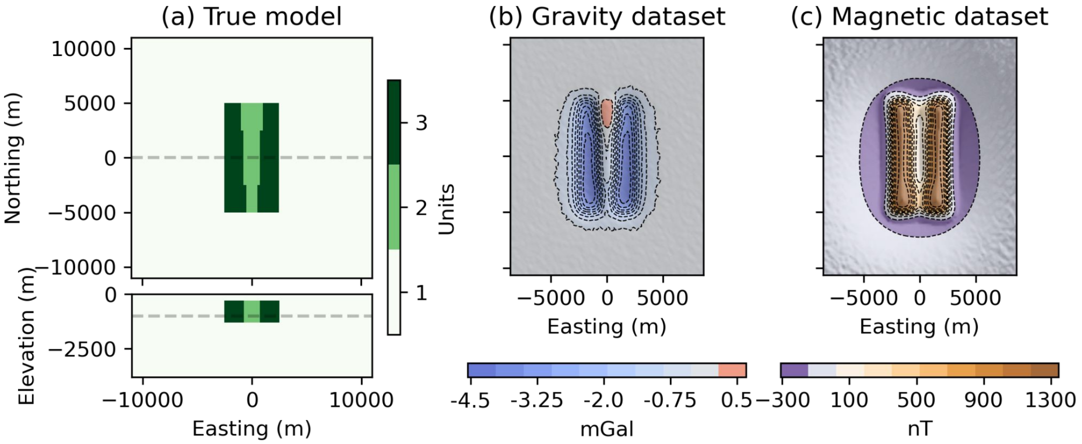
Serpentinized rocks with **good potential:**

- Low density
- Higher* susceptibility

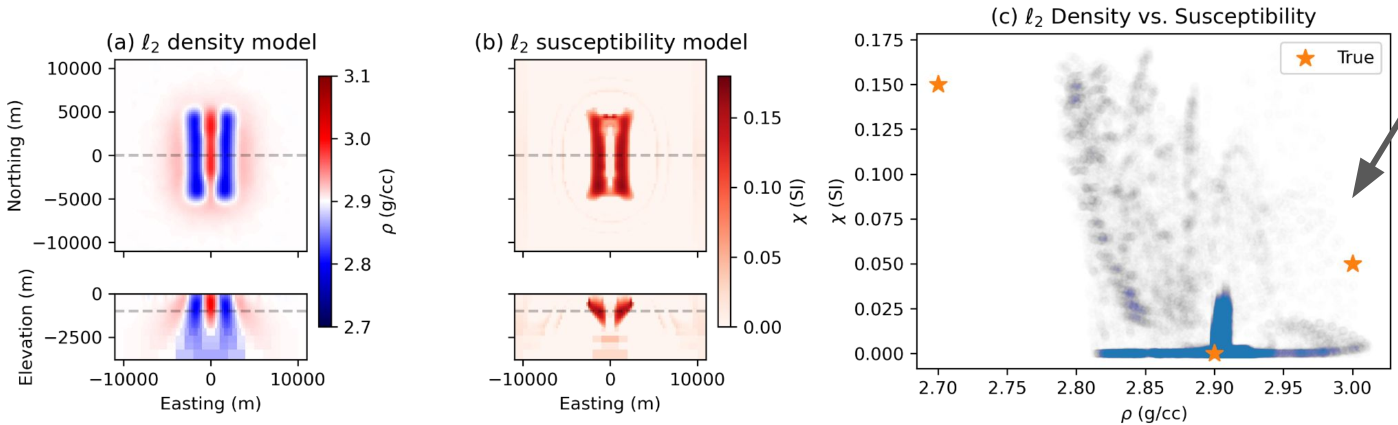
Cutts et al. (2021), doi: [10.1029/2021GC009989](https://doi.org/10.1029/2021GC009989)

simple synthetic model

- serpentinized region with central carbonated region
- L2 results show poor correlations



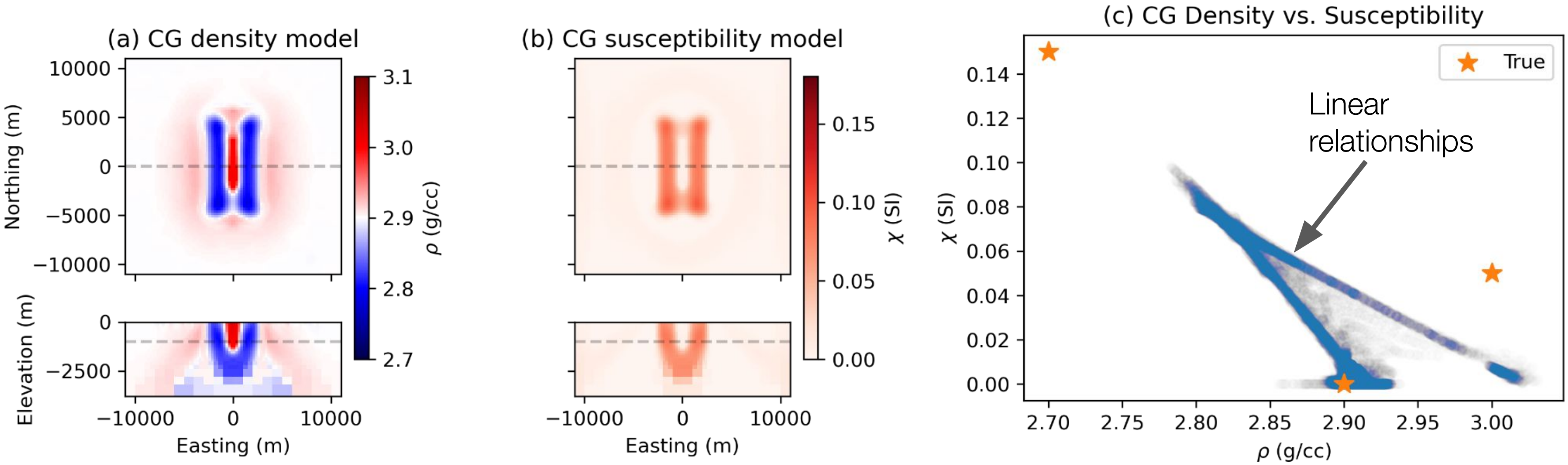
Poor correlations



cross gradient

$$\phi_{CG}(m_1, m_2) = \int_V |\nabla m_1 \times \nabla m_2|^2 dV$$

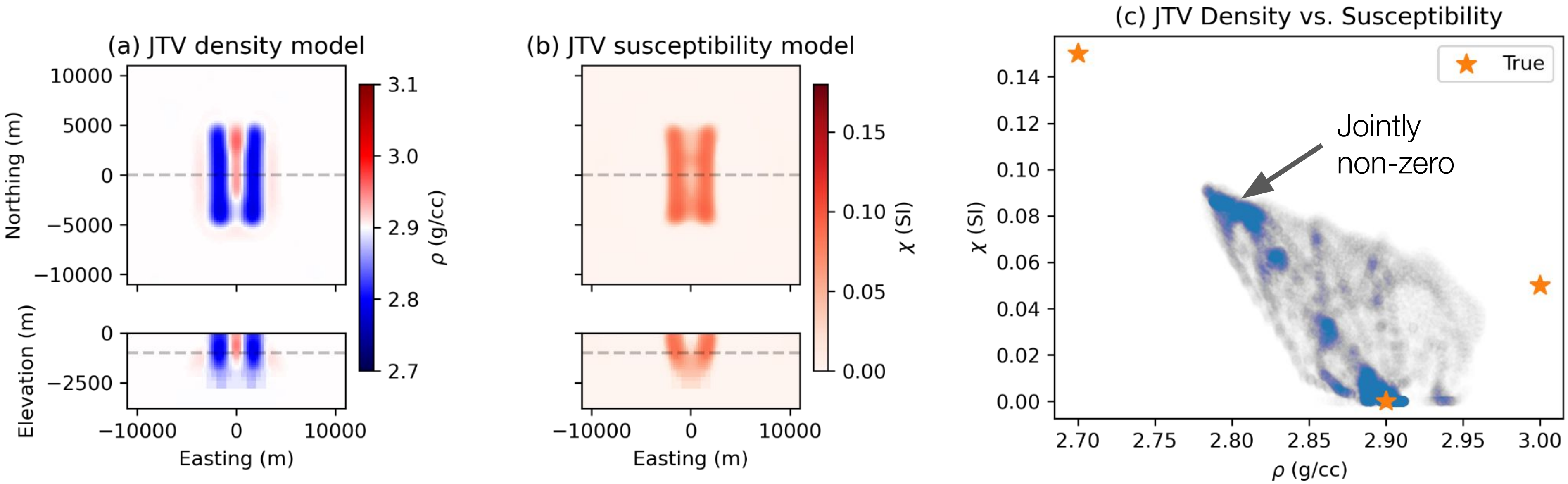
Encourages model spatial gradients to be parallel to each other



joint total variation

$$\phi_{jtv}(m_1, m_2) = \int_V \sqrt{|\nabla m_1|^2 + |\nabla m_2|^2} dV$$

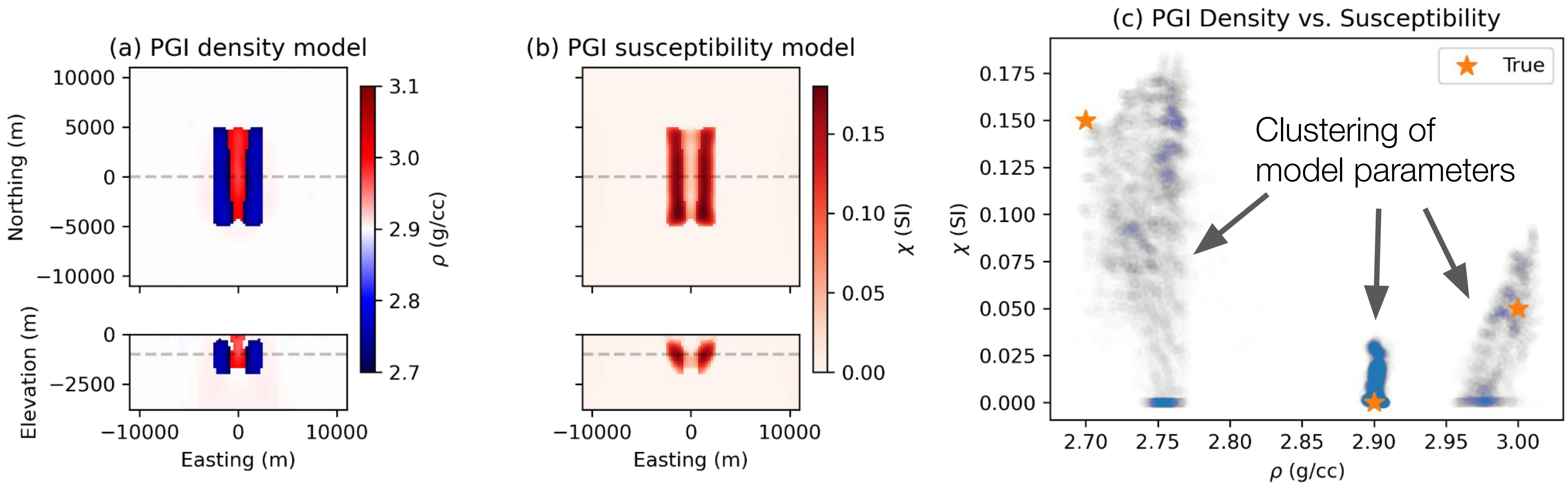
Encourages model spatial gradients to occur sparsely in the same locations



petrophysically guided inversion (PGI)

$$\phi_{m,1}(m_1) = \alpha_{s,1} \left| W_m(m_1 - m_{ref,1}) \right|^2$$
$$\phi_{m,2}(m_2) = \alpha_{s,2} \left| W_m(m_2 - m_{ref,2}) \right|^2$$

No direct similarity measure, instead use directives to update reference models using a common geology model

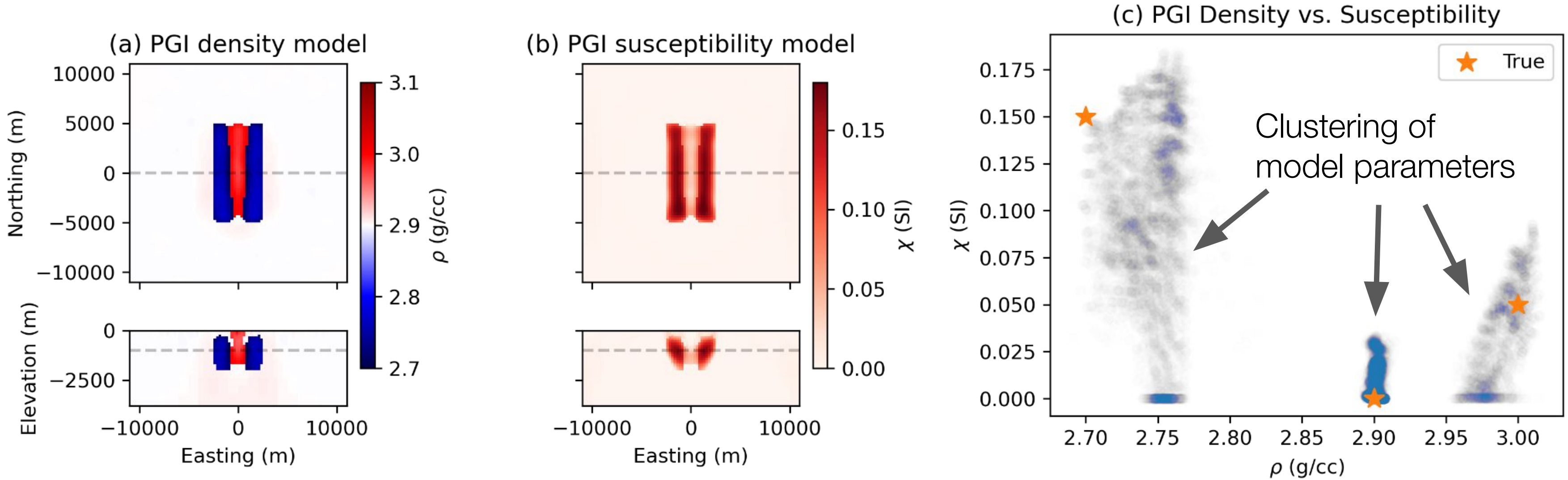


petrophysically guided inversion (PGI)

Quasi-Geo Model (from GMM)

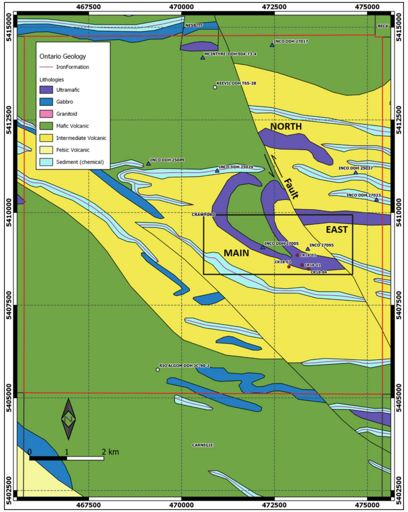
$$\phi_{m,1}(m_1) = \boxed{\alpha_{s,1}} |W_m(m_1 - \boxed{m_{ref,1}})|^2$$
$$\phi_{m,2}(m_2) = \boxed{\alpha_{s,2}} |W_m(m_2 - \boxed{m_{ref,2}})|^2$$

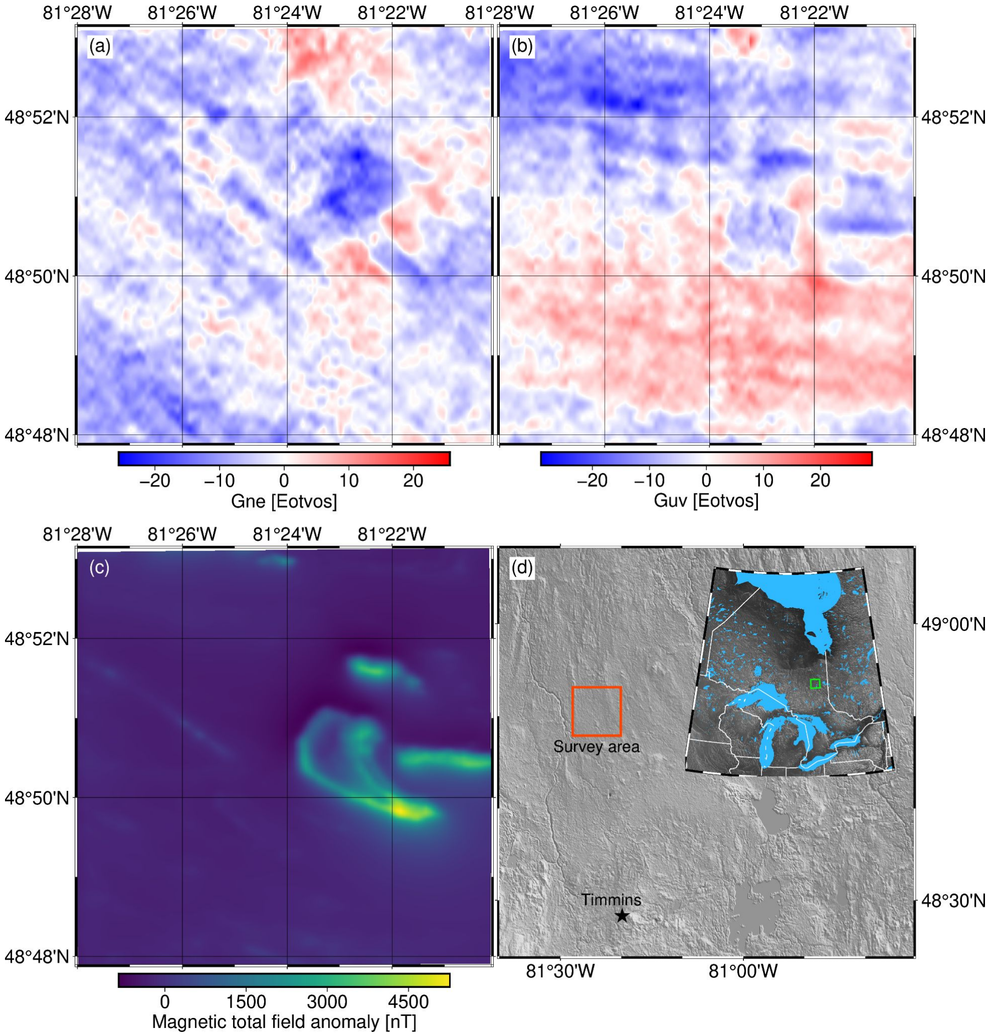
No direct similarity measure, instead use directives to update reference models using a common geology model



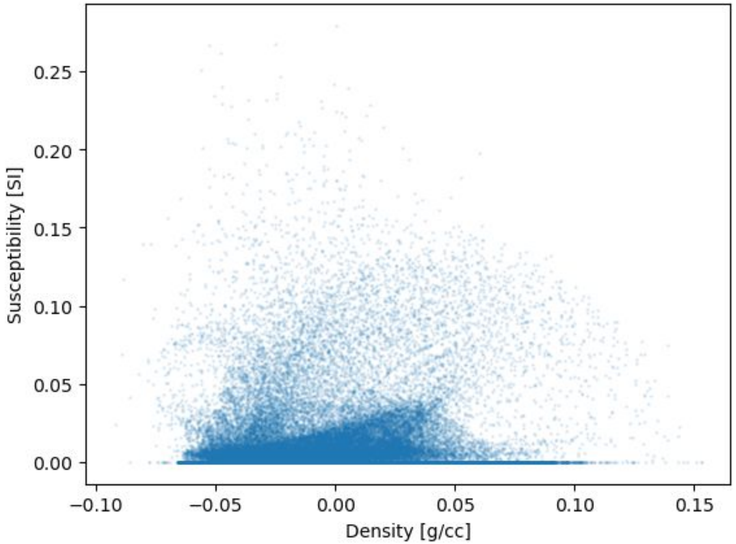
Crawford Fe-Ni deposit

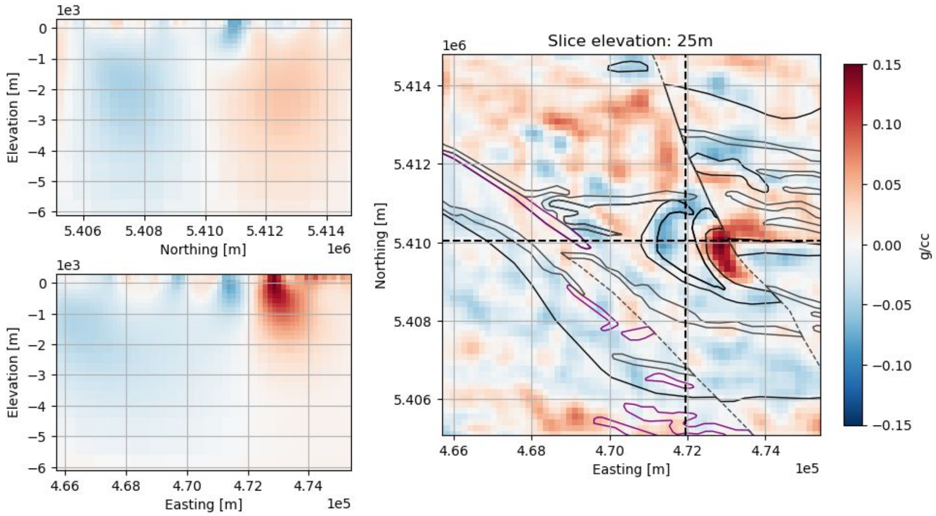
- Falcon Gravity Gradient
- Airborne TMI data
- Ultramafics (Serpentinized)

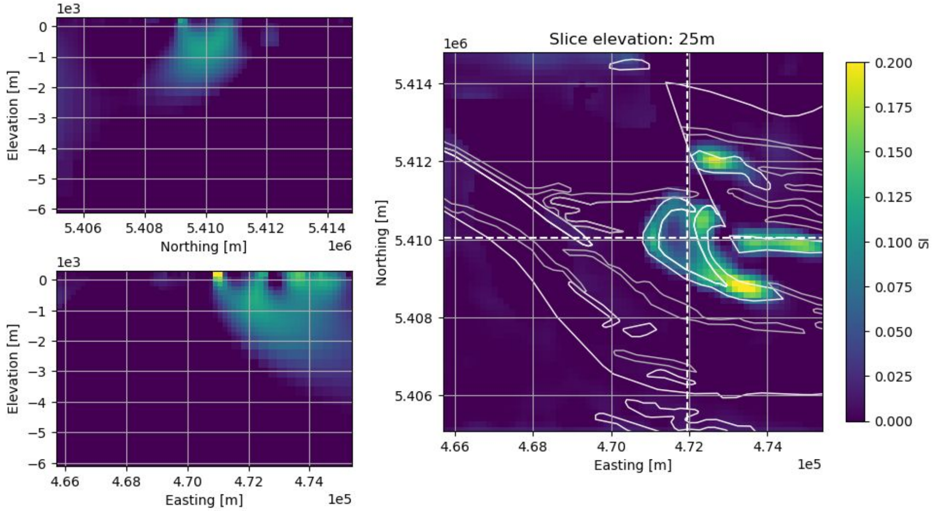




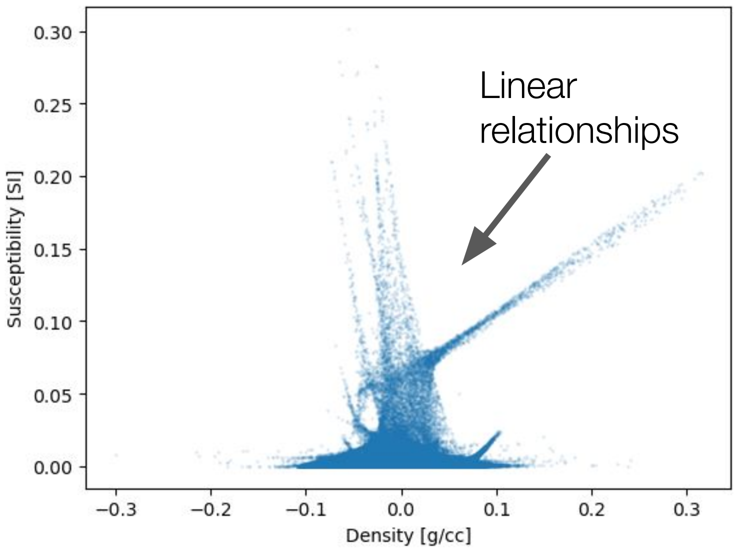
separate l2 inversions

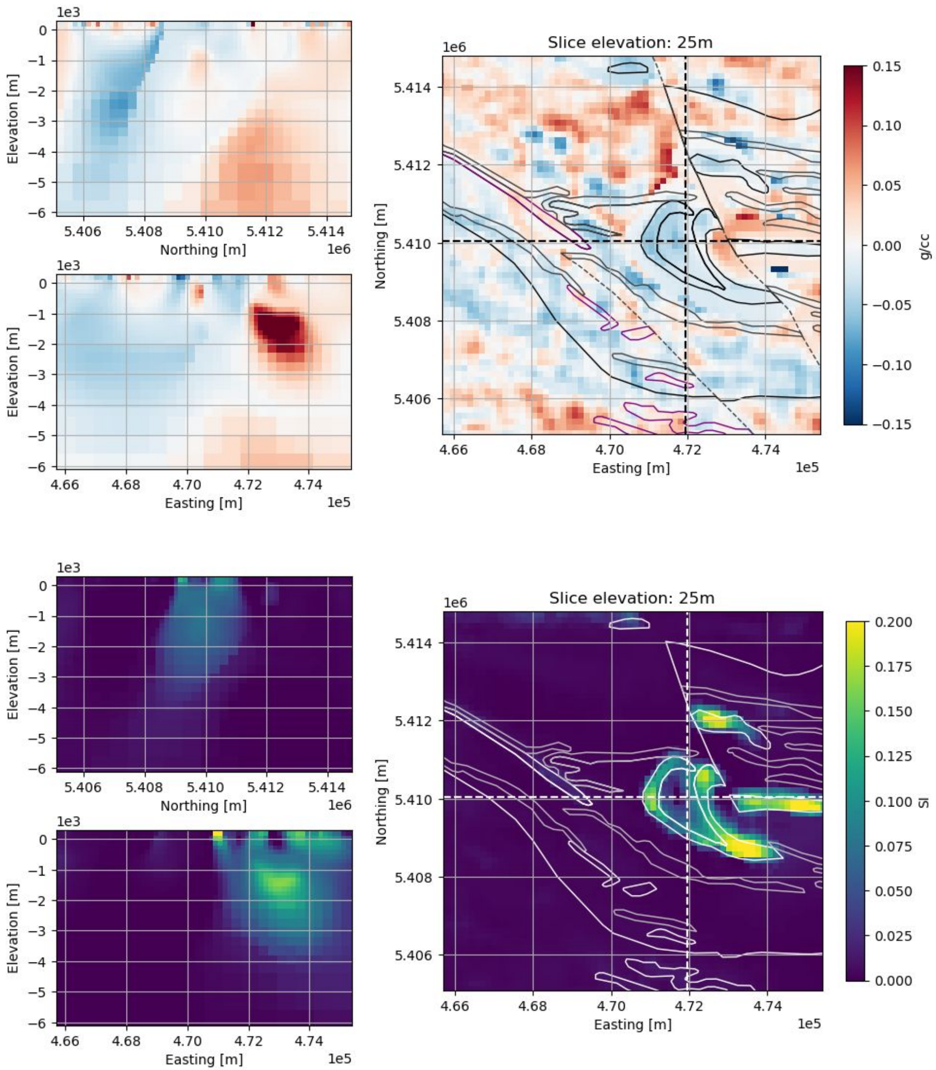




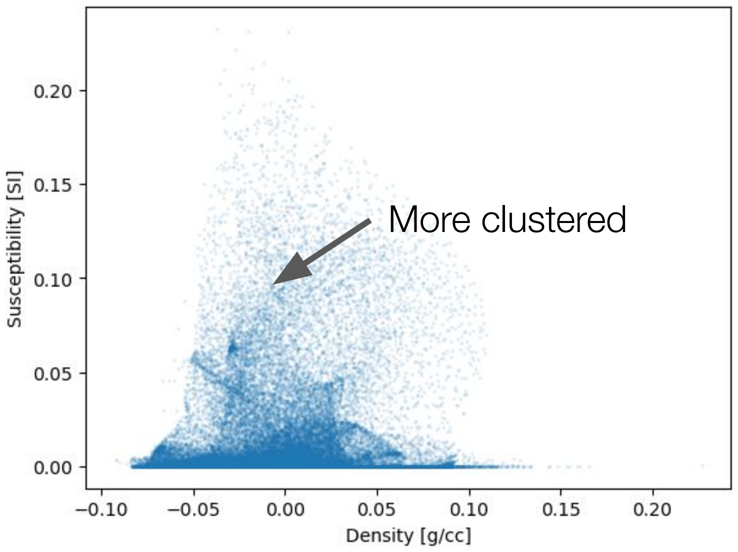


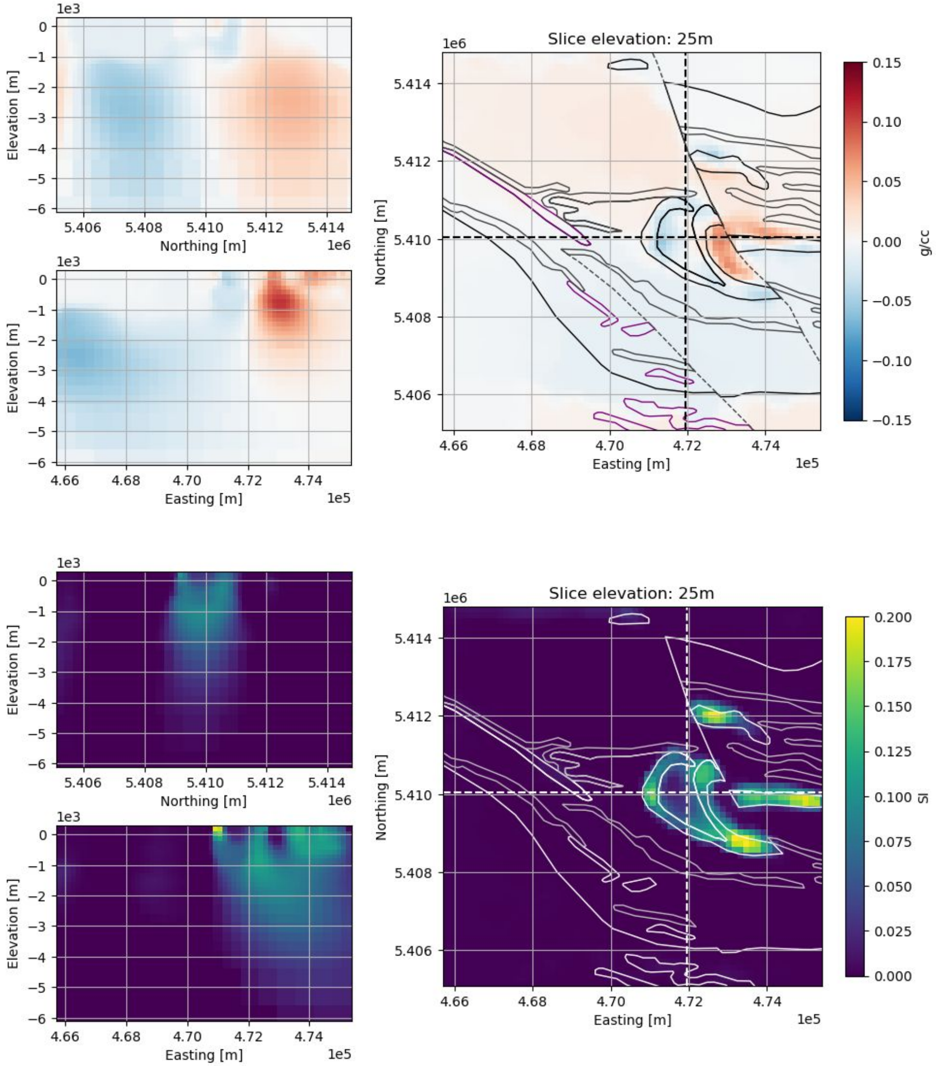
cross gradient



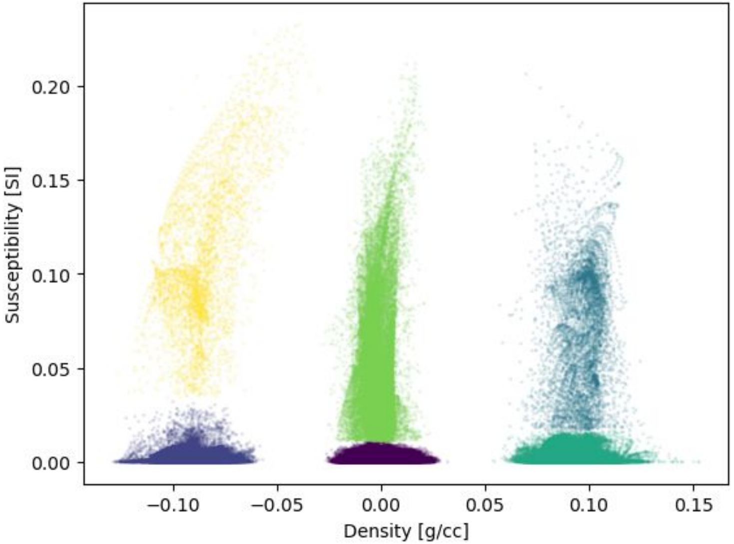


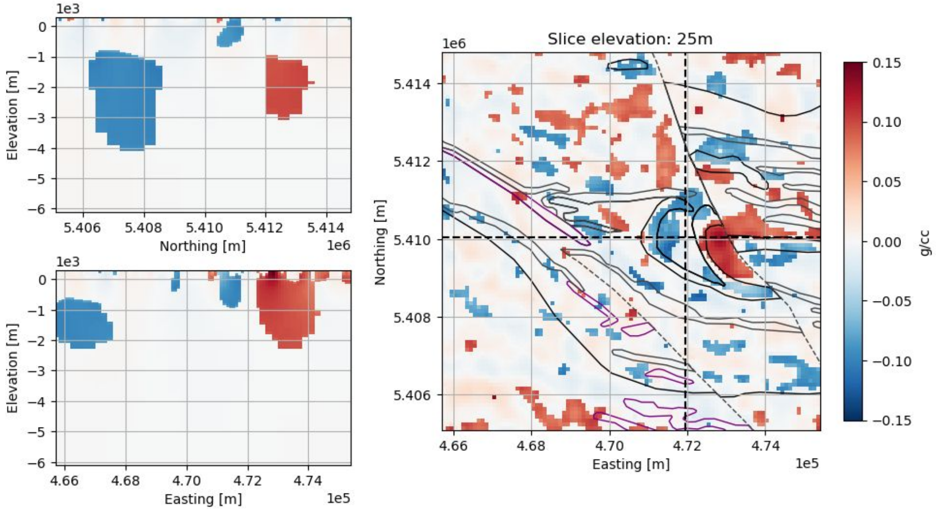
joint total variation

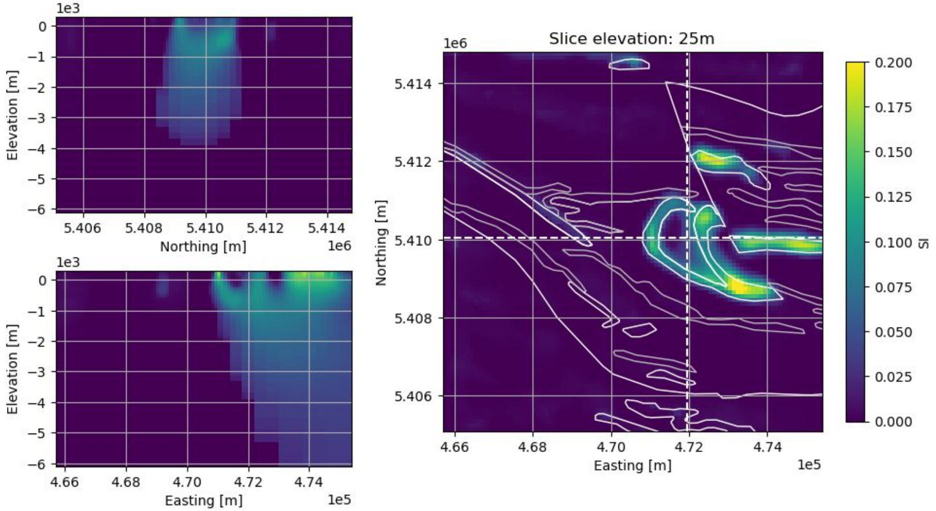




PGI


Susceptibility [SI]
Density [g/cc]


Elevation [m]
Northing [m]
Easting [m]
Slice elevation: 25m
g/cc


Elevation [m]
Northing [m]
Easting [m]
Slice elevation: 25m
SI

code comparison

Cross Gradient vs JTV

```
reg_mag = SimPEG.regularization.WeightedLeastSquares(
    mesh,
    active_cells=active_cells,
    mapping=wires.susceptibility,
    reference_model=reference_model,
)
```

In [14]:

```
reg_grav.set_weights(distance_weighting=wr_gravity)
reg_mag.set_weights(distance_weighting=wr_magnetic)
```

Cross Gradient

In [15]:

```
# Define the coupling term to connect two different physical property models
lamda = 2e10  # weight for coupling term
cross_grad = SimPEG.regularization.CrossGradient(mesh, wires, active_cells=active_cells)
```

In [16]:

```
# combo
dmis = 10 * dmis_grav + dmis_mag
reg = reg_grav + reg_mag + lamda * cross_grad
```

In [17]:

```
lower = np.r_[np.full(n_active, -0.8) , np.zeros(n_active) ]
upper = np.r_[np.full(n_active, 0.8) , np.full(n_active, np.infty)]

opt = SimPEG.optimization.ProjectedGNCG(
    maxIter=20,
    lower=lower,
    upper=upper,
    maxIterLS=15,
    maxIterCG=50,
    tolCG=1e-5,
    tolX=1e-3,
)
```

JTV

In [15]:

```
# Define the coupling term to connect two different physical property models
lamda = 1E-4  # weight for coupling term
jtv = SimPEG.regularization.JointTotalVariation(mesh, wires, eps=1E-16,
    active_cells=active_cells)
jtv.reference_model = reference_model

jtv.set_weights(sensitivity_weights=wr_gravity**2 + wr_magnetic**2)
```

In [35]:

```
# combo
dmis = 2*dmis_grav + dmis_mag
reg = reg_grav + reg_mag + lamda * jtv
```

In [40]:

```
lower = np.r_[np.full(n_active, -0.8) , np.zeros(n_active) ]
upper = np.r_[np.full(n_active, 0.8) , np.full(n_active, np.infty)]

opt = SimPEG.optimization.ProjectedGNCG(
    maxIter=20,
    lower=lower,
    upper=upper,
    maxIterLS=15,
    maxIterCG=50,
    tolCG=1e-5,
    tolX=1e-3,
)
```

code comparison

Cross Gradient
vs JTV

Roughly 5 lines
of different code

Cross Gradient

```
# Here we define the inverse problem that is to be
solved
inv_prob =
SimPEG.inverse_problem.BaseInvProblem(dmis, reg, opt)

starting_beta =
SimPEG.directives.PairedBetaEstimate_ByEig(beta0_rati
o=1E-2)

# Defining the fractional decrease in beta and the
number of Gauss-Newton solves
# for each beta value.
beta_schedule = SimPEG.directives.PairedBetaSchedule(
    cooling_factor=3, cooling_rate=1
)

# Options for outputting recovered models and
predicted data for each beta.
save_iteration =
SimPEG.directives.SimilarityMeasureSaveOutputEveryIte
ration()

joint_inv_dir =
SimPEG.directives.SimilarityMeasureInversionDirective
()

stopping =
SimPEG.directives.MovingAndMultiTargetStopping(tol=1e
-6)

# Updating the preconditionner if it is model
dependent.
update_jacobi =
```

JTV

```
# Here we define the inverse problem that is to be
solved
inv_prob =
SimPEG.inverse_problem.BaseInvProblem(dmis, reg, opt)

starting_beta =
SimPEG.directives.PairedBetaEstimate_ByEig(beta0_rati
o=1E0)

# Defining the fractional decrease in beta and the
number of Gauss-Newton solves
# for each beta value.
beta_schedule = SimPEG.directives.PairedBetaSchedule(
    cooling_factor=3, cooling_rate=1
)

# Options for outputting recovered models and
predicted data for each beta.
save_iteration =
SimPEG.directives.SimilarityMeasureSaveOutputEveryIte
ration()

joint_inv_dir =
SimPEG.directives.SimilarityMeasureInversionDirective
()

stopping =
SimPEG.directives.MovingAndMultiTargetStopping(tol=1e
-6)

# Updating the preconditionner if it is model
dependent.
update_jacobi =
```

Conclusions

- We have extended SimPEG to support multiple joint inversions
- Generalized the concept of joint inversions to form a framework
- Framework allows us to easily test different joint inversion methods
- Developed a good sense of how the three methods performed

Future Work

- Further iteration on directives
- More joint inversion methods

Acknowledgments

- Mitacs and Mira Geoscience
- Canada Nickel
- SimPEG contributors

thank you & questions

email: josephrcapriotti@gmail.com

slides: bit.ly/capriotti-image-2023