Open-source hydrogeophysical modeling and inversion with pyGIMLi 1.1

Recent advances and examples in research and education

Florian Wagner¹, Carsten Rücker²*, Thomas Günther³, Friedrich Dinsel², Nico Skibbe³, Maximilian Weigand⁴, and Joost Hase⁴

May 7, 2020

¹RWTH Aachen University, Applied Geophysics and Geothermal Energy, Aachen, Germany
²Berlin University of Technology, Department of Applied Geophysics, Berlin, Germany
³Leibniz Institute for Applied Geophysics (LIAG), Hannover, Germany
⁴University of Bonn, Institute of Geosciences, Section Geophysics, Bonn, Germany
*now at: Federal Office for the Safety of Nuclear Waste Management, BASE, Berlin, Germany


www.pygimli.org
Computational hydrogeophysics is currently not reproducible

- In hydrogeophysics, researchers gain quantitative information on the subsurface by studying the dynamic process of interest together with its geophysical response.
- This requires coupling of different numerical models → obstacle for many practitioners and students.
- Even technically versatile users tend to build individually tailored solutions by coupling different (and often proprietary) forward simulators.
- The lack of reproducibility represents an impediment for the advancement of hydrogeophysics.
pyGIMLi: An open-source library for modelling and inversion in geophysics

- management of **structured and unstructured meshes** in 2D & 3D
- computationally efficient **finite-element and finite-volume solvers**
- **various geophysical forward operators**: ERT/IP, Traveltime, Gravimetry, SP
- Gauss-Newton based frameworks for **constrained, joint and process-based inversions** with region-specific regularization
- offers opportunities for **hydrogeophysical modeling and inversion**
- **open-source, platform compatible**, documented & tested code
- well suited for **teaching & reproducible research**
- 1.0 version published in 2017 in *Computers and Geosciences* [4] (and among the Most Downloaded & Most Cited papers)
pyGIMLi 1.1: So what’s new?

- ERT Manager with full functionality and improved accuracy of traveltime calculations [1]
- Anisotropic parameters
- Simulation of electrical streaming potentials
- Complex-valued electrical forward modeling
- Geostatistical regularization operators for unstructured meshes [2]
- Petrophysical joint inversion [6]
- Built-in 3D visualization leveraging upon PyVista [5]
- One-line installation on Windows, Mac & Linux (www.pygimli.org/installation.html)
- Improved website & documentation: New examples (e.g., on hydrogeophysical modeling and IP forward modeling), better API documentation, modern CSS framework, and a searchable database with 40+ peer-reviewed pyGIMLi-based publications: www.pygimli.org/publist.html
Support for anisotropic parameters

**Theory**

Finite-element solver allows for more flexible parameter $a$ in:

$$\nabla \cdot (a \nabla \phi) = f$$

$a$ can be float, complex or an anisotropy or constitutive matrix.

```python
import pygimli as pg
mesh = pg.meshtools.createGrid(10, 10)
a = pg.solver.anisotropyMatrix(
    lon=1.0, trans=10.0, theta=45/180 * np.pi)
u = pg.solve(mesh, a=a, f=f,
    bc={'Dirichlet': {'*': 0}})
```

**Example in finite-element tutorial**
Example: Simulating electrical streaming potentials

**Theory**

\[ \nabla \cdot (\sigma \nabla \varphi) = \nabla \cdot \left( \frac{Q_v}{S_w} \mathbf{v} \right) \text{ with } \mathbf{v} = -K \nabla h \]

Divergence of water flow acts as electric current source if pore water is in contact with electrically charged interfaces

```python
from pygimli.solver import solve, grad

# Flow problem
h = solve(mesh, a=K, uB=bc) # hydraulic head (m)
v = -K * grad(mesh, h) # Darcy velocity (m/s)
pg.show(mesh, v)
```

Divergence of water acts as electric current source if pore water is in contact with electrically charged interfaces.
Example: Simulating electrical streaming potentials

Theory

$$\nabla \cdot (\sigma \nabla \phi) = \nabla \cdot \left( \frac{Q_v}{S_w} \mathbf{v} \right) \text{ with } \mathbf{v} = -K \nabla h$$

Divergence of water flow acts as electric current source if pore water is in contact with electrically charged interfaces

```python
from pygimli.solver import solve, div

# Electrical problem
sources = div(mesh, coupling_coefficient * v)
SP = solve(mesh, a=sigma, f=sources, uB=ertbc)
pg.show(mesh, SP)
```
Towards complex inversion of spectral induced polarization data

- Complex-valued electrical forward modeling and sensitivity calculation is supported.
- Modeled transfer impedances and complex sensitivity values in good agreement to CRTomo [3].
- Inversion not yet fully integrated in new frameworks, but proof-of-concept complex inversion available.

**Complex-valued modeling example**

**Preliminary inversion example**
Geostatistical regularization operators (Jordi et al., 2018) [2]

**Key points**
- Regularization becomes mesh-independent
- Larger footprint reduces dependence on survey geometry
- Can be used as the basis for structural coupling

\[
\mathbf{C}_{M,ij} = \sigma^2 \exp\left(-3\sqrt{\left(\frac{H_{x,ij}}{l_x}\right)^2 + \left(\frac{H_{y,ij}}{l_y}\right)^2 + \left(\frac{H_{z,ij}}{l_z}\right)^2}\right)
\]

\[
\begin{pmatrix}
H'_{x} \\
H'_{z}
\end{pmatrix} =
\begin{pmatrix}
\cos \theta & \sin \theta \\
-\sin \theta & \cos \theta
\end{pmatrix}
\begin{pmatrix}
H_{x} \\
H_{z}
\end{pmatrix}
\]
Petrophysical joint inversion

- Different Method Managers (e.g., ERT and TravelTime) can be combined to estimate petrophysical quantities
- Arbitrary petrophysical models can be used
- Sensitivities are scaled with regard to the petrophysical models
- Allows to incorporate constraints on the petrophysical target parameters (i.e., include soil moisture measurements in the inversion)
- For details see examples linked to the right and Wagner et. al, GJI, 2019 [6]

Joint inversion for water, ice, and air

- Electrical resistivity data
- Seismic refraction data
- Petrophysical joint inversion
- Petrophysical constraints
- Porosity
- Water content
- Ice content
- Air content

Petrophysical Joint inversion for water saturation
Petrophysical Joint inversion for four phases
Built-in pyVista-based 3D viewer

Features

• Supports arbitrarily complex 3D meshes
• Intuitive GUI to adjust colors and perform slicing
• VTK export (e.g., for ParaView)
• also works in Jupyter Notebooks

Example

```python
import pygimli as pg
n = 20
mesh = pg.meshtools.createGrid(n, n, n)
data = pg.x(mesh.cellCenters())
pg.show(mesh, data)
```
Teaching & Outreach

- pyGIMLi is well suited for teaching in conjunction with Jupyter Notebooks (www.jupyter.org)
- Different abstraction levels (equation vs. application level) allow for tailored exercises at Bachelor and Master level
- Used in classes at several (inter)national universities
- Centralized installations (JupyterHub) allow large numbers of students to participate with a web browser only
- Was used to interactively illustrate electrical imaging of trees at “Highlights der Physik” in Bonn in September 2019
pyGIMLi is open-source and welcomes new contributors

Ways to contribute to the project

1. If you used pyGIMLi for your work, add yourself to this table.
2. If you experience issues or miss a certain feature, please open a new issue on GitHub.
3. Send an interesting usage example to mail@pygimli.org
4. Contribute to the code as explained here.

B. Giroux and B. Larouche. 
Task-parallel implementation of 3D shortest path raytracing for geophysical applications. 

Geostatistical regularization operators for geophysical inverse problems on irregular meshes. 

A. Kemna.  

C. Rücker, T. Günther, and F. Wagner.  

PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK). 

Quantitative imaging of water, ice, and air in permafrost systems through petrophysical joint inversion of seismic refraction and electrical resistivity data.  