pycycle

Teaching code for New Directions Seminar. Implements a simple Boundary Element Method for SEAS modelling.

Dependencies

- Python 3
- NumPy
- SciPy
- matplotlib
- Jupyter notebooks

Exercises

Your goal is to implement missing functionality such that the unit tests pass.

In order to run the unit tests, open a terminal and run the following command from the directory in which the README file resides:

```
python3 -m unittest
```

You may also run individual unit tests:

```
python3 -m unittest test.test_bem
python3 -m unittest test.test_mesh
python3 -m unittest test.test_seas
```

1. Implement mesh module

Implement the following functions:

```
mesh.LineElement.xi
mesh.LineElement.basis
mesh.LineElement.factor
mesh.InfiniteLineElement.xi
mesh.InfiniteLineElement.basis
mesh.InfiniteLineElement.factor
```

Hints for InfiniteLineElement:

1. The map $\xi(\theta)$, with $\theta \in [-1, 1]$ has the form $a\frac{p(\theta)}{q(\theta)}$, where p and q are linear functions of θ.
2. The basis function is $\frac{|a|^2}{|\xi|^2}$, as discussed in the lecture.

2. Implement bem.assemble

The function bem.assemble returns the A matrix.

Hints:

1. Do not make any assumptions about the mesh, i.e. LineElements and InfiniteLineElements may come in any order.
2. Mind the weak singularities.

3. Implement seas.Context.slip_rate

The function seas.Context.slip_rate computes the slip rate by finding the root of

$$C(V) := \tau + f(V, \psi) + \eta V = 0$$

Hints:

1. f is already implemented in the Context class.
2. Prove that the only root of $C(V)$ lies in the interval $[0, -\tau/\eta]$ if $\tau < 0$, $[-\tau/\eta, 0]$ if $\tau > 0$, and is equal to 0 if $\tau = 0$.
3. scipy.optimize.toms748

4. Implement seas.Context.traction

The function seas.Context.traction computes

$$\tau = \tau_0 + \mu \mathcal{M}^T A^{-1} b \left(\mathcal{M} \frac{1}{2}(S - 1 V_p t) \right)$$

Hints:

1. N is the number of elements in the mesh, Nf is the number of on-fault elements.
2. Use self.map to implement the action of $\mathcal{M}$.
3. Implement $b(g)$ with `b = self.B @ g`.
4. scipy.linalg.lu_solve
5. Use self.imap to implement the action of $\mathcal{M}^T$.

5. Run BP1

When all unit tests pass you may run

```
jupyter notebook
```

and open the bp1 notebook. The notebook simulates the SEAS benchmark problem 1 for about 250 years and interactively updates maximum slip-rate and displacement along the fault.

Takes only a few hours.

Feel free to play around with the parameters.