Efficient Filtering and Fitting of Models Derived from Integro-Difference Equations

Evan Tate Paterson Hughes

Table of contents

Chapter 1

Fitting IDEM using `jax_idem`

The primary use of the jax_idem package is to fit Integro-difference equation models to data.

Currently, the only supported way to do this is through maximum-likelihood estimation with the kalman/information filter and OPTAX.

1.1 Simple example; synthetic simple data

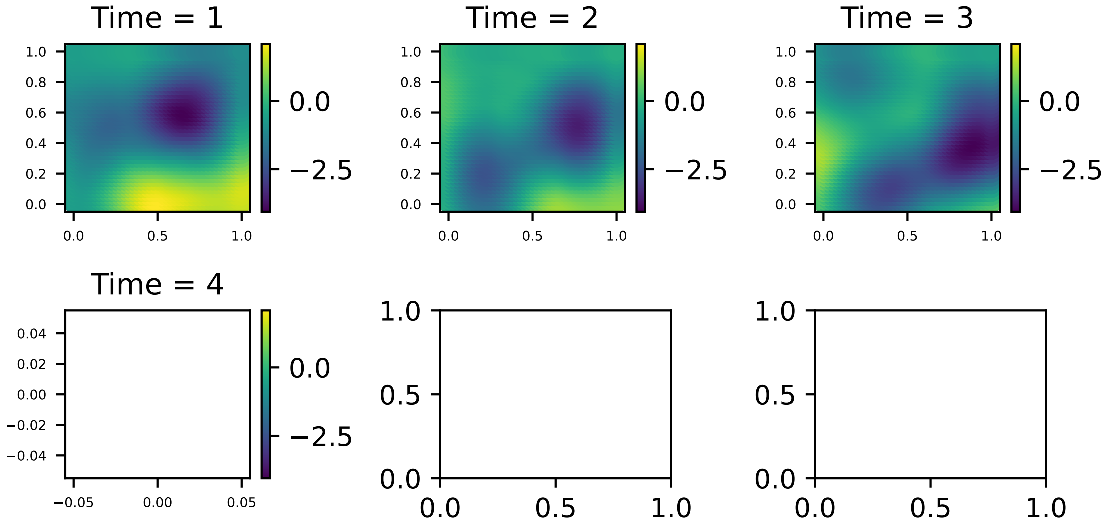
We will start by simulating from a simple IDEM with only three time steps. We can quickly make a model using `gen_example_idem`;

```python
import sys
import os
sys.path.append(os.path.abspath('../src/jax_idem'))

import jax.random as rand
import jax.numpy as jnp
import jax
import matplotlib.pyplot as plt
import filter_smoother_functions as fsf
import plotly.express as px
import plotly.io as pio
import plotly.graph_objs as go
import pandas as pd

import utilities
import IDEM as idem
import filter_smoother_functions as fsf
```

```python
key = jax.random.PRNGKey(1)
keys = rand.split(key, 3)

process_basis = utilities.place_basis(nres=1, min_knot_num=5)
nbasis = process_basis.nbasis

m_0 = jnp.zeros(nbasis).at[7].set(1)
```

```
sigma2_0 = 0.01

truemodel = idem.gen_example_idem(
    keys[0], k_spat_inv=True,
    process_basis=process_basis,
    m_0=m_0, sigma2_0=sigma2_0
)

# Simulation
T = 3

process_data, obs_data = truemodel.simulate(nobs=50, T=T + 1, key=keys[1])

process_data.show_plot()
```

```
/home/tate/MyProjects/JAX-IDEM/src/jax_idem/utilities.py:393: UserWarning:

Matplotlib is currently using module://matplotlib_inline.backend_inline, which is a non-GUI backend, so ca
```


Time = 1
Time = 2
Time = 3
Time = 4