Module `bstpp.main`

Classes

```
class Point_Process_Model (data, A, model='cox_hawkes', spatial_cov=None, cov_names=None, cov_grid_size=None,
                           **priors)
```

Spatiotemporal Point Process Model.

Parameters

data : `str` or `pd.DataFrame`

either file path or DataFrame containing spatiotemporal data. Columns must include 'X', 'Y', 'T'.

A : `np.array [2x2]`

Spatial region of interest. First row is the x-range, second row is y-range.

model : `str`

one of ['cox_hawkes','lgcp','hawkes'].

spatial_cov : `str,pd.DataFrame,gpd.GeoDataFrame`

Either file path (.csv or .shp), DataFrame, or GeoDataFrame containing spatial covariates. Spatial covariates must cover all the points in data. If spatial_cov is a csv or pd.DataFrame, the first 2 columns must be 'X', 'Y' and cov_grid_size must be specified.

cov_names : `list`

List of covariate names. Must all be columns in spatial_cov.

cov_grid_size : `list-like`

Spatial covariate grid (width, height).

priors : `dict`

priors for parameters (a_0,w,alpha,beta,sigmax_2). Must be a numpyro distribution.

Methods

```
def cov_weight_post_summary(self, plot_file=None, summary_file=None)
```

Plot posteriors of weights and bias and save summary of posteriors.

Parameters

plot_file : `str`

Path in which to save plot.

summary_file : `str`

Path in which to save summary

Returns

`pd.DataFrame`

summary of weights and bias

```
def plot_spatial_background(self, output_file=None, include_cov=False)
```

Plot mean posterior spatial background with/without covariates

Parameters

output_file : `str`

Path in which to save plot.

include_cov : `bool`

Include effects of spatial covariates.

```
def plot_temporal_background(self, output_file=None)
```

Plot mean posterior temporal gaussian process.

Parameters

plot_file : `str`

Path in which to save plot.

```
def plot_trigger_posterior(self, output_file=None)
```

Plot histograms of posterior trigger parameters.

Parameters

output_file : `str`

Path in which to save plot.

Returns

`pd.DataFrame`

Summary of trigger parameters.

```
def plot_trigger_time_decay(self, output_file=None, t_units='days')
```

Plot temporal trigger kernel sample posterior.

Parameters

output_file : `str`

Path in which to save plot.

t_units : `str`

Time units of original data.

```
def run_mcmc(self, batch_size=1, num_warmup=500, num_samples=1000, num_chains=1, thinning=1,
            output_file=None)
```

Run MCMC posterior sampling on model.

Parameters

batch_size : `int`

See numpyro documentation for description

num_warmup : `int`

num_samples : `int`

num_chains : `int`

thinning : `int`

output_file : `str`

File to save output to.