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 given by,

$$\lambda(t,s) = \mu(s,t) + \sum_{i:t_i<t} \alpha f(t-t_i;\beta)\varphi(s-s_i;\sigma^2)$$

where f is the exponential pdf, φ is the gaussian pdf, and μ is given by

$$\mu(s,t) = exp(a_0 + X(s)w + f_s(s) + f_t(t))$$

where $X(s)$ is the spatial covariate matrix, f_s and f_t are Gaussian Processes. Both f_s and f_t are simulated by a pretrained VAE. We used a squared exponential kernel with hyperparameters $l \sim InverseGamma(10,1)$ and $\sigma^2 \sim LogNormal(3,0.5)$

Parameters

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

`A` : `np.array [2x2], GeoDataFram`
: Spatial region of interest. If np.array 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, **kwargs)`

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.

kwargs : `dict`

Plotting parameters for geopandas plot.

`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.

```
def run_svi(self, num_samples=1000, output_file=None, resume=False, **kwargs)
```

Perform Stochastic Variational Inference on the model. Parameters

---

num_samples : `int` , default= `1000`
: Number of samples to generate after SVI.

output_file : `string` , default= `None`
: File name to save results.

resume : `bool` , default= `False`
: Pick up where last SVI run was left off. Can only be true if model has previous run_svi call.

lr : `float` , default= `0.001`
: learning rate for SVI

num_steps : `int` , default= `10000`

Number of interations for SVI to run.

auto_guide : `numpyro AutoGuide` , default= `AutoMultivariateNormal`

See numpyro AutoGuides for details.

init_strategy : `function` , default= `init_to_median`

See numpyro init strategy documentation