Code for seasonal marked point processes analysis with example of US hurricane data

Sai Xiao

2014/09/25

This readme file describes the details of the code, inputs, outputs and post-processing of the DDP-PBAR model and DDP-AR model with US hurricane data. It helps to replicate the analysis presented in the paper "Modeling for Seasonal Marked Point Processes: An Analysis of Evolving Hurricane Occurrences" by Xiao, Kottas and Sansó.

- Software needed: gcc complier, R
- C++ package needed: GNU Scientific Library
- R packages needed: MCMCpack, MASS

A quick start

Once you have downloaded the zipped file, unzip it and get the directory "hurricane".

All the codes are in the folder. Some major files include:

`preProcess.R`: To pre-process the original data set.

`ddpPBARSampler.cc`: The main function implementing the MCMC sampler of time-varying density function in the DDP-PBAR model

`ddpARSampler.cc`: The main function implementing the MCMC sampler of DDP-AR model for the marked point process.

`timeInference.cc`: To get various inference on time from DDP-PBAR model.

`markInference.cc`: To get various inferences on marks from DDP-AR model.

`gamma.R`: To obtain the posterior samples of normalizing constants, $\gamma_1, \ldots, \gamma_K$.

`postProcess.R`: All the inference results are written into files. Use this R code to draw plots from the inference output.

In the **data** folder:

`stormData2013.csv`: the original data set including US hurricane from 1900 to 2013.

`public_data_may_2007.xls`: the inflation, wealth per capita and affected county population size.

After data processing and transformation, the following data files are generated as input data sets for our model.

y_correction.txt: the time points of 239 hurricane in the US, scaled to be between (0, 1).

ten_year.txt: the number of hurricanes in 11 decades

winds_log_norm.txt: the log of maximum wind speed, minus the mean

damage_std_log_norm.txt: the log of standardized damage, minus the mean. If the values equals to 100, that means it is missing.

In the **example_inference** folder: All the output files needed to produce the plots in the paper in section 4. All these files can be reproduced by following the guideline in next section.

How to use the code

The DDP-PABAR model and DDP-AR model decompose the intensity function to two parts. One is normalized density function, which is modeled by dependent Dirichlet process mixture model. The other is the normalizing constant γ, which is modeled by a time series model.

We need to generate posterior samples from two models and then draw various inferences of our interest by using these posterior samples. Here is the guideline to use the code.

1. Compile. To sure that GNU Scientific Library (GSL) and GCC complier are installed. Simply run

   ```
   make
   ```

   to generate execution file: ddpPBARSampler, ddpARampler, timeInference and markInference.

2. Run the MCMC sampler. The MCMC sampler ddpPBARSampler implements the DDP-PBAR model for normalized three-dimensional density function in Section 3 of the paper. If time-varying intensities are only what you are interested, ddpPBARSampler is enough.

   The MCMC sampler ddpARSampler implements the DDP-AR model for normalized three-dimensional density function in Section 4 of the paper.

   ```
   ./ddpPBARampler configuration_filename
   ./ddpARampler configuration_filename

   Example: ./ddpPBARampler PBAR\_conf1.txt
            ./ddpARampler conf15.txt
   ```

   In the configuration file, several arguments are defined, e.g. the input file, MCMC iteration number, etc. Two sample configuration files are included in **data** folder. PBAR_conf1.txt is for ddpPBARSampler and conf15.txt is for ddpARSampler. They share some arguments but ddpARSampler need more arguments.

   Arguments are all included in the configuration file:

   M: The number of MCMC iterations after the burn-in period.

   B: The number of burn-in iterations.

suf: The suffix for the output files.

NN: The number of observations.

KK: The number of decades.

JJ: The truncation level in the stick-breaking definition of DP prior.

NUM_GRID: The number of grid points in the time line.

GAP: Only one iteration of samples is saved for every "GAP" number of iterations.

a_alpha, b_alpha: They specify the prior for precision parameter alpha in the DP prior. $\alpha \sim$ `Gamma(a_alpha, b_alpha)`.

a_rho, b_rho: They specify the prior for correlation parameter rho in the PBAR process. $\rho \sim$ `Beta(a_rho, b_rho)`.

a_sigma2, b_sigma2: They specify the prior for the variance parameter in the Log-normal kernel for maximum wind speed. See equation (4) in the paper. $\sigma^2 \sim$ `InvGamma(a_sigma2, b_sigma2)`.

a_sigma22, b_sigma22: They specify the prior for the variance parameter in the Log-normal kernel for standardized damage. See equation (4) in the paper. $\xi^2 \sim$ `InvGamma(a_sigma22, b_sigma22)`.

a_epsilon, b_epsilon: They specify the prior for the variance parameter in the AR1 process, see equation on page 21. $\sigma_1^2 \sim$ `InvGamma(a_epsilon, b_epsilon)`.

a_epsilon2, b_epsilon2: They specify the prior for the variance parameter in the AR1 process, see equation on page 21. $\sigma_2^2 \sim$ `InvGamma(a_epsilon, b_epsilon)`.

pdsd_rho: The standard deviation of normal proposal distribution for logit(ρ).

pdsd_v: The standard deviation of normal proposal distribution for logit(v).

theta_initial: The initial value of all atoms $\{\theta_{j,k}\}$, $j = 1, \ldots, J$ and $k = 1, \ldots, K$.

v_initial: The initial value of latent variable $\{v_{j,k}\}$, $j = 1, \ldots, J$ and $k = 1, \ldots, K-1$.

alpha_initial: The initial value of α.

rho_initial: The initial value of ρ.

sigma2_initial: The initial value of σ^2.

sigma22_initial: The initial value of ξ^2.

beta_initial: The initial value of β.

eta_initial: The initial value of η.

epsilon1_initial: The initial value of σ_1^2.

epsilon2_initial: The initial value of σ_2^2.

timefile: the data set including all time points. In the sample configuration file, we use ./data/y_correction.txt

countfile: the data set including counts. In the sample configuration file, we use ./data/ten_year.txt

windfile: the data set including the second mark. In the sample configuration file, we use ./data/winds_log_norm.txt

damagefile: the data set including the third mark. In the sample configuration file, we use ./data/damage_std_log_norm.txt

Output: posterior samples of all parameters are yielded in the **data** folder by default. Details of some outputted files are as follows.

There are two outputted file including `M` rows/iterations of samples and have tab-separated columns. The column information are as follows.

`intensity_output_suf.txt`: τ, ρ, α, n^*.

`mark_output_suf.txt`: $\alpha, n^*, \sigma^2, \beta, \sigma_1^2, \xi^2, \phi, \sigma_2^2$.

In addition,

`parameter1_suf.txt` contains all sampled atoms $\{\theta_{j,k}\}, \{\nu_{j,k}\}$ and $\{\eta_{j,k}\}$, $j = 1, \ldots, J$ and $k = 1, \ldots, K$ in M iterations.

`parameter2_suf.txt` contains all sampled weights $\{w_j\}$, $j = 1, \ldots, J$ in M iterations.

`intensity_acceptance_suf.txt`: the acceptance rate of ρ, $\{\theta_{j,k}\}$ and latent variables $\{v_{j,k}\}$.

3. Generate posterior samples for $\gamma_1, \ldots, \gamma_K$. In `gamma.R`, we specify the value for number of samples to be 10000. It can be changed as needed.

```
source("gamma.R")
```

By default, M runs of posterior samples of $\gamma_1, \ldots, \gamma_K$ are generated and written to the file `./data/gammaSamples.txt`.

To plot the 95% interval and mean of $\gamma_1, \ldots, \gamma_K$, simply run

```
plot.gamma()
```

4. Make inference results in Section 3. Make sure that from step 2, the outputted files are generated in the **data** folder.

Run the following command :

```
./timeInference configuration_name function#

Example: ./timeInference PBAR_conf1 1
```

This example command is to use output results suffixed with "PBAR_conf1" to generate density function across all decades. For details of functions, see comments in the code.

Output:

`mixture_interval_list_suf.txt`: for function 1, posterior predictive density across all decades. Use `density.plot()` to visualize the results.

5. Make inference results in Section 4.

Make sure that from step 2, the outputted files are generated in the directory.

Run the following command to get various inferences:

```
./markInference suf function# month1 month2 [argv5]

Example: ./markInference conf15 1 8 11
```

This example command is to use output results suffixed with "conf15", to get the conditional density of maximum wind speed given ASO (from Aug to Oct).

Arguments are explained as follows,

`suf`: the suffix to specify which output files to be used.

`function#`: from 1 to 8. There are 8 functions implemented in the markInference.cc. Specify which function to use. Check the comments in the makeInference.cc for details of these eight functions.

`month1 and month2`: the values range from 4 to 13. 4 means the beginning of the April and likewise 5 means the beginning of May and so on. 13 means the end of a year. "month1" specifies the starting month of a period. "month2" specifies the last month of a period.

`argv5`: optional arguments. It might be needed for some functions. Check the comments in the makeInference.cc for setting of arguments.

Output:

For each function, inference results are output into different files. For details, see comments in `markInference.cc`. Post process the inference results to draw plots and get summary.

6. Post procession

   Post-process the inference output, such as drawing plots or do other summary of parameter estimation in R.

   To get plots of each function, corresponding R code is written. Load the code in R, run

   ```
   source("postProcess.R")
   ```

   Most of inference results presented in the paper are stored in **example_inference** folder.

   To produce all the plots in Section 4 of the paper, run

   ```
   example1.plot(), ..., example8.plot()
   ```

   To produce all plots in Section 3 of the paper, run

   ```
   density.plot()
   ```

   For other details of functions in `postProcess.R`, see comments in the R file.