Paper To-Do List

Waleed Alhaddad

19/11/2019

1 To do

1. Read, understand and reference Japan literature (3 Days)
2. Add references through out the paper and learn how to use bibilex (1 hr)
3. Copy subject calssification and keywords from Somaya's paper. Then discuss the choice and improve it with Raul and Ahmad. (30 m)
4. Add labels to the paper sections. We do this because it is easier to refer to sections after they have an associated label.
5. Add two simulation function with zero diffusion, where one is with derivative tracking and the other without it. Select an oscilattory forecast or synthesize one to test on it and plot bth plots to show the phase error when we don't have derivative tracking. (30 m)
6. Rename section 3 to: state Independent Diffusion; Lamperti transform
7. Number and emphasize the Lamperti transform excplicitly.
8. Change title of section 4 to: Likelihood in V space.
9. Fockker-Plank equation
   - Change variable in Fockker-Plank from y to $V_{t_{i,j}}$
   - define the functions a and b in the statement
   - clearly state the time interval that it is between $t_{i,j}$ and $t_{j,i+1}$
   - State initial condition to be a Dirac function at $V_{j,i}$
   - make a connection between the fokker plank and the likelihood transitions, i.e. $\rho(V_{j,i+1}|V_{j,i}) = f(V_{j,i+1}, t_{j,i+1}|V_{j,i}, t_{j,i})$
10. check dimensions of θ and if it looks reasonable with the average number of crossings (~ 5) (0.5 Day)
11. relate the θ to the time correlation structure and see how big of a gap should be left between two consecutive best samples. Then cut accordingly if possible. Check the ratio of time correlation to the length of the path if its much less than one as we would like. (2 Days)

12. Import data (1 Weeks)

    - import the data again
    - Detect when the forecast has started or read it as indicated in the data set.
    - arrange the forecast in chronological order and know which are consecutive forecasts
    - check if the forecast has no missing or corrupt data points
    - check the difference of time between the start times of the forecasts and record that information.
    - adapt the number of samples to be variable in every path. Then adjust all the code and solvers for this added flexibility.
    - Check which segments are the most recent for the time in question , i.e. choosing the path of best available forecast.

13. Add discussion about Lamperti and linearization of the non-linear drif that arises. Mention that the best proxy is Gaussian and approximation is good enough as the time between samples is small.

14. obtain an estimate of the product $\theta_0\alpha$ using quadratic variations and implement a function that computes that. Justify assuming that θ_t is independent of time here is somehow acceptable otherwise it is not possible to estimate $\theta_t\alpha$ using the quadratic variations.write a funciton that checks the number of crossing in the processed data set in order to seperate the product of $\alpha\theta_0$. (0.25 Day)

15. Implement Model 0 and adjust the rest of the code and plotting to this. (3 Days)

16. Remove Model 1 and write a paragraph about it because when $p = 1$ or $p = 0$ it is not physical to have zero diffusion. (0.2 Days)

17. Import, clean, filter and process the french data as done in step (12). Then run the best model in standard and in Lamperti space (1 week)

18. Fix the mini-batching in the optimization as it resets the samples in each evaluation. Should store the sample for next evaluations within one optimization run. (3 Days)

19. Produce documentation for all functions, scripts and objects. (1.5 weeks)

2 Code correction report:

1. The data is wrong. There are many repeated paths. After correcting most of the code, it is needed to download all again and do the processing again.

2. We have to be careful about that is (θ, α) in the Likelihood. When we optimize, we need to optimize over (θ_0, α) so we would have

$$\mathcal{L} = \mathcal{L}\left(\theta_0, \alpha, \{V\}, \{p\}\right).$$

   Notice that $\theta = \theta(\theta_0, \dot{p})$.

3 Pseudocode:

Algorithm 1: Main Algorithm

```
* load paths information into this_model *;
while current_batch_size <= max_batch_size do
    this_model.optimize(intial_point, current_batch_size, ...);
    current_batch_size = current_batch_size * batch_multiplier;
end
** Methods from this_model object **
Method this_model.optimize(intial_point, current_batch_size, ...):
    likelihood = self.rand_beta_objective;
    batch = self.gen_mini_batch(batch_size);
    min_param = scipy.optimize.minimize(fun = likelihood, x0 = param_initial, args =
     (batch_size, batch), method = "Nelder-Mead");
    Returns = min_param
return
Method this_model.gen_mini_batch(current_batch_size, ...):
    self.{X,p,N,M};
    * create ṗ *;
    * create linked-lists with V,p, and ṗ *;
    * create 2D array with the linked-lists (call it combined_iter) *;
    batch = random_combination(combined_iter, batch_size);
    Returns = batch
return
Method this_model.rand_beta_objective(current_batch_size, ...):
    for j in range(batch_size) do
        * find moments using beta_moment *;
        L_n = L_n + (...);
    end
    Returns = -L_n
return
```