Extend Variational Inference Methods in PyMC3

Abstract

Variational inference is a great approach for doing really complex, often intractable Bayesian inference in approximate form. Common methods (e.g. ADVI) lack from complexity so that approximate posterior does not reveal the true nature of underlying problem. In some applications it can yield unreliable decisions.

Recently on NIPS 2017 OPVI framework was presented. It generalizes variational inverence so that the problem is build with blocks. The first and essential block is Model itself. Second is Approximation, in some cases $logQ(D)$ is not really needed. Necessity depends on the third and forth part of that black box, Operator and Test Function respectively.

Operator is like an approach we use, it constructs loss from given Model, Approximation and Test Function. The last one is not needed if we minimize KL Divergence from Q to posterior. As a drawback we need to compute $loqQ(D)$. Sometimes approximation family is intractable and $loqQ(D)$ is not available, here comes LS(Langevin Stein) Operator with a set of test functions.

Test Function has more unintuitive meaning. It is usually used with LS operator and represents all we want from our approximate distribution. For any given vector based function of z LS operator yields zero mean function under posterior. $loqQ(D)$ is no more needed. That opens a door to rich approximation families as neural networks.

Not only ADVI and Langevin Stein Operator VI are applicable with OPVI framework. Normalizing, Householder Flows fit well for it.

Motivation

My recent contributions (Implementing OPVI) to PyMC3 created a good basis for extending variational inference in PyMC3 even further. I tried to transfer theoretical framework to python code and it succeed. Now main logic is in base classes and all rotines are abstracted and use public interface that is provided by developer. Implementing state-of-the-art methods is now not a challenge, you should just break the problem into 4 blocks described above and implement abstract methods.

I also have a side project Gelato for using PyMC3 in neural networks. So my future plans are the following:

1. Implement Normalizing Flows
2. Implement Householders Flows
3. Implement Langein Stein Operator

4. Integrate OPVI to Gelato

Technical Details

I'm going to use the following libraries:

- **Theano**
- **PyMC3**
- **Gelato**
- **Lasagne**
- **NumPy**

As support material I'll use papers from arXiv:

Variational Inference with Normalizing Flows Improving Variational Auto-Encoders using Householder Flow Operator Variational Inference

Schedule of Deliverables

May 1th - May 28th, Community Bonding Period

Integrate OPVI to Gelato

Work on documentation for OPVI and Histogram. Add a noteboook with comprehensive example using Gelato.

May 29th - June 3rd

Implement Normalizing and Householder Flows

They have similar interface so implementing them both at once is the best decision

June 5th - June 9th

Debug and documentation period for Flows

June 12th - June 16th

Make sure all works fine. This period I leave for unexpected problems caused by my exams

June 19th - June 23th, End of Phase 1

Prepare PR to PyMC3 with implemented Flows

June 26 - June 30th, Begin of Phase 2

Implement Langevin Stein Operator This week I devote to LS Operator only

July 3rd - July 14th

Debug period. It can be really hard. LS Op uses Neural Network Test Functions, here I need much more time for making things work.

July 17th - July 21th, End of Phase 2

Extra time for unexpected problems with convergence, discussions

July 24th - July 28th, Begin of Phase 3

Finishing work with LS Operator, make sure convergence is fine on prepared toy examples

July 31st - August 4th

Documentation period

Prepare a notebook comparing all variational inference methods

Begin with ADVI, FullRankADVI

August 7th - August 11th

Continue with Normalizing and Householder Flows

August 14th - August 18th

Finish with Langevin Stein Operator

August 21st - August 25th, Final Week

Some unexpected stuff with examples / Bayesian Summer School

August 28th - August 29th, Submit final work

Submit

Future works

Read arXiv, collect ideas

Development Experience

Yandex Analyst-Developer Intern (summer 2016), PyMC3 developer

Other Experiences

Yandex Data Factory Analyst Intern (now)

Why this project?

I'm a great fan of Bayesian statistics and see it is usefull for many practical applications. I also love development and good codestyle. This project is interesting for me from both point of views. I'm also planning to use my results for my research projects and work.

Links

GitHub GSoC PR#178, Issue#152