ReactiveMP.jl: A Julia package for reactive variational Bayesian inference

Citation for published version (APA):

Document license:
CC BY-NC-ND

DOI:
10.1016/j.simpa.2022.100299

Document status and date:
Published: 01/05/2022

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.

Download date: 12. Jun. 2022
1. Introduction

Bayesian inference and probabilistic programming have become increasingly popular in model-based machine learning applications. Still, efficient Bayesian inference in many probabilistic models of practical interest remains a big challenge. Computing posterior distributions with Bayes rule often requires solving high-dimensional integrals that quickly become intractable. While popular methods such as Hamiltonian Monte Carlo (HMC) and the No-U-Turn Sampler (NUTS) use advanced sampling techniques to enable Bayesian inference in a very broad class of models, these methods do not scale well with the number of latent variables in the probabilistic model and may yield inaccurate posterior approximations for high-dimensional distributions. Variational Bayesian (VB) inference provides an interesting alternative to sampling-based inference, because it allows trading off computational complexity with accuracy and scales much better to high-dimensional conjugate probabilistic state-space models [1]. Almost all variants of VB inference can be formulated as a constrained “Bethe Free Energy” (BFE) minimization problem [2–4], which can be implemented by message passing on a factor graph-based representation of the probabilistic model [5].

To support the probabilistic programming community, we have developed ReactiveMP.jl [6], which is a Julia package for an automatable, efficient, and reactive message passing-based implementation of VB inference. ReactiveMP.jl uses analytical solutions for messages where possible and supports factorization and form constraints on the BFE optimization objective to enable approximation methods only where needed.

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

* Corresponding author.
E-mail addresses: d.v.bagaev@tue.nl (D. Bagaev), b.v.erp@tue.nl (B. van Erp), a.podusenko@tue.nl (A. Podusenko), bert.de.vries@tue.nl (B. de Vries).
1 Main author of the present software.
2 Partly financed by research program ZERO (no. P15-06), funded by the Netherlands Organisation for Scientific Research (NWO).

https://doi.org/10.1016/j.simpa.2022.100299
Received 6 April 2022; Accepted 24 April 2022

2665-9638/© 2022 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
2. Description and features

Given a probabilistic model definition, ReactiveMP.jl automatically generates a factor graph representation of this model and provides utilities to obtain posterior marginal distributions, conditioned on observed data. The main feature of our new implementation is the reactivity of the inference procedure. The core API has been designed with signal processing-related applications in mind. ReactiveMP.jl works well with infinite real-time data streams, reacts to new observations, performs online VB inference and integrates easily with event-driven applications, see Fig. 1. The inference back-end exposes posterior marginals as infinite streams that listen to changes in data and update themselves automatically. Although ReactiveMP.jl was designed for real-time reactive VB inference on infinite asynchronous data streams, it also supports static data sets. Since ReactiveMP.jl automates the inference process, it is very useful for quickly testing and iterating probabilistic model proposals before using them in practical applications.

ReactiveMP.jl comes with a comprehensive model specification language (GraphPPL.jl) that uses Julia macros to generate a factor graph representation of the probabilistic model. For usability, the model specification syntax closely resembles the mathematical model description, see Fig. 2. The model specification language supports local factorization and form constraints on the variational family of posteriors in different parts of the factor graph.

The current GitHub repository includes several demo usage examples such as linear Gaussian state-space models, autoregressive models, invertible neural networks (normalizing flows), hierarchical Gaussian filter models, infinite data stream processing, inference in mixture models, and others. The toolbox comes with documentation and a user-friendly API to enhance built-in functionality with additional custom factor nodes, custom message passing update rules, and custom approximation methods. The computational pipeline of ReactiveMP.jl is customizable as well. By default, the pipeline supports sum–product message passing, variational message passing, expectation propagation, and expectation maximization, but users are free to experiment and enable novel message passing-based inference algorithms on top of the existing tools.

ReactiveMP.jl integrates well with other Julia packages. For example, the resulting inference procedure is compatible with automatic differentiation packages such as ForwardDiff.jl [7] or ReverseDiff.jl, supports static matrices from the StaticArrays.jl package, and works well with different floating point numbers such as Float128 from DoubleFloats.jl or BigFloat from Julia Base. A user may also use global optimization packages from the Julia ecosystem, such as Optim.jl, to perform a global optimization on model (hyper) parameters.

3. Impact overview

Over the past two years, we have extended and improved ReactiveMP.jl in response to our own needs for automating inference in practical models. ReactiveMP.jl currently enables Bayesian inference in complex and sophisticated probabilistic state-space models. The built-in model specification language significantly simplifies testing and iteration of different probabilistic model proposals for signal processing applications. The built-in support for specifying constraints on the BFE enables custom approximation methods in different parts of a factor graph representation of the model. As a result, even with these ongoing developments, our team has tested new sophisticated probabilistic models and published multiple scientific articles in peer-reviewed journals and conferences with experimental results that were generated by ReactiveMP.jl [1,5,8–11].

We designed and implemented the reactive message passing-based Bayesian inference engine to be very efficient from a computational resources point of view; see Table 1. ReactiveMP.jl runs Bayesian inference for conjugate state-space models that may include hundreds of thousands of random variables in a model on a standard office Macbook laptop [1,8,10]. In models where advanced sampling-based inference methods take minutes to complete, ReactiveMP.jl yields inference results in milliseconds. However, it should be noted that ReactiveMP.jl currently supports automated inference for a smaller set of models than state-of-the-art sampling-based inference toolboxes. We expect to support and extend our toolbox for a larger set of possible probabilistic models in the future releases.

We believe that a fast iterative model design process, due to the focus on speed and efficiency of the inference procedures in ReactiveMP, will play a vital role in getting Bayesian modeling to become a highly valued standard tool for the practicing engineer.

4. Limitations and future work

While ReactiveMP.jl supports both conjugate and non-conjugate model definitions, inference capabilities for non-conjugate cases are still limited. We plan to extend the list of possible approximation methods for non-conjugate cases, e.g., by the Laplace approximation, in future releases of our toolbox.

Currently, ReactiveMP.jl does not support automatic Bayesian inference for any nonlinear deterministic relationships between latent states in a probabilistic model. This is purely a technical limitation, and we plan to resolve this in future releases of our toolbox. As mentioned before, the user API provides tools, sufficient documentation, and examples to create custom factor nodes, including nonlinear deterministic factor nodes with custom approximation methods for messages as a work-around.
Another technical limitation of the current implementation is the lack of the ability to create a factor node with a dynamic number of edges. The factor graph representation of a probabilistic model in ReactiveMP.jl requires a fixed number of edges for each type of factor node. For example, the Gaussian node has exactly three edges. Static edges allow for pre-compiling different parts of the inference procedure and to use computational resources more efficiently. Nevertheless, this limitation can be resolved manually with a custom factor node implementation, as we did for the Gaussian and Gamma mixture nodes [8]. However, there is currently no user-friendly API to simplify this process.

For future versions of the ReactiveMP.jl package, we consider several possible research directions. As mentioned, we plan to enable automatic VB inference for probabilistic models involving non-linear deterministic relationships between unknowns with the help of sampling-based solutions and stochastic variational inference methods [12,13]. We plan to integrate with existing sampling-based Bayesian inference solutions, such as Turing.jl [14] and others. Another interesting research direction is to enable factor graph modifications during the inference procedure at run-time. This feature, in principle, would enable model structure adaptation based on new incoming observations without the need to stop the inference procedure, see Fig. 1. We are also investigating automatic interrupt criteria for the Bethe Free Energy optimization process based on local BFE contributions, known as scale factors [15, Ch.6]. We plan to build an interactive, real-time, and user-friendly graphical browser that will simplify debugging and visualizing VB inference. In addition, we plan to enable the execution of message passing inference in parallel with multiple CPUs.

5. Conclusions

We have developed and continue to enhance a new Bayesian inference toolbox named ReactiveMP.jl, which is based on a reactive implementation of variational message passing in a factor graph. Together with other Julia packages, including GraphPLL.jl, Rocket.jl, Distributions.jl [16], and others, this toolbox forms an ecosystem to automate variational Bayesian inference in a large set of probabilistic models. The resulting inference procedure is very efficient and allows learning from both static and dynamic large data sets. The reactive implementation enables VB inference to run in real-time for potentially infinite asynchronous data streams. Factorization and form constraints on the variational family of posteriors allow trading off computational complexity versus accuracy in the inference process. ReactiveMP.jl is easy to extend and supports adding custom factors and message passing-based inference methods for advanced users. We believe that the core functionality can be used by a wider audience of researchers in the field of probabilistic modeling. We hope that ReactiveMP.jl advances the application of Bayesian modeling methods in both the industrial and academic machine learning communities.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We would like to acknowledge Ismail Senoz and Thijs van de Laar from Eindhoven University of Technology for their support and informative discussions about the theory and implementation aspects of message passing-based Bayesian inference. This work was partly financed by research program ZERO (no. P15-06), funded by the Netherlands Organisation for Scientific Research (NWO).

References


Table 1

<table>
<thead>
<tr>
<th>Model</th>
<th>Dataset size</th>
<th>VB (ReactiveMP.jl)</th>
<th>HMC + PG (Turing.jl)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Gaussian State-Space Model (2-dimensional state)</td>
<td>300</td>
<td>33 ms</td>
<td>36 s</td>
</tr>
<tr>
<td>Linear Gaussian State-Space Model (4-dimensional state)</td>
<td>300</td>
<td>40 ms</td>
<td>117 s</td>
</tr>
<tr>
<td>Hidden Markov Model (3-dimensional state)</td>
<td>250</td>
<td>62 ms</td>
<td>784 s</td>
</tr>
<tr>
<td>Hierarchical Gaussian Filter Model (1-dimensional state)</td>
<td>250</td>
<td>95 ms</td>
<td>13 s</td>
</tr>
</tbody>
</table>


CRediT authorship contribution statement


