juliacon

My JuliaCon proceeding

1st author[1], 2nd author[1, 2], and 3rd author[2]

[1]University
[2]National Lab

ABSTRACT

GraphPPL.jl is a Julia package for convenient probabilistic model and variational Bayesian inference constraints specification. The package includes a set of functions defining DSL for graphical model creation ....

Keywords

Julia, Optimization, Game theory, Compiler

1. Background

Bayesian modeling has become a popular framework for many important real-time machine learning applications, such as object-tracking, speech recognition, robot navigation, financial applications, etc.

(dmitry) add references

. Many useful probabilistic time-series models often contain a large number of latent variables and complex conditional dependencies. In addition, exact inference in many models of interests is usually intractable and requires variational Bayesian methods. Thus, it is important to have an easy way for specification such large models as well as convenient way for specification of variational distribution that acts as a proxy to the exact solution.

2. Problem statement

Existing packages for automated Bayesian inference in the Julia language ecosystem, such as Turing.jl, Stan.jl, and Soss.jl, support probabilistic model specification by well-designed macro-based meta languages. These packages assume that inference is executed by black-box variational or sampling-based methods. In principle, for conjugate probabilistic time-series models, message passing-based variational inference by minimization of a constrained Bethe Free Energy yields approximate inference solutions obtained with cheaper computational costs.

3. Solution proposal

In this contribution, we develop a user-friendly and comprehensive meta language for specification of both a probabilistic model and variational inference constraints that balance accuracy of inference results with computational costs. The GraphPPL.jl package implements a user-friendly specification language for both the model and the inference constraints. GraphPPL.jl exports the ‘@model‘ macro to create a probabilistic model in the form of a factor graph that is compatible with ReactiveMP.jl’s

(dmitry) add reference

reactive message passing-based inference engine. To enable fast and accurate inference, all message update rules default to precomputed analytical solutions. The ReactiveMP.jl package already implements a selection of precomputed rules. If an analytical solution is not available, then the GraphPPL.jl package provides ways to tweak, relax, and customize local constraints in selected parts of the factor graph. To simplify this process, the package exports the ‘@constraints‘ macro to specify extra factorization and form constraints on the variational posterior [1]. For advanced use cases, GraphPPL.jl exports the ‘@meta‘ macro that enables custom message passing inference modifications for each node in a factor graph representation of the model. This approach enables local approximation methods only if necessary and allows for efficient variational Bayesian inference.

4. Evaluation

Over the past two years, our probabilistic modeling ecosystem comprising GraphPPL.jl, ReactiveMP.jl, and Rocket.jl has been battle tested with many sophisticated models that led to several publications in high-ranked journals such as Entropy [1] and Frontiers [2], and conferences like MLSP-2021 [3] and ISIT-2021 [4]. The current contribution enables a user-friendly approach to very sophisticated Bayesian modeling problems.

5. Conclusions

We believe that a user-friendly specification of efficient Bayesian inference solutions for complex models is a key factor to expedite application of Bayesian methods. We developed a complete ecosystem for running efficient, fast, and reactive variational Bayesian inference with a user-friendly specification language for the probabilistic model and variational constraints. We are excited to present GraphPPL.jl as a part of our complete variational Bayesian inference ecosystem and discuss the advantages and drawbacks of this approach.

6. ORIGINAL DOCUMENT BELOW

7. Introduction

The LaTeX document preparation system is a special version of the TeX; typesetting program where-in a collection of TeX; commands are added to LaTeX to simplify typesetting. Importantly, it allows the author to concentrate on the logical structure of the document rather than its visual layout.

Moreover, LaTeX provides a consistent and comprehensive document preparation interface. There are simple-to-use commands for

generating a table of contents, lists of figures and/or tables, and indexes. LaTeX can automatically number list entries, equations, figures, tables, and footnotes, as well as articles, sections, and subsections. Using this numbering system, bibliographic citations, page references, and cross references to any other numbered entity (e.g. article, section, equation, figure, list entry, etc.) become quite simple and straightforward. The use of LaTeX document classes allows a simple change of class to transform the appearance of your document.

LaTeX is a powerful tool for managing long and complex documents. In particular, partial processing enables long documents to be produced article by article without losing sequential information. The use of document classes allows a simple change of style (or style option) to transform the appearance of your document.

8. The JuliaCon Article Class

The juliacon class file preserves the standard LATEX interface such that any document that can be produced using the standard LATEX article class can also be produced with the class file.

It is likely that the make up will change after file submission. For this reason, we ask you to ignore details such as slightly long lines, page stretching, or figures falling out of synchronization, as these details can be dealt with at a later stage.

Use should be made of symbolic references (`\ref`) in order to protect against late changes of order, etc.

9. USING THE JuliaCon Article CLASS FILE

If the file `juliacon.cls` is not already in the appropriate system directory for LaTeX files, either arrange for it to be put there or copy it to your working directory. The `juliacon` document class is implemented as a complete class, not a document style option. In order to use the `juliacon document class, replace` \verbarticle| by `juliacon` in the `\documentclass` command at the beginning of your document:

```
\documentclass{article}
```

replace by

```
\documentclass{juliacon}
```

In general, the following standard document `style` options should *not* be used with the *article* class file:

(1) `10pt`, `11pt`, `12pt` ? unavailable;
(2) `twoside` (no associated style file) ? `twoside` is the default;
(3) `fleqn`, `leqno`, `titlepage` ? should not be used;

10. Additional Document Style Options

The following additional style option is available with the `juliacon` class file:

Please place any additional command definitions at the very start of the LaTeX file, before the `\begin{document}`. For example, user-defined `\def` and `\newcommand` commands that define macros for technical expressions should be placed here. Other author-defined macros should be kept to a minimum.

Commands that differ from the standard LaTeX interface, or that are provided in addition to the standard interface, are explained in this guide. This guide is not a substitute for the LaTeX manual itself. Authors planning to submit their papers in LaTeX are advised to use `juliacon.cls` as early as possible in the creation of their files.

11. Additional features

In addition to all the standard LaTeX design elements, the `juliacon` class file includes the following features: In general, once you have used the additional `juliacon.cls` facilities in your document, do not process it with a standard LaTeX class file.

11.1 Titles, Author's Name, and Affiliation

The title of the article, author's name, and affiliation are used at the beginning of the article (for the main title). These can be produced using the following code:

```
\title{ This is an example of article title} }
\author{
   \large 1st Author \\[-3pt]
   \normalsize 1st author's affiliation  \\[-3pt]
    \normalsize 1st line of address \\[-3pt]
    \normalsize 2nd line of address \\[-3pt]
    \normalsize 1st author's email address \\[-3pt]
  \and
   \large 2nd Author \\[-3pt]
   \normalsize 2nd author's affiliation  \\[-3pt]
    \normalsize 1st line of address \\[-3pt]
    \normalsize 2nd line of address \\[-3pt]
    \normalsize 2nd author's email address \\[-3pt]
\and
   \large 3rd Author \\[-3pt]
   \normalsize 3rd author's affiliation  \\[-3pt]
    \normalsize 1st line of address \\[-3pt]
    \normalsize 2nd line of address \\[-3pt]
    \normalsize 3rd author's email address \\[-3pt]
}
\maketitle
```

11.2 Writing Julia code

A special environment is already defined for Julia code, built on top of *listings* and *jlcode*.

```
\begin{lstlisting}[language = Julia]
using Plots

x = -3.0:0.01:3.0
y = rand(length(x))
plot(x, y)
\end{lstlisting}
```

```julia
using Plots

x = -3.0:0.01:3.0
y = rand(length(x))
plot(x, y)
```

11.3 Abstracts, Key words, term etc...

At the beginning of your article, the title should be generated in the usual way using the `\maketitle` command. For genaral tem and keywords use `\terms`, `\keywords` commands respectively.

Table 1. If necessary, the tables can be extended both columns.

Label	Description	Number of Users	Number of Queries
Test 1	Training Data	70	104
Test 2	Testing Data I		105
Test 3	Testing Data II	30	119
	Total	100	328

This is an example of table footnote.

The abstract should be enclosed within an abstract environment, All these environment can be produced using the following code:

```
\terms{Experimentation, Human Factors}

\keywords{Face animation, image-based modelling...}

\begin{abstract}
In this paper, we propose a new method for the
systematic determination of the model's base of
time varying delay system. This method based on
the construction of the classification data related
to the considered system. The number, the orders,
the time delay and the parameters of the local
models are generated automatically without any
knowledge about the full operating range of the
process. The parametric identification of the local
models is realized by a new recursive algorithm for
on line identification of systems with unknown time
delay. The proposed algorithm allows simultaneous
estimation of time delay and parameters of
discrete-time systems. The effectiveness of
the new method has been illustrated through
simulation.
\end{abstract}
```

12. Some guidelines

The following notes may help you achieve the best effects with the `juliacon` class file.

12.1 Sections

LaTeX 2$_\varepsilon$ provides four levels of section headings and they are all defined in the `juliacon` class file:

—`\section`

—`\subsection`

—`\subsubsection`

—`\paragraph`

Section headings are automatically converted to allcaps style.

12.2 Lists

The `juliacon` class file provides unnumbered lists using the `unnumlist` environment for example,

First unnumbered item which has no label and is indented from the left margin.

Second unnumbered item.

Third unnumbered item.

The unnumbered list which has no label and is indented from the left margin. was produced by:

```
\begin{unnumlist}
\item First unnumbered item...
\item Second unnumbered item...
\item Third unnumbered item...
\end{unnumlist}
```

The `juliacon` class file also provides hyphen list using the `itemize` environment for example,

—First unnumbered bulleted item which has no label and is indented from the left margin.

—Second unnumbered bulleted item.

—Third unnumbered bulleted item which has no label and is indented from the left margin.

was produced by:

```
\begin{itemize}
\item First item...
\item Second item...
\item Third item...
\end{itemize}
```

Numbered list is also provided in acmtog class file using the enumerate environment for example,

(1) The attenuated and diluted stellar radiation.
(2) Scattered radiation, and
(3) Reradiation from other grains.

was produced by:

```
\begin{enumerate}
\item The attenuated...
\item Scattered radiation, and...
\item Reradiation from other grains...
\end{enumerate}
```

12.3 Illustrations (or figures)

The `juliacon` class file will cope with most of the positioning of your illustrations and you should not normally use the optional positional qualifiers on the `figure` environment that would override these decisions.

The figure 1 is taken from the JuliaGraphs organization [1].
Figure captions should be *below* the figure itself, therefore the `\caption` command should appear after the figure or space left for an illustration. For example, Figure 1 is produced using the following commands:

```
\begin{figure}
\centerline{\includegraphics[width=20pc]{Graphics.eps}}
\caption{An example of the testing process for a
binary tree. The globa null hypothesis is tested
```

[1] https://github.com/JuliaGraphs

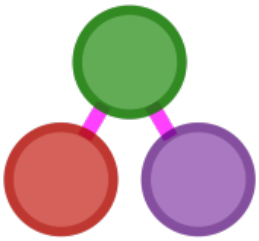

Fig. 1. This is example of the image in a column.

Table 2. Tuning Set and Testing Set

Label	Description	Number of Users	Number of Queries
Test 1	Training Data	70	104
Test 2	Testing Data I		105
Test 3	Testing Data II	30	119
	Total	100	328

```
first at level $\alpha$ (a), and the level of
individual variables is reached last (d). Note
that individual hypotheses can be tested at
level $\alpha/4$ and not $\alpha/8$ as one might
expect at first.}
\label{sample-figure_2}
\end{figure}
```

Figures can be resized using first and second argument of `\includegraphics` command. First argument is used for modifying figure height and the second argument is used for modifying figure width respectively.

Cross-referencing of figures, tables, and numbered, displayed equations using the `\label` and `\ref` commands is encouraged. For example, in referencing Figure 1 above, we used `Figure~\ref{sample-figure}`

12.4 Tables

The `juliacon` class file will cope with most of the positioning of your tables and you should not normally use the optional positional qualifiers on the table environment which would override these decisions. Table captions should be at the top.

```
\begin{table}
\tbl{Tuning Set and Testing Set}{
\begin{tabular}{|l|l|c|c|}\hline
Label & \multicolumn{1}{c|}{Description}
& Number of Users &
Number of Queries\\\hline
Train70 & Training Data &
\smash{\raise-7pt\hbox{70}} & 104\\
\cline{1-2}\cline{4-4}
Test70 & Testing Data I & & 105\\\hline
Test30 & Testing Data II & 30 & 119\\\hline
& Total & 100 & 328\\\hline
\end{tabular}}
\end{table}
```

12.5 Landscaping Pages

If a table is too wide to fit the standard measure, it may be turned, with its caption, to 90 degrees. Landscape tables cannot be produced directly using the `juliacon` class file because TEX itself cannot turn the page, and not all device drivers provide such a facility. The following procedure can be used to produce such pages.

Use the package `rotating` in your document and change the coding from

```
\begin{table}...\end{table}
to
\begin{sidewaystable}...\end{sidewaystable}
and for figures
\begin{figure}...\end{figure}
to
\begin{sidewaysfigure}...\end{sidewaysfigure}
```

environments in your document to turn your table on the appropriate page of your document. For instance, the following code prints a page with the running head, a message half way down and the table number towards the bottom.

```
\begin{sidewaystable}
\tbl{Landscape table caption to go here.}{...}
\label{landtab}
\end{sidewaystable}
```

12.6 Double Column Figure and Tables

For generating the output of figures and tables in double column we can use the following coding:

(1) For Figures:

```
\begin{figure*}...\end{figure*}
```

(2) For landscape figures:

```
\begin{sidewaysfigure*}...\end{sidewaysfigure*}
```

(3) For Tables:

```
\begin{table*}...\end{table*}
```

(4) For landscape tables:

```
\begin{sidewaystable*}...\end{sidewaystable*}
```

12.7 Typesetting Mathematics

The `juliacon` class file will set displayed mathematics with center to the column width, provided that you use the LATEX 2_ε standard of open and closed square brackets as delimiters. The equation

$$\sum_{i=1}^{p} \lambda_i = (S)$$

was typeset using the acmtog class file with the commands

```
\[
\sum_{i=1}^p \lambda_i = (S)
\]
```

For display equations, cross-referencing is encouraged. For example,

```
\begin{equation}
(n-1)^{-1} \sum^n_{i=1} (X_i - \overline{X})^2.
\label{eq:samplevar}
\end{equation}
Equation~(\ref{eq:samplevar}) gives the formula for
sample variance.
```

The following output is generated with the above coding:

$$(n-1)^{-1}\sum_{i=1}^{n}(X_i-\overline{X})^2. \tag{1}$$

Equation (1) gives the formula for sample variance.

12.8 Enunciations

The `juliacon` class file generates the enunciations with the help of the following commands:

```
\begin{theorem}...\end{theorem}
\begin{strategy}...\end{strategy}
\begin{property}...\end{property}
\begin{proposition}...\end{proposition}
\begin{lemma}...\end{lemma}
\begin{example}...\end{example}
\begin{proof}...\end{proof}
\begin{definition}...\end{definition}
\begin{algorithm}...\end{algorithm}
\begin{remark}...\end{remark}
```

The above-mentioned coding can also include optional arguments such as

```
\begin{theorem}[...]. Example for theorem:
\begin{theorem}[Generalized Poincare Conjecture]
Four score and seven ... created equal.
\end{theorem}
```

THEOREM 1 GENERALIZED POINCARE CONJECTURE. *Four score and seven years ago our fathers brought forth, upon this continent, a new nation, conceived in Liberty, and dedicated to the proposition that all men are created equal.*

12.9 Extract

Extract environment should be coded within

```
\begin{extract}..\end{extract}
```

12.10 Balancing column at last page

For balancing the both column length at last page use :

```
\vadjust{\vfill\pagebreak}
```

at appropriate place in your TEX file or in bibliography file.

13. Handling references

References are most easily (and correctly) generated using the BIBTEX, which is easily invoked via

```
\bibliographystyle{juliacon}
\bibliography{ref}
```

When submitting the document source (.tex) file to external parties, the ref.bib file should be sent with it. [1]

14. References

[1] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017. doi:10.1137/141000671.