Software Packages for Holonomic Gradient Method

Tamio Koyama[1], Hiromasa Nakayama[2], Katsuyoshi Ohara[3], Tomonari Sei[4], and Nobuki Takayama[5]

[1] Department of Mathematical Informatics, Graduate School of Information Science and Technology, University of Tokyo, JSPS Research Fellow, Japan
tkoyama@stat.t.u-tokyo.ac.jp,
http://www.stat.t.u-tokyo.ac.jp/~tkoyama/
[2] Department of Mathematics, Tokai University, Japan
nakayama@tokai-u.jp,
http://sm.u-tokai.ac.jp/~nakayama/
[3] Faculty of Mathematics and Physics, Kanazawa University, Japan
ohara@air.s.kanazawa-u.ac.jp,
http://air.s.kanazawa-u.ac.jp/~ohara/
[4] Department of Mathematics, Keio University , Japan
sei@math.keio.ac.jp,
http://www.math.keio.ac.jp/~sei/
[5] Department of Mathematics, Kobe University, Japan
takayama@math.kobe-u.ac.jp,
http://www.math.kobe-u.ac.jp/~taka/

Abstract. We present software packages for the holonomic gradient method (HGM). These packages compute normalizing constants and the probabilities of some regions. While many algorithms which compute integrals over high-dimensional regions utilize the Monte-Carlo method, our HGM utilizes algorithms for solving ordinary differential equations such as the Runge-Kutta-Fehlberg method. As a result, our HGM can evaluate many integrals with a high degree of accuracy and moderate computational time. The source code of our packages is distributed on our web page [12]. .

Keywords: holonomic gradient method, normalizing constant, region probability, Bingham prior, R project

1 Introduction

The numerical evaluation of the normalizing constant for a given statistical distribution is a fundamental problem in statistics. For example, the normalizing constant of the Gaussian distribution is expressed in terms of a rational expression of a parameter of the distribution named the standard deviation. However, normalizing constants of many interesting statistical distributions do not have such closed expressions.

The *holonomic gradient method*, HGM in short, is a general method to evaluate normalizing constant numerically for several parameters in the framework of Zeilberger's holonomic systems approach [11]. In fact, broad classes of normalizing constants are holonomic functions with respect to parameters. Then, such normalizing constants satisfy holonomic systems of linear partial differential equations.

The HGM consists of three steps for a given normalizing constant. (1) Finding a holonomic system satisfied by the normalizing constant. We may use computational or theoretical methods to find it. Gröbner basis and related methods are used. (2) Finding an initial value vector for the holonomic system. This is equivalent to evaluating the normalizing constant and its derivatives at a point. This step is usually performed by a series expansion. (3) Solving the holonomic system numerically. We utilize several methods in numerical analysis such as the Runge-Kutta method of solving ordinary differential equations and solvers of systems of linear equations.

The HGM was proposed in 2011 by a group of people including us [6] and has given several new results. For example, the orthant probability is the normalizing constant of the multivariate normal distribution restricted to the first orthant. The HGM can evaluate it in a high accuracy up to the 20 dimensional case when the mean vector is near the origin. In the 20 dimensional case, we numerically solve an ordinary differential equation of rank $2^{20} = 20,148,576$.

We have developed software packages for the HGM. Packages based on computer algebra systems help us to solve steps (1) and (2). We have implemented the step (3) for the Fisher-Bingham distribution, the Bingham distribution, the orthant probability, the Fisher distribution on SO(3), some of A-distributions, and the distribution function of the largest root of a Wishart matrix in the language C and/or in the system for statistics R [7]. An implementation for the polyhedral probability is a project in progress. We find an interesting interplay with systems for polytopes in the project. Further references and current implementations are listed in [12].

This paper is dedicated to Kenta Nishiyama in our memory.

2 Distributions and Algorithms

We give a brief discussion on the Bingham distribution and the orthant probability in view of the HGM in this section. As to the Fisher distributions on $SO(3)$, the largest roots of Wishart matrices, Fisher-Bingham distributions, and A-distributions, we refer to papers in [12].

2.1 Bingham distribution

The Bingham distribution is a probability distribution on the (p-1)-dimensional sphere defined as

$$\frac{1}{Z(\Sigma)} \exp(x^\top \Sigma^{-1} x) \mu(dx) \quad (x \in S^{p-1})$$

where Σ is a p times p positive definite matrix, $\mu(dx)$ is the uniform measure on the sphere, and $Z(\Sigma)$ is the normalizing constant ($Z(\Sigma)$ is also denoted by $c(\Sigma)$ in literatures). We denote by $x^\top$ the transpose of x. We can assume without loss of generality that the matrix Σ^{-1} is a diagonal matrix such as $\mathrm{diag}(\theta_1, \ldots, \theta_{p-1}, 0)^\top$.

In [9], the HGM for this normalizing constant is discussed, and an explicit form of a Pfaffian equation associated with the normalizing constant has been given. The size of the matrix in the Pfaffian equation is p. In the current implementation, we evaluate the initial value for the HGM by a series expansion. The complexity to evaluate it is proportional to the number of the terms in the truncated series. Hence, the complexity is $O(p^N)$. Here, we denote by N the degree of the truncated series. Thus, the computational complexity of the HGM for this problem is estimated as

Theorem 1. *The complexity of the series expansion method and the HGM for the normalizing constant of the Bingham distribution on the* $(p-1)$*-dimensional sphere is bounded by*

$$O(p^N) + O(p^2) \times (\text{steps of the Runge-Kutta method}).$$

Note that the holonomic system for the normalizing constant of the Bingham distribution and it's holonomic rank are not determined rigorously. There might exist a smaller system than that in [9]. Thus, the complexity in the above theorem is the upper bound of the complexity. We conjecture that the above complexity gives the lower bound of the complexity of the HGM for the Bingham distributions.

We provide a package of the HGM for the system for statistics R [7]. The function `hgm.ncBingham(th, ...)` in our R package `hgm` performs the HGM for Bingham distributions with the deSolve package. The initial value for the HGM is computed by the power series expansion. This function also computes derivatives of the normalizing constant of the Bingham distribution at any specified point. The variable `th` is a $(p-1)$-dimensional vector which specifies the first $(p-1)$ components of the parameter vector of the Bingham distribution on the $(p-1)$-dimensional sphere. The p-th parameter is assumed to be zero.

For $\Sigma^{-1} = \mathrm{diag}(1, 3, 5, 0)$, we can obtain the normalizing constant as

```
hgm.ncBingham(c(1,3,5))
```

after loading the package with the command `library('hgm')`

2.2 The orthant probability

The orthant probability is the probability with which the random vector, which is normally distributed with the mean vector μ and the covariance matrix Σ, falls in the first orthant, and it can be written as

$$\int_0^\infty \cdots \int_0^\infty \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right) dx_1 \ldots dx_d.$$

where we denote by d the dimension.

In [3], the HGM for the orthant probability is discussed, and an explicit form of a holonomic system and a Pfaffian equation associated with the probability is given. The holonomic rank of the system, which equals to the size of the Pfaffian equation, is 2^d. The initial value of the HGM for the orthant probability can be given exactly at a point and the computational complexity of the evaluation of the initial value is $O(1)$. The following complexity statement is an easy consequence of Theorem 15 in [3], but it is fundamental.

Theorem 2. *The complexity of evaluating the d-dimensional orthant probability by the HGM is*

$$O(2^{2d}) \times (\text{steps of the Runge-Kutta method}).$$

The function `hgm.ncorthant(sigma, mu, ...)` in our R package evaluates the orthant probability by the HGM. The first variable `sigma` is the covariance matrix, and the second variable `mu` is the mean vector. This function calls a program written by the language C internally, which solves an ordinary differential equation with rank 2^d by a routine in the GNU scientific library (GSL) [2].

For example, when

$$d = 2, \quad \Sigma = \begin{pmatrix} 1 & 1/2 \\ 1/2 & 1 \end{pmatrix}, \quad \mu = \begin{pmatrix} 1 \\ 2 \end{pmatrix},$$

the orthant probability can be computed by the following script of R:

```
sigma <- matrix(c(1, 0.5, 0.5, 1), nrow =2)
mu <- c(1,2)
hgm.ncorthant(sigma,mu)
```

In this example, the rank of the ordinary differential equation equals to $2^2 = 4$. The performance of our implementation for larger d will be illustrated in 3.3.

3 Implementations

3.1 Building blocks of our package

Our algorithms for the holonomic gradient method require efficient and reliable numerical implementations of the Runge-Kutta method and solving numerically systems of linear equations. Our package uses the GSL [2], the deSolve package in R, BLAS, and LAPACK for this purpose.

Most of our algorithms are implemented in the language C. We provide two interfaces for our C-code. One is a command line interface and the other interface is R, which is a software system for statistics [7]. For example, in the problem mh (the largest roots of Wishart matrices), the function `mh_cwishart_gen` performs the HGM for mh. The both of the `main` function for the command line interface and the interface module `Rmh_cwishart_gen` for R call the common function `mh_cwishart_gen`.

The system R provides an easy and strong mechanism to include C code into the R system [10]. It is recommended by the CRAN repository policy to minimize the size of code and make effort to provide cross-platform code. Then, we have extracted the source code of some of the functions defined in `odeiv.h` in GSL for solving ordinary differential equations, and include them in our HGM package.

In the current implementation, both of command line interface and R interface are available for the problems mh, orthant, and so3. We provide only a command line interface for the Fisher-Bingham distribution. Because, our implementation relies on linear algebra functions of the GSL and extracting these functions for R or rewriting them in BLAS and LAPACK need some works, which will generate some new bugs without taking relatively long time of careful porting and debugging. We hope that R officially supports the GSL in a future.

3.2 Use of computer algebra systems for a reliable implementation

Since some ordinary differential equations for the HGM contain complicated expressions and also evaluation formulas of initial values are complicated, we utilize computer algebra systems to avoid bugs caused by writing programs by hand and to provide correct code. For example, our C implementation for the HGD (holonomic gradient descent) of the Fisher-Bingham distribution is automatically generated by code in Risa/Asir [8], which is a computer algebra system. Our implementation for the Wishart distribution is firstly written in Risa/Asir and contains several debugging and checking code of correctness of each steps (see `tk_jack.rr` in our package). After the code by Risa/Asir works correctly, we translate it into code in C.

3.3 Performance

We illustrate the performance of our implementation of the orthant probability.

We evaluate the orthant probability by the HGM for $\Sigma = ((1 + \delta_{ij})/2)$ and $\mu = 0$ where δ_{ij} is the Kronecker's delta. In this case, it is known that the orthant probability equals to $1/(d + 1)$. The table 1 shows the result of the HGM, the exact value of the orthant probability, and the CPU time for each d. The HGM is performed by the command `hgm_ko_orthant`, and it is compiled by the GNU C compiler v.4.7.2. We performed the experiments on an Intel(R) Xeon(R) CPU E5-4650 0 @ 2.70GHz with 252GB RAM, running Linux.

The computational time of `hgm_ko_orthant` increases rapidly when the dimension increases. This is a consequence of our complexity result (Theorem 2). However, our algorithm and implementation are faster in comparison with the existing software systems which evaluate the orthant probability with high accuracy. For example, the CPU time to compute the same problem by `pmvnorm` [5], which is in the R package, for the case $d = 10$ is 61.991 seconds. The function `pmvnorm` dissects an orthant probability into $(d - 1)!$ orthoscheme probabilities, and apply an effective iterative integration whose complexity is $O(d)$. Thus, the theoretical complexity of `pmvnorm` is proportional to $d!$. No other algorithms and

Table 1. Computational experiments on `hgm_ko_orthant`

dimension	HGM	exact	CPU time
2	0.333333331	0.333333333	0.00
3	0.249999998	0.250000000	0.00
4	0.199999998	0.200000000	0.00
5	0.166666666	0.166666667	0.02
6	0.142857142	0.142857143	0.06
7	0.125000000	0.125000000	0.14
8	0.111111111	0.111111111	0.39
9	0.100000000	0.100000000	0.91
10	0.090909091	0.090909091	2.40
20	0.047619048	0.047619048	22721.30

implementations achieve our timing with more than the 9 digits accuracy as far as we know.

4 Applications

Normalizing constants are fundamental in statistics. In [6], we demonstrate that Fisher's maximal likelihood estimate can be performed by utilizing the HGM to evaluate normalizing constants and its derivatives. The orthant probability of multivariate normal distribution is also used in various area of statistics. In this section, we sketch an application to Bayesian analysis.

Consider the multinomial distribution of size n

$$f(y|\pi) = \frac{n!}{y_1! \cdots y_p!} \pi_1^{y_1} \cdots \pi_p^{y_p}, \quad y = (y_1, \ldots, y_p) \in \mathbb{Z}^p_{\geq 0}, \quad \sum_{i=1}^p y_i = n,$$

where $\pi = (\pi_1, \ldots, \pi_p)$ belongs to the simplex $\Delta^{p-1} = \{\pi \geq 0 \mid \sum_i \pi_i = 1\}$. For the multinomial distribution, the Dirichlet prior density is often used in the Bayesian context (e.g. [1]).

We introduce a different class of prior densities. Put $\Sigma^{-1} = \mathrm{diag}(\theta_1, \ldots, \theta_p)$ and $\pi_i = x_i^2$ for each i in the Bingham distribution defined in Section 2. The random variable $\pi = (\pi_1, \ldots, \pi_p)$ has the density function

$$f(\pi) = \frac{2\pi_1^{-1/2} \cdots \pi_p^{-1/2} e^{\sum_{i=1}^p \theta_i \pi_i}}{c(\theta)}, \quad \pi \in \Delta^{p-1},$$

with respect to $d\pi = d\pi_1 \cdots d\pi_{p-1}$, where $c(\theta) = Z(\Sigma)$ is the Bingham normalizing constant. We call it the Bingham prior density.

One of important quantities in Bayesian analysis is the marginal likelihood $f_{\mathrm{mar}}(y) = \int f(y|\pi) f(\pi) d\pi$ (see e.g. Section 3.4 of [1]). For the Bingham prior, it is shown that

$$f_{\mathrm{mar}}(y) = \frac{n!}{y_1! \cdots y_p!} \frac{\prod_{i=1}^p \Gamma(y_i + \frac{1}{2})}{\pi^{n+p/2}} \frac{c(\theta, 2y + 1_p)}{c(\theta)}, \tag{1}$$

where $1_p = (1, \ldots, 1) \in \mathbb{R}^p$, and $c(\theta, d)$ for $d = (d_1, \ldots, d_p)$ denotes the Bingham normalizing constant on the $(\sum_{i=1}^p d_i - 1)$-dimensional sphere with the multiplicity index d, that is, $c(\theta, d) = c((\theta_1, \ldots, \theta_1, \ldots, \theta_p, \ldots, \theta_p))$, where θ_i appears d_i times. The formula (1) is proved in the same way as Proposition 1 of [4]. The other quantities such as posterior density and predictive density are written in terms of $c(\theta, d)$ as well.

For example, if $p = 4$, $\theta = (1, 3, 5, 0)$ and $y = (2, 0, 3, 1)$, then the marginal likelihood is evaluated by the following R script

```
y = c(2,0,3,1); th = c(1,3,5); n = sum(y); p = length(y)
a0 = lfactorial(n) - sum(lfactorial(y))
a1 = sum(lgamma(y+1/2)) - (n+p/2)*log(pi)
a2 = hgm.ncBingham(th, d=2*y+1, withvol=TRUE, logarithm=TRUE)[1]
a3 = hgm.ncBingham(th, withvol=TRUE, logarithm=TRUE)[1]
exp(a0 + a1 + a2 - a3)
```

where the `withvol` option specifies that the total uniform measure on the sphere is its volume (not normalized to 1) and the `logarithm` option specifies the output is in the logarithmic scale. The result of the script is 0.008963549. One can check that the total of $f_{\mathrm{mar}}(y)$ over possible y's given n is 1 (up to numerical error).

For a given data y, the hyper-parameter θ in (1) can be selected by maximizing the marginal likelihood. This maximization problem is analogous to the maximum likelihood estimation and then the HGD [6] can be applied in principle, but details have not been studied and this MLE has not been implemented yet in our package. It is also an interesting project in progress to derive a holonomic system and a Pfaffian system for other prior densities.

References

1. C.M.Bishop, Pattern Recognition and Machine Learning, 2006, Springer.
2. GNU Scientific Library, `http://www.gnu.org/s/gsl`
3. T.Koyama, A.Takemura, Calculation of Orthant Probabilities by the Holonomic Gradient Method, arxiv:1211.6822
4. A. Kume, A.T.A. Wood, On the Derivatives of the Normalizing Constant of the Bingham Distribution, Statistics and Probability Letters 77 (2007), 832–837.
5. T.Miwa, H.J.Hayter, S.Kuriki, The evaluation of general non-centered orthant probabilities, Journal of the Royal Statistical Society: Series B 65 (2003), 223-234.
6. H.Nakayama, K.Nishiyama, M.Noro, K.Ohara, T.Sei, N.Takayama, A.Takemura, Holonomic Gradient Descent and its Application to the Fisher-Bingham Integral, Advances in Applied Mathematics 47 (2011), 639–658.
7. The R Project for Statistical Computing, `http://www.r-project.org`
8. Risa/Asir, a computer algebra system, `http://www.math.kobe-u.ac.jp/Asir`
9. T.Sei, A.Kume, Calculating the Normalizing Constant of the Bingham Distribution on the Sphere using the Holonomic Gradient Method, Statistics and Computing, 2013 (online).
10. Writing R Extensions, `http://cran.r-project.org/doc/manuals/R-exts.html`
11. D.Zeilberger, A Holonomic Systems Approach to Special Function Identities, Journal of Computational and Applied Mathematics 32 (1990), 321–368.
12. `http://www.math.kobe-u.ac.jp/OpenXM/Math/hgm/ref-hgm.html`