Comput Stat
DOI 10.1007/s00180-011-0232-x

SHORT NOTE

A short note on parameter approximation for von Mises-Fisher distributions: and a fast implementation of $I_s(x)$

Suvrit Sra

Received: 19 September 2008 / Accepted: 18 January 2011

Abstract In high-dimensional directional statistics one of the most basic probability distributions is the *von Mises-Fisher* (vMF) distribution. Maximum likelihood estimation for the vMF distribution turns out to be surprisingly hard because of a difficult transcendental equation that needs to be solved for computing the *concentration parameter* κ. This paper is a followup to the recent paper of Tanabe et al. (Comput Stat 22(1):145–157, 2007), who exploited inequalities about Bessel function ratios to obtain an interval in which the parameter estimate for κ should lie; their observation lends theoretical validity to the heuristic approximation of Banerjee et al. (JMLR 6:1345–1382, 2005). Tanabe et al. (Comput Stat 22(1):145–157, 2007) also presented a fixed-point algorithm for computing improved approximations for κ. However, their approximations require (potentially significant) additional computation, and in this short paper we show that given the *same* amount of computation as their method, one can achieve more accurate approximations using a truncated Newton method. A more interesting contribution of this paper is a simple algorithm for computing $I_s(x)$: the modified Bessel function of the first kind. Surprisingly, our naïve implementation turns out to be several orders of magnitude faster for large arguments common to high-dimensional data, than the standard implementations in well-established software such as MATHEMATICA©, MAPLE©, and GP/PARI.

Keywords von Mises-Fisher distribution · Maximum-likelihood · Numerical approximation · Modified Bessel function · Bessel ratio

Electronic supplementary material The online version of this article (doi:10.1007/s00180-011-0232-x) contains supplementary material, which is available to authorized users.

S. Sra (✉)
Max-Planck Institute (MPI) for biological Cybernetics, Tübingen, Germany
e-mail: suvrit.sra@tuebingen.mpg.de

Published online: 03 February 2011

1 Introduction

The *von Mises-Fisher* (vMF) distribution, defined on the unit hypersphere, is fundamental to high-dimensional directional statistics Mardia and Jupp (2000). Maximum likelihood estimation, and consequently the M-step of an Expectation Maximization (EM) algorithm based on the vMF distribution can be surprisingly hard because of a difficult nonlinear equation that needs to be solved for estimating the *concentration parameter* κ.

In this paper we review maximum-likelihood parameter estimation for κ and our work is a followup to the recent paper of Tanabe et al. (2007), who showed a simple interval of values within which the parameter estimate should lie. Tanabe et al. (2007) actually go further than just deriving bounds on the m.l.e. of κ. They also derive a new approximation based on their bounds combined with a fixed-point approach. However, their approximation requires some additional computation, and in this note we show that given the same amount of computation as their method, one can achieve more accurate approximations using a truncated Newton method.

A more useful contribution of this paper is however, a simple algorithm (and implementation) for computing $I_s(x)$, the modified Bessel function of the first kind. Quite surprisingly, our naïve implementation turns out to be significantly faster for large arguments (that arise frequently when dealing with high-dimensional data) than standard implementations in well-established software such as MATHEMATICA©, MAPLE©, and GP/PARI.

Before we present our discussion on the approximation for κ, we first provide background on the vMF distribution in Sect. 2. Then we discuss the various approximations in Sect. 3, followed by an experimental evaluation in Sect. 4. We describe our algorithm for computing the modified Bessel function of the first kind in Sect. 5, and show several experimental results illustrating its efficiency (Sect. 5.1).

2 Background

Let $\mathbb{S}^{p-1}$ denote the p-dimensional unit hypersphere, i.e., $\mathbb{S}^{p-1} = \{\boldsymbol{x} | \boldsymbol{x} \in \mathbb{R}^p, \text{and } \|\boldsymbol{x}\|_2 = 1\}$. We denote the probability element on $\mathbb{S}^{p-1}$ by $d\mathbb{S}^{p-1}$, and parametrize $\mathbb{S}^{p-1}$ using polar coordinates $(r, \boldsymbol{\theta})$, where $r = 1$, and $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_{p-1}]$. Consequently $x_j = \sin\theta_1 \cdots \sin\theta_{p-1} \cos\theta_p$ for $1 \le j < p$, and $x_p = \sin\theta_1 \cdots \sin\theta_{p-1}$. It is easy to show that $d\mathbb{S}^{p-1} = \left(\prod_{k=2}^{p-1} \sin^{p-k}\theta_{k-1}\right) d\boldsymbol{\theta}$ (see e.g., Sra 2007, Sect. B.1).

2.1 The von Mises-Fisher density

A unit norm random vector $\boldsymbol{x}$ is said to follow the p-dimensional von Mises-Fisher (vMF) distribution if its probability element is $c_p(\kappa)e^{\kappa\boldsymbol{\mu}^T\boldsymbol{x}}d\mathbb{S}^{p-1}$, where $\|\boldsymbol{\mu}\| = 1$ and $\kappa \ge 0$. The normalizing constant for the density function is (see Sra 2007, Sect. B.4.2 for a derivation)

$$c_p(\kappa) = \frac{\kappa^{p/2-1}}{(2\pi)^{p/2} I_{p/2-1}(\kappa)},$$

where $I_s(\kappa)$ denotes the modified Bessel function of the first kind and is defined as Abramowitz and Stegun (1974):

$$I_p(\kappa) = \sum_{k\geq 0} \frac{1}{\Gamma(p+k+1)k!}\left(\frac{\kappa}{2}\right)^{2k+p},$$

where $\Gamma(\cdot)$ is the well-known Gamma function.

Note that when computing the normalizing constant, researchers in directional statistics usually normalize the integration measure by the uniform measure, so that instead of $c_p(\kappa)$ one uses $c_p(\kappa)2\pi^{p/2}/\Gamma(p/2)$; we ignore this distinction here as it does not impact parameter estimation.

The vMF density is thus

$$p(\boldsymbol{x}; \boldsymbol{\mu}, \kappa) = c_p(\kappa) e^{\kappa \boldsymbol{\mu}^T \boldsymbol{x}},$$

and it is parametrized by the mean direction $\boldsymbol{\mu}$ and the *concentration* parameter κ—so-called because it characterizes how strongly the unit vectors drawn according to $p(\boldsymbol{x}; \boldsymbol{\mu}, \kappa)$ are concentrated around the mean direction. For example, when $\kappa = 0$, $p(\boldsymbol{x}; \boldsymbol{\mu}, \kappa)$ reduces to the uniform density on $\mathbb{S}^{p-1}$, and as $\kappa \to \infty$, $p(\boldsymbol{x}; \boldsymbol{\mu}, \kappa)$ tends to a point density peaking at $\boldsymbol{\mu}$.

The vMF distribution is one of the simplest distributions for directional data, and has properties analogous to those of the multi-variate Gaussian distribution for data in $\mathbb{R}^p$. For example, the maximum entropy density on $\mathbb{S}^{p-1}$ subject to the constraint that $E[\boldsymbol{x}]$ is fixed, is a vMF density (see Mardia and Jupp 2000 for details).

2.2 Maximum-likelihood estimates

Let $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ be a set of points drawn from $p(\boldsymbol{x}; \boldsymbol{\mu}, \kappa)$. We wish to estimate $\boldsymbol{\mu}$ and κ via maximizing the log-likelihood

$$L(\mathcal{X}; \boldsymbol{\mu}, \kappa) = \log c_p(\kappa) + \sum_i \kappa \boldsymbol{\mu}^T \boldsymbol{x}_i, \tag{1}$$

subject to the condition that $\boldsymbol{\mu}^T\boldsymbol{\mu} = 1$ and $\kappa \geq 0$. Maximizing (1) subject to these constraints we find that

$$\boldsymbol{\mu} = \frac{\sum_i \boldsymbol{x}_i}{\|\sum_i \boldsymbol{x}_i\|}, \quad \kappa = A_p^{-1}(\bar{R}), \tag{2}$$

where

$$A_p(\kappa) = \frac{-c'_p(\kappa)}{c_p(\kappa)} = \frac{I_{p/2}(\kappa)}{I_{p/2-1}(\kappa)} = \frac{\|\sum_i \boldsymbol{x}_i\|}{n} = \bar{R}. \tag{3}$$

These m.l.e. equations may also be found in Banerjee et al. (2005); Dhillon and Sra (2003); Mardia and Jupp (2000). The challenge is to solve (3) for κ; the simple estimates that Mardia and Jupp (2000) provide are inaccurate for large p, or when $\kappa/p \not\ll 1$—situations that are common for high-dimensional data in modern data mining applications. Banerjee et al. (2005) provided efficient numerical estimates for κ that were obtained by truncating the continued fraction representation of $A_p(\kappa)$ and solving the resulting equation. The estimates obtained via this truncation are rough, and Banerjee et al. (2005) introduced an empirically determined correction term to yield the estimate (4), which turns out to be quite accurate in practice.

Subsequently, Tanabe et al. (2007) showed simple bounds for κ by exploiting inequalities about the Bessel ratio $A_p(\kappa)$—this ratio possesses several nice properties, and is very amenable to analytic treatment Amos (1974). The work of Tanabe et al. (2007) lent theoretical support to the empirically determined approximation of Banerjee et al. (2005), by essentially showing that their approximation lay in the "correct" range. Tanabe et al. (2007) also presented a fixed-point iteration based algorithm to compute an approximate solution κ.

In the next section we show that the approximation obtained by Tanabe et al. (2007) can be improved upon without incurring additional computational expense. We illustrate this via a series of experiments.

3 Parameter approximations

The solution to the parameter estimation equation (3) can be approximated to varying degrees of accuracy. Three simple methods are summarized below; the third one is the method proposed by this paper.

3.1 Banerjee et al. [3]

This is the simplest approximate solution of (3), and is given by

$$\hat{\kappa} = \frac{\bar{R}(p - \bar{R}^2)}{1 - \bar{R}^2}. \tag{4}$$

The *critical* difference between this approximation and the next two is that it does not involve any Bessel functions (or their ratio). That is, not a single evaluation of $A_p(\kappa)$ is needed—an advantage that can be significant in high-dimensions where it can be expensive to compute $A_p(\kappa)$. Naturally, one can try to compute $\log I_s(\kappa)$ ($s = p/2$) to avoid overflows (or underflows as the case may be), though doing so introduces yet another approximation. Therefore, when running time and simplicity are of the essence, approximation (4) is preferable.

3.2 Tanabe et al. [10]

Tanabe et al.'s approximation for κ, which was motivated by linear interpolation combined with a fixed point approach, is given by

$$\hat{\kappa} = \frac{\kappa_l \Phi_{2p}(\kappa_u) - \kappa_u \Phi_{2p}(\kappa_l)}{(\Phi_{2p}(\kappa_u) - \Phi_{2p}(\kappa_l)) - (\kappa_u - \kappa_l)}, \tag{5}$$

where the bounds on the m.l.e. $\hat{\kappa}$ are given by

$$\kappa_l = \frac{\bar{R}(p-2)}{1-\bar{R}^2} \leq \hat{\kappa} \leq \kappa_u = \frac{\bar{R}p}{1-\bar{R}^2}.$$

The function Φ in approximation (5) is defined as (we note that that there is a typo in Eqns. (34) and (35) of Tanabe et al. (2007), where they write Φ_p instead of Φ_{2p}),

$$\Phi_{2p}(\kappa) = \bar{R}\kappa A_p(\kappa)^{-1}.$$

3.3 Truncated Newton approximation

Approximation (4) can be made more exact by performing a few iterations of Newton's method. However, to remain competitive in terms of running time with (5), we perform only two-iterations of Newton's method. We make use of the fact that Mardia and Jupp (2000)

$$A'_p(\kappa) = 1 - A_p(\kappa)^2 - \frac{p-1}{\kappa} A_p(\kappa),$$

while deriving the Newton updates for solving $A_p(\kappa) - \bar{R} = 0$. We set κ_0 to the value yielded by (4), and compute the following two Newton steps

$$\begin{aligned} \kappa_1 &= \kappa_0 - \frac{A_p(\kappa_0) - \bar{R}}{1 - A_p(\kappa_0)^2 - \frac{(p-1)}{\kappa_0} A_p(\kappa_0)} \\ \kappa_2 &= \kappa_1 - \frac{A_p(\kappa_1) - \bar{R}}{1 - A_p(\kappa_1)^2 - \frac{(p-1)}{\kappa_1} A_p(\kappa_1)}. \end{aligned} \tag{6}$$

Note that just like approximation (5), the computation (6) also requires only two calls to a function that computes evaluating $A_p(\kappa)$—which entails two calls to a function that computes $I_s(\kappa)$.[1] The approximation (6) is thus competitive in running time with (5), which also requires only two calls to compute $A_p(\kappa)$. However, as our experiments show (6) is on average more accurate than (5).

[1] One can also directly compute the ratio $A_p(\kappa)$ itself to desired accuracy either by using its continued fraction expansion or otherwise, for example, using the methods of Amos (1974). However, for simplicity we compute it by making two calls to a function computing $I_s(x)$.

Remarks

1. If in an application, the cost of computing $\bar{R}$ is larger than the cost of computing $A_p(\kappa)$, then one could invoke approximation (6), otherwise the fastest approximation is (4).
2. The concerns about accuracy of the different approximations are more of an academic nature, as also noted by Tanabe et al. (2007), because in an actual application the variance in the data or the algorithm itself will easily outweigh the effects that the extra digits of accuracy can have. However, it is also obvious that given three different approximations, one would choose the most accurate one, especially if the computational costs are as low as that of a less accurate approximation.

4 Experiments for κ

Table 1 summarizes how the three different approximations for κ stand in relation to each other. In this section we show experiments that illustrate the accuracies achieved by these three approximations. We note that for all our numerical experiments both (5) and (6) used the same implementation of $A_p(\kappa)$.

In Table 2 we present numerical values for several $(p, \kappa_{\text{true}})$ pairs. Here we show all three approximations given by (4)–(6). The truncated Newton method based approximation (6) is seen to yield results superior to the fixed point interpolation (5), most of the time. From the table it is obvious that all the approximations become progressively worse as κ increases.

Figure 1 compares the approximation (5) to (6) as κ_{true} is varied from 1000 to 100,000 and the dimensionality p is held fixed at 100,000—to model a typical high-dimensional scenario. From the figure, one can see that the truncated Newton approximation (6) outperforms the fixed-point based interpolation (5) on an average.

Next, Figs. 2 and 3 show the absolute errors of approximation for a fixed value of κ_{true} as the dimensionality p is varied from 1000 to 100,000 (Fig. 2) and then from 100,000 to 200,000 (Fig. 3). We observe that in Fig. 2 the truncated Newton approximation performs much better than Tanabe et al.'s approximation, though these differences become less significant with increasing p.

From our experiments we can conclude that for most situations the truncated Newton approximation (6) yields a better approximation to κ_{true}, while incurring essentially the same computational cost as (5).

Table 1 Comparison of parameter estimating methods for vMF distributions

Method	Advantages	Disadvantages
(4)	No function evaluations; very fast	Can have lower accuracy
(5)	Higher accuracy	2 $A_p(\kappa)$ evaluations; Can be slow
(6)	Best accuracy	2 $A_p(\kappa)$ evaluations; Can be slow

These differences become important with increasing dimensionality of the data

Table 2 Errors for the different approximations of κ

$(p, \kappa_{\text{true}})$	Banerjee (4)	Tanabe et al. (5)	Newton (6)
(500, 100)	6.84e−03	1.82e−06	**1.32e−12**
(500, 500)	1.71e−01	1.99e−04	**3.04e−11**
(500, 1000)	2.96e−01	3.49e−04	**1.46e−11**
(500, 5000)	4.52e−01	4.76e−04	**3.50e−11**
(500, 10000)	4.75e−01	4.90e−04	**1.75e−08**
(500, 20000)	4.88e−01	4.97e−04	**5.68e−08**
(500, 50000)	4.95e−01	5.00e−04	**5.32e−07**
(500, 100000)	4.98e−01	5.01e−04	**2.11e−06**
(1000, 100)	9.58e−04	3.65e−08	**3.30e−12**
(1000, 500)	6.06e−02	2.59e−05	**2.95e−11**
(1000, 1000)	1.71e−01	9.93e−05	**2.08e−10**
(1000, 5000)	4.07e−01	2.23e−04	**2.22e−09**
(1000, 10000)	4.52e−01	2.37e−04	**2.29e−09**
(1000, 20000)	4.75e−01	2.44e−04	**2.73e−08**
(1000, 50000)	4.90e−01	2.48e−04	**3.94e−07**
(1000, 100000)	4.95e−01	2.48e−04	**2.03e−06**
(5000, 100)	7.98e−06	**1.38e−12**	3.66e−12
(5000, 500)	9.61e−04	7.62e−09	**1.31e−10**
(5000, 1000)	6.88e−03	1.83e−07	**6.52e−10**
(5000, 5000)	1.71e−01	1.98e−05	**5.20e−11**
(5000, 10000)	2.96e−01	3.46e−05	**2.70e−09**
(5000, 20000)	3.87e−01	4.28e−05	**2.87e−08**
(5000, 50000)	4.52e−01	4.73e−05	**9.22e−08**
(5000, 100000)	4.75e−01	4.87e−05	**7.90e−08**
(10000, 100)	9.99e−07	**5.71e−11**	5.96e−11
(10000, 500)	1.24e−04	8.60e−10	**5.15e−10**
(10000, 1000)	9.61e−04	4.62e−09	**1.57e−10**
(10000, 5000)	6.07e−02	2.59e−06	**2.32e−09**
(10000, 10000)	1.71e−01	9.90e−06	**9.35e−09**
(10000, 20000)	2.96e−01	1.73e−05	**4.63e−08**
(10000, 50000)	4.07e−01	2.22e−05	**9.33e−08**
(10000, 100000)	4.52e−01	2.40e−05	**1.22e−07**
(20000, 100)	1.25e−07	**9.95e−11**	2.08e−10
(20000, 500)	1.56e−05	1.27e−09	**7.74e−10**
(20000, 1000)	1.24e−04	1.95e−09	**1.69e−09**
(20000, 5000)	1.25e−02	1.13e−07	**9.64e−09**
(20000, 10000)	6.07e−02	1.28e−06	**1.02e−08**
(20000, 20000)	1.71e−01	4.89e−06	**1.04e−08**
(20000, 50000)	3.30e−01	9.86e−06	**4.14e−07**
(20000, 100000)	4.07e−01	1.24e−05	**1.34e−06**

Table 2 continued

$(p, \kappa_{\text{true}})$	Banerjee (4)	Tanabe et al. (5)	Newton (6)
(100000, 100)	2.84e−10	1.28e−09	**5.14e−10**
(100000, 500)	1.25e−07	**7.06e−10**	3.04e−09
(100000, 1000)	9.99e−07	**1.15e−09**	3.26e−09
(100000, 5000)	1.24e−04	5.37e−08	**3.33e−08**
(100000, 10000)	9.61e−04	8.58e−08	**4.56e−08**
(100000, 20000)	6.89e−03	1.47e−07	**4.66e−08**
(100000, 50000)	6.07e−02	3.70e−07	**8.67e−08**
(100000, 100000)	1.71e−01	1.69e−06	**2.36e−08**

We display $|\hat{\kappa} - \kappa_{\text{true}}|$
Bold values indicate best accuracy

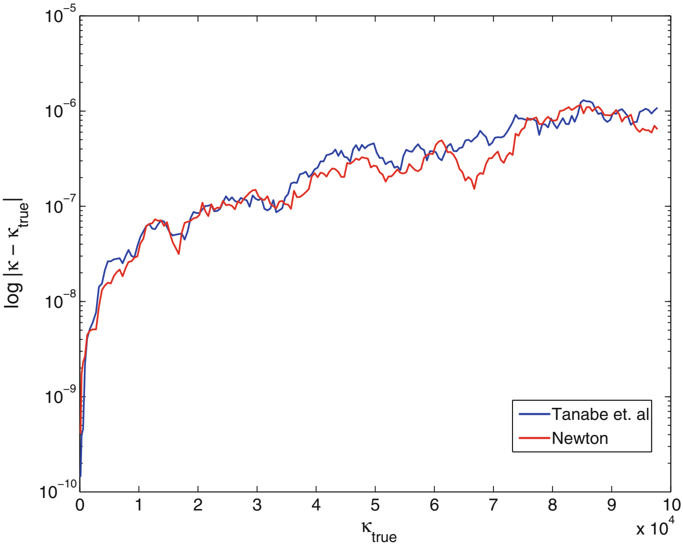

Fig. 1 Average absolute errors of approximation with varying κ and fixed $p = 100{,}000$

5 An interesting byproduct: computing $I_s(x)$

As noted in Table 1, computing approximations to κ requires evaluation of the ratio $A_p(\kappa)$. This ratio could either be computed by using its continued fraction expansion, by explicitly computing the Bessel functions and dividing, or by using more sophisticated methods (Amos 1974).

For completeness, we provide a simple algorithm below for computing modified Bessel functions of the first-kind, so that the reader can quickly try out all the approximations mentioned in this note for himself. Our particular implementation of the modified Bessel function is interesting in its own right, because surprisingly it significantly outperforms (often by several orders of magnitude) some well-established implementations in software such as MATHEMATICA©, MAPLE©, and GP/PARI [11].

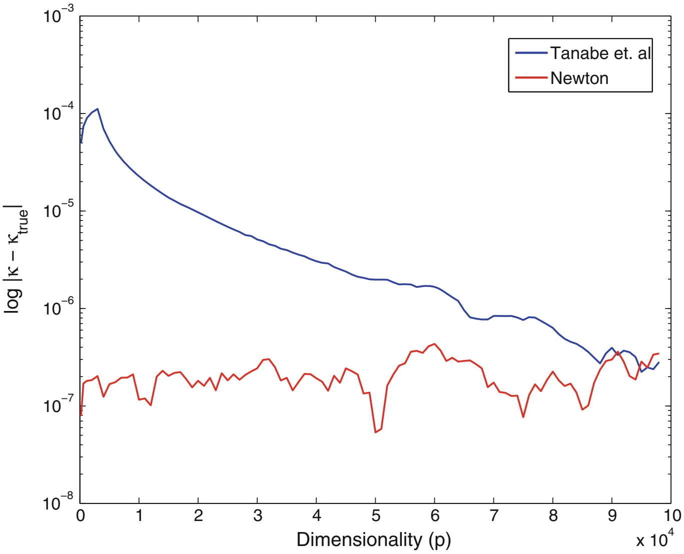

Fig. 2 Average absolute errors of approx. as p varies from 1000 to 100,000; $\kappa_{\text{true}} = 50{,}000$

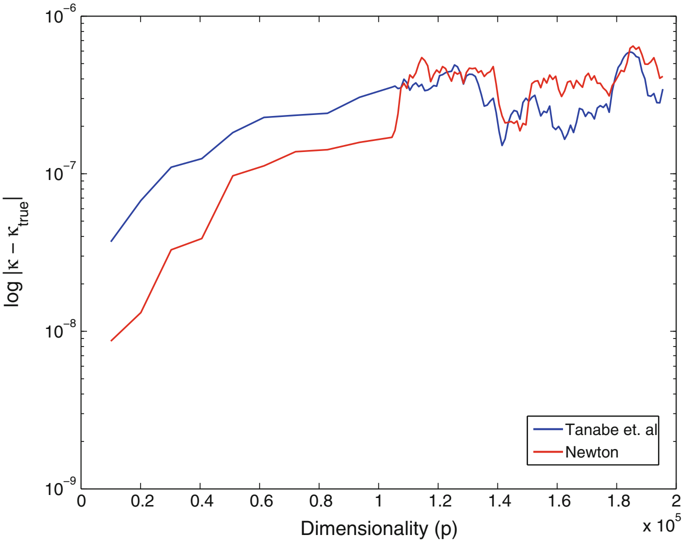

Fig. 3 Average absolute errors of approx. as p varies from 100,000 to 200,000; $\kappa_{\text{true}} = 50{,}000$

Our method should be preferred when both s and x can be large; for smaller arguments the functions available in standard software libraries should suffice. Note that previously various authors, including Tanabe et al. (2007) have suggested using an approximation to $\log I_s(x)$ instead. Indeed, one can use such an approximation, though this approximation may not be that accurate for the case where $s \sim x$ (as opposed to the commonly assumed asymptotic scenarios where $s \ll x$ or $x \ll s$).

A standard power-series representation (see Abramowitz and Stegun 1974) for the modified Bessel function of the first kind is

$$I_s(x) = (x/2)^s \sum_{k \geq 0} \frac{(x^2/4)^k}{\Gamma(k+s+1)k!}. \tag{7}$$

Using the fact that $\Gamma(x+1) = x\Gamma(x)$, we can rewrite (7) as

$$I_s(x) = \frac{(x/2)^s}{\Gamma(s)} \sum_{k \geq 0} \frac{(x^2/4)^k}{s(s+1)\cdots(s+k)k!}. \tag{8}$$

The power-series (8) is amenable to a computational procedure as the ratio of the $(k+1)$st term to the kth term is

$$\frac{x^2}{4(k+1)(s+k+1)}. \tag{9}$$

We can also use Stirling's approximation formula for the Gamma function (see Abramowitz and Stegun 1974, Sect. 6.1.37) to further speed up computation for large arguments:

$$\Gamma(x) \approx \left(\frac{x}{e}\right)^x \sqrt{\frac{2\pi}{x}} \left(1 + \frac{1}{12x} + \frac{1}{288x^2} - \frac{139}{51840x^3} + \cdots\right). \tag{10}$$

Thus we arrive at Algorithm 1 for approximating $I_s(x)$.

Algorithm 1 Computing $I_s(x)$ via truncated power-series

```
Input: $s, x$: positive real numbers, $\tau$: convergence tolerance
Output: approximation to $I_s(x)$
$R \leftarrow 1.0$,    $t_1 \leftarrow \left(\frac{xe}{2s}\right)^s$
$t_2 \leftarrow 1 + \frac{1}{12s} + \frac{1}{288s^2} - \frac{139}{51840s^3}$
$t_1 \leftarrow t_1 \sqrt{\frac{s}{2\pi}} / t_2$
$M \leftarrow 1/s, k \leftarrow 1$
while $not\ converged$ do
    $R \leftarrow R \frac{0.25x^2}{k(s+k)}$
    $M \leftarrow M + R$
    if $R/M < \tau$ then
        $converged \leftarrow true$
    end if
    $k \leftarrow k + 1$
end while
return $t_1 M$.
```

Table 3 Running times (in seconds) of different methods for computing $I_s(x)$

(s, x)	Algorithm 1	Mathematica	Maple	Gp/Pari	Relative error
(1000, 1000)	0.000	0.011	0.062	0.016	4.53×10^{-16}
(1000, 2000)	0.000	0.006	0.122	0.062	3.13×10^{-16}
(1000, 4000)	0.000	0.003	0.454	0.219	1.12×10^{-15}
(2000, 2000)	0.000	0.070	0.239	0.062	4.42×10^{-16}
(2000, 4000)	0.015	0.025	0.448	0.203	9.61×10^{-16}
(2000, 8000)	0.000	0.006	1.966	1.264	1.86×10^{-15}
(4000, 4000)	0.000	0.290	0.919	0.234	5.89×10^{-16}
(4000, 8000)	0.000	0.091	1.791	1.248	1.49×10^{-15}
(4000, 16000)	0.000	0.020	11.009	6.676	1.95×10^{-15}
(8000, 8000)	0.015	1.546	4.192	1.170	8.31×10^{-16}
(8000, 16000)	0.015	0.416	8.768	6.537	2.19×10^{-15}
(8000, 32000)	0.015	0.088	52.011	34.632	3.57×10^{-16}
(16000, 16000)	0.000	7.860	19.063	6.396	1.28×10^{-15}
(16000, 32000)	0.016	2.057	49.203	34.585	1.45×10^{-15}
(32000, 32000)	0.015	41.483	102.430	34.383	2.10×10^{-16}
(32000, 64000)	0.015	10.504	252.472	195.672	3.94×10^{-15}
(64000, 64000)	0.000	233.534	559.498	194.923	3.68×10^{-15}
(128000, 128000)	0.000	1109.67	3103.281	840.041	8.00×10^{-15}
(256000, 256000)	0.015	–	–	–	na
(512000, 512000)	0.031	–	–	–	na
(1024000, 1024000)	0.062	–	–	–	na

A '–' indicates that the computation took too long to run. The last column shows the relative error to the value computed by Mathematica, i.e., $|\kappa_1 - \kappa_2|/\kappa_2$, where κ_1 is computed by our method and κ_2 by Mathematica

5.1 Computational experiments

For our experiments we implemented Algorithm 1 using the MPFR library [8] for multi-precision floating-point computations.[2] All experiments were run on a Lenovo T61 Laptop with a core 2 duo CPU @ 2.50 GHz, and 2GB RAM, running the Windows Vista$^{\text{TM}}$ operating system. We used Mathematica version 6.0 and Maple version 12.

At this point, we would like to again stress that that we do *not* claim that our implementation to be superior across all ranges of inputs to $I_s(x)$. Certainly, when the traditional situations such as $s \ll x$ or $x \ll s$ hold, asymptotic approximations will probably perform the best, or for s and x of moderate size, standard implementations will probably be more accurate. However, for several applications, one is in the

[2] MPFR comes with a built in function to compute $\Gamma(s)$—using it increases the running time of Algorithm 1 slightly, though without significantly impacting the overall cost.

domain where $s \sim x$, i.e., s and x are of comparable size. In such a case, traditional approximations for $I_s(x)$ break down, and standard software also becomes too slow.

Table 3 shows a sample of running time experiments to illustrate the performance of our implementation. We experimented with various settings for both Mathematica and Maple, and report results that led to the fastest answers. All the timing results presented are averages over 5 to 10 runs.

From Table 3 we see that our implementation produces results that agree with Mathematica up to 15 or 16 digits of accuracy, while being obtained several orders of magnitude faster. We note that Maple was even slower than Mathematica in all our experiments and Gp/Pari was competitive with it.

Our next two experiments briefly illustrate the running time behavior of our implementation. Figure 4 plots the running time as a function of x when the argument s is held fixed. We see that in this case, the running time increases linearly with x. Figure 5 treats the alternate case where the running time is plotted as a function of s with x held fixed. One sees that the running time decreases linearly with increasing s.

6 Conclusions

In this paper we discussed parameter estimation for high-dimensional von Mises-Fisher distributions and showed that performing two steps of a Newton method leads to significantly more accurate estimates for the concentration parameter κ than the method proposed by Tanabe et al. (2007). The more interesting contribution of our work associated with computing κ is a simple method to compute the modified Bessel function of the first kind. Our simplistic implementation was seen

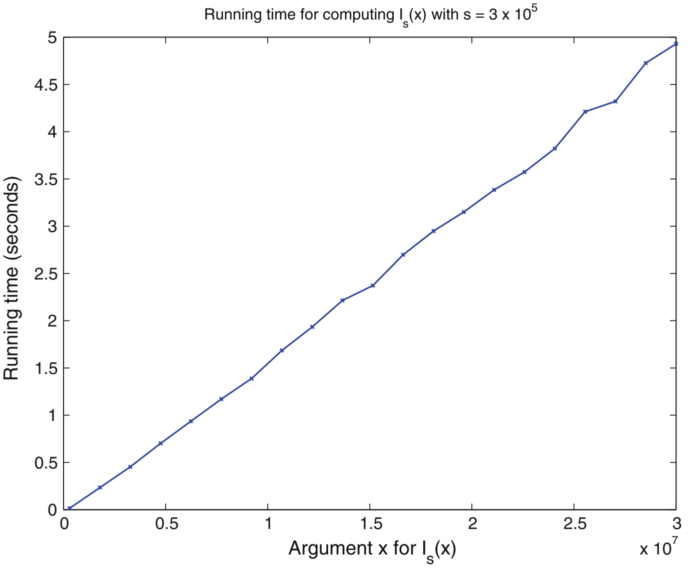

Fig. 4 Running time of Algorithm 1 as a function of x with $s = 5 \times 10^5$

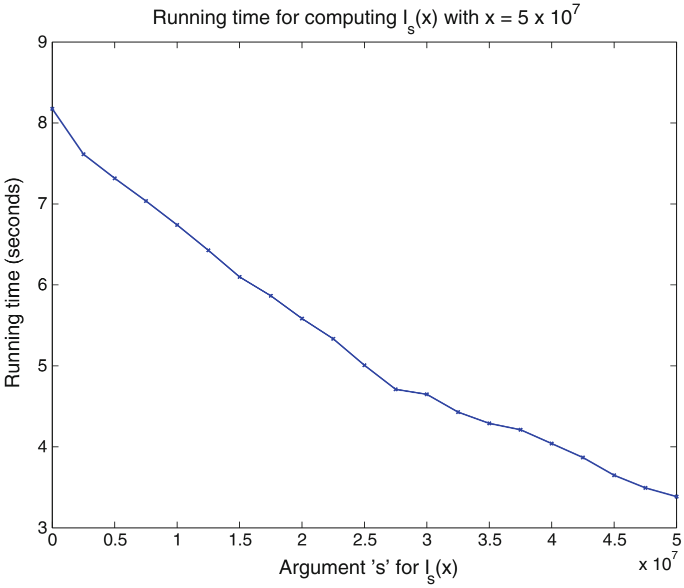

Fig. 5 Running time of Algorithm 1 as a function of s with $x = 5 \times 10^7$

to outperform standard software such as MATHEMATICA and MAPLE, sometimes by several orders of magnitude (Table 3). Our implementation can be further improved by using methods such as Aitken's process or other techniques for convergence acceleration of series Gourdon and Sebah (2002) if needed, though we have not found that necessary at this stage. On a more theoretical note, we believe that using the results of Amos (1974) one can derive even tighter bounds on the m.l.e. $\hat{\kappa}$—this is a question of purely academic interest.

Acknowledgments The author thanks the two referees whose comments helped to improve the presentation of this paper.

References

Abramowitz M, Stegun IA (eds) (1974) Handbook of mathematical functions, with formulas, graphs, and mathematical tables. Dover, New York, ISBN 0486612724

Amos DE (1974) Computation of modified Bessel functions and their ratios. Math Comput 28(125):235–251

Banerjee A, Dhillon IS, Ghosh J, Sra S (2005) Clustering on the unit hypersphere using von Mises-Fisher distributions. JMLR 6:1345–1382

Dhillon IS, Sra S (2003) Modeling data using directional distributions. Technical Report TR-03-06, Computer Sciences, The University of Texas at Austin

Gourdon X, Sebah P (2002) Convergence acceleration of series. http://numbers.computation.free.fr/Constants/constants.html

Mardia KV, Jupp P (2000) Directional statistics, second edition. Wiley, London

Maxima (2008) http://maxima.sourceforge.net. Computer Algebra System *version* 5.16.3

MPFR (2008) http://www.mpfr.org. Multi-precision floating-point library *version* 2.3.1

PARI/GP (2008) version 2.3.4. The PARI Group, Bordeaux. available from http://pari.math.u-bordeaux.fr/

Sra S (2007) Matrix nearness problems in data mining. PhD thesis, University of Texas at Austin

Tanabe A, Fukumizu K, Oba S, Takenouchi T, Ishii S (2007) Parameter estimation for von Mises-Fisher distributions. Comput Stat 22(1):145–157