J Geod (2006) 79: 689–693
DOI 10.1007/s00190-006-0023-2

ORIGINAL ARTICLE

Toshio Fukushima

Transformation from Cartesian to geodetic coordinates accelerated by Halley's method

Received: 10 December 2005 / Accepted: 09 January 2006 / Published online: 22 February 2006

Abstract By using Halley's third-order formula to find the root of a non-linear equation, we develop a new iterative procedure to solve an irrational form of the "latitude equation", the equation to determine the geodetic latitude for given Cartesian coordinates. With a limit to one iteration, starting from zero height, and minimizing the number of divisions by means of the rational form representation of Halley's formula, we obtain a new non-iterative method to transform Cartesian coordinates to geodetic ones. The new method is sufficiently precise in the sense that the maximum error of the latitude and the relative height is less than 6 micro-arcseconds for the range of height, $-10\,\mathrm{km} \le h \le 30{,}000\,\mathrm{km}$. The new method is around 50% faster than our previous method, roughly twice as fast as the well-known Bowring's method, and much faster than the recently developed methods of Borkowski, Laskowski, Lin and Wang, Jones, Pollard, and Vermeille.

Keywords Geodetic coordinate transformation · Halley's method · Latitude equation

1 Introduction

GPS and other space-geodetic techniques have enhanced the importance of the transformation between the geodetic coordinates (φ, λ, h) and Cartesian rectangular coordinates (x, y, z). For instance, some mobile phones now incorporate a GPS receiver in order to provide on-line location information to the end-users. In Japan, the Ministry of General Affairs announced in May 2004 that all third-generation mobile phones in Japan must contain a GPS receiver by April 2007. This policy is to identify the location of emergency calls from mobile phones. Since the computational resource is limited in a mobile phone and the frequency of calling the transformation is tremendously high, the development of fast, concise, and yet accurate procedures is urgently needed.

The forward transformation from geodetic to Cartesian coordinates is an easy task. However, the backward one, especially the conversion from the coordinates on a meridional plane ($p \equiv \sqrt{x^2 + y^2}, z$) to (φ, h), is a complicated problem (Heiskanen and Moritz 1967). This issue has been the target of many investigations (Bowring 1976, 1985; Borkowski 1987, 1989; Laskowski 1991; Lin and Wang 1995; Fukushima 1999; Gerdan and Deakin 1999; Pollard 2002; Jones 2002; Vermeille 2002, 2004; Zhang et al. 2005). Fok and Iz (2003) give a comparison among some of these methods.

Fukushima (1999) presented an iterative method for this problem by using Newton's second-order formula to solve a non-linear equation (Abramowitz and Stegun, 1972). It was shown numerically that it runs faster than the methods of Bowring (1976) and Borkowski (1989), and the repeated usage of Bowring's formula (Laskowski 1991). Other methods have been proposed (Lin and Wang 1995; Pollard 2002; Jones 2002; Vermeille 2002, 2004). However, all of them are slower than our previous method, as will be shown later.

In this article, we report that the usage of Halley's formula (Danby 1988, Sect. 6.6), which is known as an economic third-order method to solve a general non-linear equation numerically, coupled with a technique to avoid division operations significantly accelerates the backward transformation without degrading the precision.

2 Method

The main problem of the transformation from Cartesian to geodetic coordinates is the solution of the "latitude equation", an equation to determine the geodetic latitude or its equivalent from the given Cartesian coordinates.

For the equation to be solved, we adopt a non-dimensionalization of the irrational form of the equation (Fukushima 1999, Appendix B) with respect to the tangent of the reduced latitude, $T \equiv \tan\psi$, as

T. Fukushima
National Astronomical Observatory,
Ohsawa, Mitaka, Tokyo 181-8588, Japan
E-mail: Toshio.Fukushima@nao.ac.jp
Tel.: +81-422-34-3613
Fax: +81-422-34-3810

$$g(T) \equiv PT - Z - \frac{ET}{\sqrt{1+T^2}} = 0, \tag{1}$$

where

$$P \equiv a^{-1}p, \quad Z \equiv \left(a^{-1}e_{\mathrm{c}}\right)|z|, \quad E \equiv e^2. \tag{2}$$

Here a and e are the semi-major axis and the first eccentricity of the reference ellipsoid, respectively, and $e_{\mathrm{c}} \equiv \sqrt{1-e^2}$.

We apply Halley's third-order method (Danby 1988) to the solution of the above non-linear equation $g(T) = 0$. The resulting iterative formula becomes

$$T_{n+1} = T_n - \frac{g\left(T_n\right)}{g'\left(T_n\right) - g''\left(T_n\right) g\left(T_n\right) / \left(2g'\left(T_n\right)\right)}, \tag{3}$$

where the first and second derivatives of $g(T)$ are given as

$$g'(T) = P - \frac{E}{\left(\sqrt{1+T^2}\right)^3} \tag{4}$$

and

$$g''(T) = \frac{3ET}{\left(\sqrt{1+T^2}\right)^5}. \tag{5}$$

If we ignore the second derivative, the method reduces to Newton's second-order formula (Abramowitz and Stegun 1972)

$$T_{n+1} = T_n - \frac{g\left(T_n\right)}{g'\left(T_n\right)}. \tag{6}$$

In general, it is very time-consuming to evaluate the second derivative rigorously. This is the reason why Halley's method is not popular, although it provides a cubic rate of convergence to the solution. Fortunately $g''(T)$ is evaluated with only a small increase in the computation time. As such, Halley's method is expected to be more efficient than Newton's method in this case.

Whether Halley's or Newton's iterative formula is used, once the improved value T_{n+1} becomes sufficiently correct, the geodetic coordinates are computed from it as

$$\varphi = \mathrm{sign}(z) \tan^{-1}\left(\frac{T_{n+1}}{e_{\mathrm{c}}}\right) \tag{7}$$

and

$$h = \frac{e_{\mathrm{c}}p + |z|T_{n+1} - b\sqrt{1+T_{n+1}^2}}{\sqrt{e_{\mathrm{c}}^2 + T_{n+1}^2}}, \tag{8}$$

where $\mathrm{sign}(z)$ denotes the signature of z and $b \equiv ae_c$ is the semi-minor axis of the reference ellipsoid.

As a starter for the iterative procedure, whether it be Halley's or Newton's, we adopt the solution in the case $h = 0$,

$$T_0 = \frac{|z|}{e_{\mathrm{c}}p} = \frac{Z}{e_{\mathrm{c}}^2 P}. \tag{9}$$

Let us discuss the applicability and stability of Halley's method. First, the expression of $g''(T)$ has no pole, and therefore, it remains finite. Also, it is positive unless $T = 0$ or $T = +\infty$, the cases of which can be excluded easily before beginning the iterative procedure by inspecting the values of p and z. Thus, Halley's method is of cubic order in any case.

As is the case for Newton's method, Halley's method loses its effectiveness when $g'(T)$ happens to be zero during the iteration. Such a situation is limited to the cases when transforming the points in a tiny region near the geocenter, as illustrated in Fig. B1 of Fukushima (1999); see a similar discussion in Jones (2002).

No theory is known for the stability of Halley's method. This situation is different from the case of Newton's method, where the stability theory was established by Fukushima (1999). Regardless, our experience tells that, in practical cases such as from the deepest ocean floor to the location of infinitely high elevations, the application of both Halley's and Newton's methods work well when the zero height solution is chosen as the starter, as in the above case.

From the computational point of view, the above procedure, whether using Halley's or Newton's formula, after n iterations ($n = 1, 2, \ldots$) uses one call of `atan` and $n + 2$ calls of `sqrt`. This small number of transcendental function calls will significantly contribute to the acceleration of the procedure when compared with frequently evaluating trigonometric and/or inverse trigonometric functions during the iteration.

Unfortunately, however, the above procedure still needs a significant number of division operations: $5n + 3$ and $3n + 3$ for Halley's and Newton's formula, respectively. Typically, the CPU time for one division operation amounts to roughly a dozen multiplication operations; see Table C1 of Fukushima (1999). Therefore, in order to reduce the total CPU time, it is crucial to minimize the number of not only the special functions (e.g., trigonometric functions and/or the square root), but also the division operations.

In order to minimize the number of division operations, we express the variable T in a fractional form, $T = S/C$, and rewrite Halley's formula, which can be regarded as an iterative formula with respect to T, into a pair of iterative formulas in terms of S and C as

$$S_{n+1} = D_n F_n - B_n S_n, \tag{10}$$

$$C_{n+1} = F_n^2 - B_n C_n, \tag{11}$$

where

$$D_n = ZA_n^3 + ES_n^3, \tag{12}$$

$$F_n = PA_n^3 - EC_n^3, \tag{13}$$

$$A_n = \sqrt{S_n^2 + C_n^2}, \tag{14}$$

$$B_n = 1.5ES_nC_n^2\left[\left(PS_n - ZC_n\right)A_n - ES_nC_n\right]. \tag{15}$$

Thus, no division appears during the iteration.

The same technique can be applied to Newton's method. The resulting formulas are

$$S_{n+1} = D_n, \quad C_{n+1} = F_n. \tag{16}$$

Table 1 Comparison of methods to transform (p, z) to (φ, h)

Reference	Case A		Case B		Equation type	Variables	Method of solution	Speed of convergence
	CPU Time	$\delta_{\max}$ (milli arcsecond)	CPU Time	$\delta_{\max}$ (milli arcsecond)				
Heiskanen and Moritz (1967)	4.06	–	3.76	–	Trigonometric	(φ, h)	Fixed point	Linear
Jones (2002)	2.66	–	2.43	–	Trigonometric	ψ	Newton	Quadratic
Lin and Wang (1995)	2.61	–	1.37	–	Rational	m	Newton	Quadratic
Laskowski (1991)	2.27	–	2.05	–	Trigonometric	ψ	Newton	Quadratic
Pollard (2002)	1.99	–	1.47	–	Irrational	(h, z_0)	Fixed point	Linear
Borkowski (1989)	1.85	8.2×10^{-4}	1.85	0.37	Quartic	t	Ferrari	–
Vermeille (2002)	1.12	–	1.12	–	Quartic	k	Ferrari	–
Bowring (1976)	1.00	3.5	1.00	4.8×10^{-4}	Trigonometric	ψ	Newton, once	Quadratic
This work (a)	0.92	–	0.77	–	Irrational	T	Newton	Quadratic
Fukushima (1999)	0.81	–	0.81	–	Quartic	t	Newton	Quadratic
This work (d)	0.65	–	0.64	–	Irrational	(S, C)	Newton	Quadratic
ibid (c)	0.60	5.4×10^{-3}	0.60	–	Irrational	T	Halley, once	Cubic
ibid (b)	0.58	1.7	0.58	2.9×10^{-4}	Irrational	T	Newton, once	Quadratic
ibid (f)	0.52	5.4×10^{-3}	0.52	–	Irrational	(S, C)	Halley, once	Cubic
ibid (e)	0.49	1.7	0.49	2.9×10^{-4}	Irrational	(S, C)	Newton, once	Quadratic

The listed time is the averaged CPU time for the Intel Pentium M processor, scaled by that of Bowring (1976). Case A is the result for the range of height, $-10\,\text{km} \le h \le +30,000\,\text{km}$, while case B is for $-10\,\text{km} \le h \le +10\,\text{km}$. The transformation error is defined as $\delta \equiv |\Delta\varphi| + |\Delta h|/(a+h)$. The unit of the error is milli-arcsecond. The errors are omitted when they are less than 2 nano-arcseconds $\sim 10^{-14}$ rad

Note that the thus computed S_n and C_n are not normalized in the sense $A_n \neq 1$. If the equation to be solved is not scaled properly, the unnormalized variables, S_n and C_n, increase and/or decrease rapidly during the iteration, and therefore, an overflow or underflow will happen. This is the reason why we non-dimensionized the equation.

The adopted starter is expressed in terms of S and C as

$$S_0 = Z, \quad C_0 = e_c P. \tag{17}$$

Note that the correction factor of Halley's method, B_n, becomes simplified for these starting values as

$$B_0 = \left(1.5E^2\right) P S_0^2 C_0^2 \left(A_0 - e_c\right). \tag{18}$$

When the improved values S_{n+1} and C_{n+1} are sufficiently correct, the geodetic coordinates are computed from them as

$$\varphi = \text{sign}(z) \tan^{-1}\left(\frac{S_{n+1}}{C_c}\right) \tag{19}$$

and

$$h = \frac{pC_c + |z|S_{n+1} - bA_{n+1}}{\sqrt{C_c^2 + S_{n+1}^2}}, \tag{20}$$

where

$$C_c = e_c C_{n+1}. \tag{21}$$

The above procedure in terms of S and C, whether using Halley's or Newton's formula, after n iterations ($n = 1, 2, \ldots$) requires one call of `atan`, $n+2$ calls of `sqrt`, and two division operations. This small number of division operations will further contribute to the acceleration of the procedure.

From a practical point of view, we may limit the number of iterations to one as Bowring (1976) did. This approach accelerates the solution procedure at the cost of sacrificing the solution precision. As will be shown later, the degraded precision is a few milli-arcseconds for Newton's method and several micro-arcseconds for Halley's method.

3 Numerical comparisons

In order to compare the performance of the new method with existing methods, we conducted test conversions from (p, z) to (φ, h) for two ranges of height: from -10 to $+30,000$ km, and from -10 to $+10$ km. For the parameters of the test reference ellipsoid, a and $1/f$, we adopted those of GRS80 (Moritz 1980). See Table 1 for the results of the comparison.

The methods compared are, in historical order,

1. the method presented in Heiskanen and Moritz (1967), which solves the trigonometric equations with respect to φ and h by repeating fixed-point iterations from the zero height solution, $h_0 = 0$;
2. the method of Bowring (1976), which solves a trigonometric equation with respect to the reduced latitude

$$\psi \equiv \tan^{-1}\left(e_c \tan\,\phi\right) \tag{22}$$

by Newton's method; the number of iterations is limited to one;
3. the non-iterative method of Borkowski (1989), which solves a quartic equation with respect to

$$t \equiv \tan\left(\frac{\pi}{4} - \frac{\psi}{2}\right) \tag{23}$$

by Ferrari's method (Abramowitz and Stegun 1972);
4. the iteration of Bowring's formula twice at most (Laskowski 1991);
5. the method of Lin and Wang (1995), which solves a rational equation with respect to a scalar multiplier m,

$$\frac{p^2}{(a + 2m/a)^2} + \frac{z^2}{(b + 2m/b)^2} = 1 \tag{24}$$

by Newton's method from the zero height solution;

6. our previous method (Fukushima 1999), which solves a variation of Borkowski's quartic equation by Newton's method from a set of stable starters;
7. the first vector iteration method of Pollard (2002), which solves a pair of irrational equations with respect to h and z_0, the z-coordinate of the foot on the reference ellipsoid, by a fixed-point iteration starting from the zero height solution;
8. the method of Jones (2002), which solves a trigonometric equation with respect to the reduced latitude by Newton's method from a set of three stable starters;
9. the non-iterative method of Vermeille (2002), which solves a quartic equation with respect to
$$k \equiv \frac{h}{N} + e_c^2 \tag{25}$$
by Ferrari's method; and
10. the six variations of the new method to solve the irrational latitude equation; namely the methods using
   (a) Newton's method with respect to T,
   (b) Newton's method with respect to T, where the number of iterations is limited to one,
   (c) Halley's method with respect to T, where the number of iterations is limited to one,
   (d) Newton's method in terms of S and C,
   (e) Newton's method in terms of S and C, where the number of iterations is limited to one, and
   (f) Halley's iteration in terms of S and C, where the number of iterations is limited to one.

In Table 1, the items compared are (i) the averaged CPU time for an Intel Pentium M processor, (ii) the maximum of the transformation error defined as

$$\delta \equiv |\Delta\varphi| + |\Delta h|/(a + h), \tag{26}$$

(iii) the type of equation, (iv) the choice of variables, (v) the method of solution, and (vi) the speed of convergence.

The numerical values were obtained from timing measurements in the double-precision environment for two sets of a few 10^9 pairs of equally spaced grid points in (φ, h). We designed case A to deal with almost all possible cases; namely the latitude in the range $0 \le \varphi \le 90°$ at every 10 arcseconds and the height in the range $-10\,\text{km} \le h \le 30,000\,\text{km}$ at every 1 km. Case B is for the ground-based transformations and the grid separation is 10 m; namely in the range $0 \le \varphi \le 90°$ at every 0.309 arcsecond and in the range $-10\,\text{km} \le h \le +10\,\text{km}$ at every 10 m. The programming language used was Compaq Visual Fortran 6.6B. The CPU times shown are scaled by the measured value of Bowring (1976). The CPU time used for preparing the constants such as $e_c \equiv \sqrt{1 - e^2}$ is excluded from the timing tests. On the other hand, the CPU time used for selecting the starter and its computation was included. We omit the errors when they are less than 2 nano-arcseconds $\sim 10^{-14}$ rad.

In Table 1, the methods are listed in the order of CPU time for case A. Some iterative methods, namely the methods of Heiskanen and Moritz (1967), Laskowski (1991), Lin and Wang (1995), Pollard (2002), become significantly faster for case B. This is because they start from the initial guess of zero height. The methods of Jones (2002) and Fukushima (1999) are not significantly accelerated since their starters are not accurate when $|h|$ is small.

First, all six variations of the new method, (a) through (f), are faster than the existing methods tested, except our previous method in case A. By limiting the number of iterations to one, the variations (b) and (c) require less CPU time than variation (a). Since the iterative procedure of Newton's method is simpler than that of Halley's method, variation (b) is slower than (c), although the difference is small. On the other hand, by avoiding the division operation as much as possible, the variations (d) through (f) run faster than the corresponding other variations (a) through (c). From these reasons, variation (e) becomes the fastest. Variation (f) is the next fastest.

Next, let us discuss the other methods. In general, the methods solving trigonometric equations are the slower. This is because the evaluation of trigonometric functions is quite time-consuming, say roughly 30–40 times slower than a multiplication (Fukushima, 1999, Table B1). Therefore, application of the technique to minimize the number of division operations is not so effective for the methods calling trigonometric functions frequently, since the amount of CPU time saved is much smaller than the total CPU time.

The method of Heiskanen and Moritz (1967) is the slowest. This is partly because it solves trigonometric equations – and therefore, each iteration is time-consuming – and partly because it uses the fixed-point iteration – and therefore, the speed of convergence is as slow as being linear. Since the starter is the solution for zero height, the method will require many iterations when the height is large.

The method of Jones (2002) is essentially the same as our previous method (Fukushima 1999). However, the difference in the type of equation to be solved led to a large difference in the CPU time.

The method of Lin and Wang (1995) is slow because it solves a rational equation requiring a lot of division operations, which are not necessary if the equation is changed into a quartic polynomial form. The comparison with the result of our previous method Fukushima (1999), which solves a quartic equation by Newton's method, indicates that the difference in speed of the evaluation between a rational function and a polynomial caused this large difference in the CPU time.

The method of Pollard (2002) uses a fixed-point iteration. As such, its speed of convergence is linear, and therefore it is slow in principle. However, this weak point is partially compensated by the simpleness of the irrational form of the latitude equation it solves.

The comparison of CPU time among Borkowski (1989), Vermeille (2002), and Fukushima (1999) shows that the repeated application of Newton's formula from a suitable starter is much faster than the single application of Ferrari's method in solving quartic equations derived from the latitude equation. This is because Ferrari's method requires several calls of square root and two calls of cubic root, the latter of which is, in general, realized by further calling a pair of the exponential

and logarithmic functions each time, and therefore, is very time-consuming.

The method of Bowring (1976) is fast because it applies Newton's method only once for the sake of speed. Therefore, its precision is not as high as of the order of milli-arcseconds.

In principle, the methods solving quartic equations must be faster than those solving irrational or trigonometric equations, since the square root and division operations run much slower than the evaluation of low-order polynomials by Horner's method (Abramowitz and Stegun 1972). This is the reason why our previous method (Fukushima 1999) is faster than variation (a) and all the methods using trigonometric functions (Jones 2002; Laskowski 1991; Bowring 1976).

Concerning the precision, the procedures are good to the level of nano-arcseconds, except the methods of Bowring (1976), Borkowski (1989), and variations (b) and (e) of the new method. Also, the variations (c) and (f) in case A are not so accurate, where the maximum relative error is of the order of micro-arcseconds. The method of Borkowski (1989) is imprecise near the poles because of an inappropriate definition of coefficients of his quartic equation (Fukushima 1999). A suitable choice of coefficients, as in the method of Vermeille (2002), leads to negligibly small errors. The reason for the inaccuracy of the other five cases is that they adopted a policy to truncate the iteration for the sake of computational speed.

Summarizing the above, variation (e) of the new method is the best choice in case B because it is the fastest and the maximum error is sufficiently small (less than a micro-arcsecond). In case A, if the error tolerance is larger than a few milli-arcseconds, again variation (e) is the best. On the other hand, if a higher precision is required, variation (f) is the best method.

Since the CPU times of variations (e) and (f) are practically the same, while their precision is only significantly different by around three digits, we recommend variation (f) as the method of best performance.

4 Conclusion

We have presented a new method to rapidly compute geodetic coordinates from Cartesian coordinates. The method solves an irrational form of the latitude equation with respect to the tangent of the reduced latitude, either by a single application of Halley's third-order formula or by one or at most two iterations of Newton's second-order formula from the solution of zero height approximation. This approach is further accelerated by transforming the equation in a fractional form and solving it with respect to a pair of unnormalized sine and cosine of the reduced latitude.

The fastest variation (e) of the new method, solving the equation in fractional form by Newton's method once, runs around 50% faster than our previous method (Fukushima 1999) and roughly twice as fast as the method of Bowring (1976). Although the solution obtained by the new method is of approximate nature, it is sufficiently precise in the sense that the maximum error in the computed latitude and relative height of the recommended variation (f) of the new method, solving the equation in fractional form by Halley's method once, is less than 6 micro-arcseconds for a very wide range of height from -10 up to $+30,000$ km.

FORTRAN code of the fastest and recommended variations of the new method, (e) and (f), are available in the form of ESM accompanying this paper.

References

Abramowitz M, Stegun IA (1972) Handbook of mathematical functions with formulas, graphs, and mathematical tables, 9th edn., Dover, New York

Borkowski KM (1987) Transformation of geocentric to geodetic coordinates without approximations. Astrophy Space Sci 139:1–4

Borkowski KM (1989) Accurate algorithms to transform geocentric to geodetic coordinates. Bull Geod 63:50–56

Bowring BR (1976) Transformation from spatial to geographical coordinates. Surv Rev 23(181):323–327

Bowring BR (1985) The accuracy of geodetic latitude and height equations. Surv Rev 28(218):202–206

Danby JMA (1988) Fundamentals of celestial mechanics. Willmann-Bell, Richmond

Fok HS, Iz HB (2003) A comparative analysis of the performance of iterative and non-iterative solutions to the Cartesian to geodetic coordinate transformation. J Geospatial Eng 5:61–74

Fukushima T (1999) Fast transform from geocentric to geodetic coordinates. J Geod 73:603–610

Gerdan GP, Deakin RE (1999) Transforming Cartesian coordinates to geographical coordinates. Aust Surv 44:55–63

Heiskanen WA, Moritz H (1967) Physical geodesy. Freeman, San Francisco

Jones GC (2002) New solutions for the geodetic coordinate transformation. J Geod 76:437–446

Laskowski P (1991) Is Newton's iteration faster than simple iteration for transformation between geocentric and geodetic coordinates? Bull Geod 65:14–17

Lin KC, Wang J (1995) Transformation from geocentric to geodetic coordinates using Newton's iteration. Bull Geod 69:14–17

Moritz H (1980) Geodetic reference system 1980. Bull Geod 54(4):594–505

Pollard J (2002) Iterative vector methods for computing geodetic latitude and height from rectangular coordinates. J Geod 76:36–40

Vermeille H (2002) Direct transformation from geocentric coordinates to geodetic coordinates. J Geod 76:451–454

Vermeille H (2004) Computing geodetic coordinates from geocentric coordinates. J Geod 78:94–95

Zhang C-D, Hsu HT, Wu XP, Li SS, Wang QB, Chai HZ, Du L (2005) An alternative algebraic algorithm to convert Cartesian to geodetic coordinates. J Geod 79:413–420. DOI 10.1007/s00190-005-0487-5