EFFICIENT TRANSFORMATION FROM CARTESIAN TO GEODETIC COORDINATES

S.J. Claessens

The Institute for Geoscience Research
School of Earth and Planetary Sciences
Curtin University
GPO Box U1987, Perth, WA 6845, Australia
Phone: +61 8 9266 3505; Email: s.claessens@curtin.edu.au

ABSTRACT

The derivation of algorithms for the computation of geodetic coordinates from 3D Cartesian coordinates has been a very active field of research among geodesists for more than forty years. Many authors have sought the most efficient method, i.e. the method that provides the fastest computational speed, which nevertheless yields sufficient accuracy for practical applications. The problem is a special case of a more general mathematical problem that has also been studied by researchers in other fields. This paper investigates the applicability of methods by Sampson (1982, *Computer graphics and image processing*, 18: 97-108) and Uteshev and Goncharova (2018, *Journal of Computational and Applied Mathematics*, 328: 232-251) to the computation of geodetic coordinates. Both methods have been modified to make them more suitable for this particular problem. The methods are compared to several commonly used geodetic methods in terms of accuracy and computational efficiency. It is found that a simple modification improves the accuracy of the methods by ~3 orders of magnitude, and the modified method of Uteshev and Goncharova

(2018) achieves an accuracy of <0.1 mm anywhere on the surface of the Earth. The methods are especially efficient in the computation of ellipsoidal height. As an additional result of this study, a new formulation of the well-known method by Bowring (1976, *Survey Review*, 23: 323-327) is derived, and it is shown to improve the computation speed of Bowring's method by ~12% to ~27% compared to the conventional formulation.

Key words: Coordinate Transformation, Geodetic Coordinates, Cartesian Coordinates

1. INTRODUCTION

The transformation from 3D Cartesian coordinates (X, Y, Z) to geodetic coordinates (geodetic latitude ϕ, longitude λ, and ellipsoidal height h) is a classical problem in geodesy and its application is extremely common. While the computation of longitude is straightforward, the computation of geodetic latitude and ellipsoidal height is more complicated. Many different methods have been published in the geodetic literature. An overview of many of these methods can be found in (Featherstone and Claessens 2008), and many more have been published since (e.g., Turner 2009, Shu and Li 2010, Civicioglu 2012, Ligas 2012, Soler et al. 2012, Zeng 2013). Most methods focus on the computation of geodetic latitude, after which the ellipsoidal height can readily be found, but it is equally possible to solve for the ellipsoidal height first and geodetic latitude second.

Methods for the computation of geodetic coordinates from Cartesian coordinates can be divided into three categories: exact, iterative and approximate methods. Here we define an approximate method as any method that is neither exact nor uses a variable number of iterations. For example, Bowring's (1976) method is iterative, but when implemented such that only a single iteration is used (as is often the case), we consider it an approximate method.

An exact solution involves the solution of a quartic equation (fourth-order polynomial) (e.g. Paul 1973, Borkowski 1989, Vermeille 2004, 2011), which inevitably leads to a computationally inefficient algorithm. Geodesists have put much effort into devising more efficient iterative or approximate methods. Some of the simplest and most efficient of these are the methods by Bowring (1976, 1985) and Fukushima (1999, 2006).

In other fields, similar problems have been tackled in parallel. For example, in the field of computer vision, a common problem is the estimation of conic sections through scattered data points. To estimate a best fitting ellipse (in the case that the conic section is an ellipse), an approximation of the distance between a point and the ellipse is required. A well-known algorithm for this problem is provided by Sampson (1982), and the approximate distance has become known as Sampson's distance. Meanwhile, mathematicians have worked on more general problems, such as computation of the shortest distance between a point and any degree 2 curve or manifold in $\mathbb{R}^n$. For example, Uteshev and Yashina (2015) provide a method for finding the distance between an ellipsoid and any first- or second-order manifold. Explicit exact and approximate formulas for the distance between a point and an ellipse are provided in Uteshev and Goncharova (2018).

The main aim of this paper is to investigate the applicability of approximate solutions by Sampson (1982) and Uteshev and Goncharova (2018), from outside of the geodetic literature, to the computation of geodetic coordinates on or near Earth. These methods are then compared to a selection of geodetic methods in terms of accuracy and computational efficiency. The focus is on simple and efficient (fast) algorithms for the computation of geodetic coordinates that are precise enough for any practical application on the Earth's surface or at flight altitude.

The geodetic transformation problem is briefly defined in section 2. In section 3, Sampson's and Uteshev's methods are outlined. It will be shown that these methods are not sufficiently accurate for geodetic applications, except for points very close to the reference ellipsoid. However, new modifications to these methods to make them more suited to the geodetic coordinate transformation are presented in section 4. In section 5, the geodetic methods of Bowring (1976, 1985), Pollard (2002), and Fukushima (2006) are outlined. The accuracy of the unmodified and modified methods of Sampson (1982) and Uteshev and Goncharova (2018) are compared to these geodetic methods in section 6, and in section 7 a comparison in terms of computational efficiency is provided. An important point is made about the variability in computational efficiency for different hardware, software and implementation. Finally, section 8 provides conclusions and recommendations.

2. THE GEODETIC TRANSFORMATION PROBLEM

The geodetic transformation problem consists of the transformation between geodetic coordinates (ϕ, λ, h) and geocentric Cartesian coordinates (X, Y, Z). The forward transformation $((\phi, \lambda, h) \rightarrow (X, Y, Z))$ defines the relation between these coordinates (e.g. Heiskanen and Moritz 1967)

$$
\begin{aligned}
X &= (N + h)\cos\phi\cos\lambda \\
Y &= (N + h)\cos\phi\sin\lambda \\
Z &= [N(1 - e^2) + h]\sin\phi
\end{aligned}
\tag{1}
$$

where

$$N = \frac{a}{\sqrt{1 - e^2\sin^2\phi}} \tag{2}$$

$$e^2 = \frac{a^2 - b^2}{a^2} \tag{3}$$

and a and b are the semi-major and semi-minor axes of the reference ellipsoid, respectively. The reference ellipsoid is an oblate spheroid (ellipsoid of revolution).

In the inverse problem ($(X, Y, Z) \rightarrow (\phi, \lambda, h)$), it follows directly from Eq. (1) that longitude can be computed from the X- and Y-coordinates in a straightforward manner (e.g., Bomford 1971)

$$\lambda = \arctan\frac{Y}{X} = 2\arctan\frac{Y}{X + W} \tag{4}$$

where

$$W = \sqrt{X^2 + Y^2} \tag{5}$$

The form on the right-hand side of Eq. (4) is often used for reasons of numerical stability.

Upon the computation of λ, the inverse problem is reduced to a problem in $\mathbb{R}^2$, more specifically a problem in the WZ-plane ($(W, Z) \rightarrow (\phi, h)$). The section of the reference ellipsoid and the WZ-plane is an ellipse. The geodetic latitude ϕ can be interpreted geometrically as the angle between the W-axis and the normal to the ellipse through the point with coordinates (W, Z), and the ellipsoidal height h as the shortest distance between the point with coordinates (W, Z) and the ellipse.

3. SAMPSON'S AND UTESHEV'S METHODS

The inverse geodetic transformation problem can be solved in an approximate fashion by applying Sampson's distance formula (Sampson 1982). Sampson's distance is often thought of as a first-order approximation of the distance from a point to a curve, but to be more exact, it is the exact geometric distance from a point to the first-order approximation of the curve (Harker and O'Leary 2006).

Sampson's method is defined for the distance between a point and any curve of degree 2, which is given by the equation

$$Q(w, z) = Aw^2 + Bwz + Cz^2 + Dw + Ez + F = 0 \tag{6}$$

where A, B, C, D, E and F are constants. Sampson (1982) approximates the shortest distance between a point with coordinates (W, Z) and the curve $Q(w, z)$ by

$$d \approx \frac{Q(W,Z)}{|\nabla Q(W,Z)|} \tag{8}$$

where $\nabla Q(W,Z)$ is the magnitude of the norm of the gradient of $Q(W,Z)$ at the point (W,Z), defined by

$$|\nabla Q(W,Z)|^2 = (2AW + BZ + D)^2 + (2CZ + BW + E)^2 \tag{8}$$

In the geodetic transformation problem, the curve is an ellipse, and the distance to the curve d is the height of the computation point h. The ellipse is defined by the implicit equation

$$G(w,z) = \frac{w^2}{a^2} + \frac{z^2}{b^2} - 1 = 0 \tag{9}$$

and is thus a special case of the curve $Q(w, z)$ with

$$A = \frac{1}{a^2}, \;\; C = \frac{1}{b^2}, \;\; F = -1 \;\; \text{and} \;\; B = D = E = 0 \tag{10}$$

The magnitude of the norm of the gradient for the case of the ellipse is then

$$|\nabla Q(W,Z)|^2 = 4\left(\frac{W^2}{a^4} + \frac{Z^2}{b^4}\right) \equiv 4S_4 \tag{11}$$

We can therefore write Sampson's method for the inverse geodetic transformation problem as

$$h_S = \frac{G(W,Z)}{2\sqrt{S_4}} \tag{12}$$

Equation (12) provides an approximation of the ellipsoidal height, and the subscript S indicates that this is the ellipsoidal height according to Sampson's formula. Once the ellipsoidal height is known, the geodetic latitude ϕ can also be computed, but Sampson's method is not concerned with latitude. We will return to the computation of latitude at the end of this section.

Another approximate method for the inverse geodetic transformation problem is herein called Uteshev's method. Uteshev and Yashina (2015) showed that the squared distance h^2 between a point and the ellipse is one of the positive zeros of the *distance equation*

$$\mathcal{F}(h, W, Z) = D_{\mu}\left\{h^2\mu^3 - \frac{A_2}{a^2b^2}\mu^2 - \frac{A_1}{a^2b^2}\mu - \frac{1}{a^2b^2}\right\} \tag{13}$$

where $D_{\mu}\{.\}$ indicates the discriminant of the function and

$$\begin{aligned} A_1 &= W^2 + Z^2 - h^2 - a^2 - b^2 \\ A_2 &= a^2b^2\left\{\left(\frac{1}{a^2} + \frac{1}{b^2}\right)h^2 - G(W, Z)\right\} \end{aligned} \tag{14}$$

Uteshev and Goncharova (2018) approximate the relevant zero of this equation by a power series of the form

$$\ell_1 G(W, Z) + \ell_2 G^2(W, Z) + \ell_3 G^3(W, Z) + \cdots \tag{15}$$

where the coefficients ℓ_1, ℓ_2, and ℓ_3 are coefficients that can be determined exactly as a function of a, b, W and Z. They show that, when this power series is truncated after the quadratic term, the resulting formula for ellipsoidal height h is Sampson's formula (Eq. 12). When the cubic term in Eq. (15) is also taken into account, a more precise approximation is found

$$h_U = h_S\sqrt{1 + \frac{S_6}{2S_4^2}G(W, Z)} \tag{16}$$

where the subscript U indicates this is Uteshev's formula for ellipsoidal height, and

$$S_6 = \frac{W^2}{a^6} + \frac{Z^2}{b^6} \tag{17}$$

Uteshev and Goncharova (2018) also provide elegant formulas for the coordinates of the point on the ellipse nearest to the computation point, i.e. the point with the same geodetic latitude as the computation point and an ellipsoidal height of zero

$$W_0 = \frac{a^2W}{a^2 - \mu_*} \quad \text{and} \quad Z_0 = \frac{b^2Z}{b^2 - \mu_*} \tag{18}$$

where

$$\mu_* = \frac{-9a^2b^2h^2 - A_1A_2}{2(A_1^2 - 3A_2)} \quad (19)$$

While Uteshev and Goncharova (2018) do not mention it, once W_0 and Z_0 are known, the geodetic latitude ϕ can be found through

$$\phi = \arctan\frac{Z - Z_0}{W - W_0} = \arctan\frac{(a^2 - \mu_*)Z}{(b^2 - \mu_*)W} \quad (20)$$

This method for the computation of geodetic latitude is exact if the ellipsoidal height h is known exactly, and will provide an approximate geodetic latitude if h_U (Eq. 16) or h_S (Eq. 12) are used instead.

4. MODIFIED SAMPSON'S AND UTESHEV'S METHODS

Sampson's and Uteshev's methods have been created for general curves of degree 2 and not specifically for the inverse geodetic transformation problem. This means these methods have a disadvantage when compared to approximate methods derived specifically for geodetic purposes, which typically make use of the fact that the Earth's reference ellipsoid has only a small eccentricity.

A crucial insight is that Sampson's and Uteshev's methods are not exact when the curve is a circle, and can therefore not be expected to perform well in the inverse geodetic transformation. The height of a point above a circle with radius R and centre in the origin of the coordinate system is easily derived as

$$h = r - R \quad (21)$$

where r is the distance from the point to the origin of the coordinate system

$$r = \sqrt{W^2 + Z^2} \quad (22)$$

It can easily be seen that Sampson's method is not exact when the distance to a circle is sought, by comparing the result for h_S (Eq. 12) for the case $a = b = R$ to Eq. (21). Sampson's method for the case of a circle gives

$$h_S(\text{circle}) = \frac{\frac{r^2}{R^2} - 1}{2\sqrt{\frac{r^2}{R^4}}} = \frac{r^2 - R^2}{2r} \tag{23}$$

The error of Sampson's method for the case of a circle is therefore

$$\epsilon_S = \frac{r^2 - R^2}{2r} - (r - R) = -\frac{h^2}{2r} \tag{24}$$

where use was made of the substitution $R = r - h$ from Eq. (21). This suggests that Sampson's method can be improved for the case of a near-circular ellipse by applying a simple correction, which leads us to suggest the following solution for ellipsoidal height:

$$h_{MS} = h_S + \frac{h_S^2}{2r} \tag{25}$$

where the subscript MS stands for *Modified Sampson*. Thanks to the correction, Eq. (25) is exact when the curve is a circle, and expectedly a good approximation of the true height when the curve is an ellipse with small eccentricity. The accuracy of both the modified and unmodified methods is examined in section 6.

Uteshev's method can be modified in the same way. For the case of a circle ($a = b = R$), Uteshev's method (Eq. 16) gives

$$h_U(\text{circle}) = \frac{(r^2 - R^2)\sqrt{6r^2 - 2R^2}}{4r^2} \tag{26}$$

The error of Uteshev's method for the case of a circle is therefore

$$\epsilon_U = \frac{(r^2 - R^2)\sqrt{6r^2 - 2R^2}}{4r^2} - (r - R) = \frac{h(2r - h)\sqrt{(2r + h)^2 - 3h^2}}{4r^2} - h \tag{27}$$

This equation is not as elegant as the equivalent in Sampson's method (Eq. 24), but it can be simplified considerably for the case $|h| \ll r$ by a series of approximations. First, we apply a Taylor series expansion to the square root in Eq. (27)

$$\epsilon_U \approx \frac{h(2r-h)\left\{(2r+h) - \frac{3h^2}{2(2r+h)}\right\}}{4r^2} - h \tag{28}$$

Since the second term within the curly brackets is very small compared to the first term, we can safely approximate $(2r+h)$ in the denominator by $2r$

$$\epsilon_U \approx \frac{h(2r-h)\left\{(2r+h) - \frac{3h^2}{4r}\right\}}{4r^2} - h = \frac{-\frac{5}{2}h^3 + \frac{3}{4r}h^4}{4r^2} \tag{29}$$

Finally, the second term in the numerator on the right-hand side of Eq. (29) is much smaller than the first term for the case $|h| \ll r$, so if this term is ignored, ϵ_U is approximated by

$$\epsilon_U \approx -\frac{5h^3}{8r^2} \tag{30}$$

The error due to the approximations introduced here is quantified in section 6. The modified Uteshev method reads

$$h_{MU} = h_U + \frac{5h_U^3}{8r^2} \tag{31}$$

where the subscript MU stands for *Modified Uteshev*. The geodetic latitude can then be found using Eq. (20) with h_{MU} inserted for h in Eqs. (14) and (19).

5. GEODETIC METHODS

As mentioned in the introduction, geodesists have derived a large number of algorithms for the computation of geodetic coordinates. Here, some of the most efficient approximate methods are listed. These serve as a benchmark for the numerical performance of Sampson's and Uteshev's methods.

Since in most geodetic applications the points of interest are situated in the proximity of the surface of the reference ellipsoid (*h=0*), approximate transformation methods must provide high accuracy in this region. Arguably the simplest transformation formula follows from the exact relation between the geodetic latitude ϕ and spherical latitude θ on the ellipsoid (e.g., Laskowski 1991)

$$\tan\phi = \frac{a^2}{b^2}\tan\theta \qquad (32)$$

It follows from the definition of the spherical coordinates (e.g., Paul 1973) that $\tan\theta = Z/W$, and the geodetic latitude of points on the ellipsoid can thus be computed from

$$\phi = \mathrm{atan}\frac{a^2 Z}{b^2 W} \qquad (33)$$

which can be used as an approximate solution for points outside the ellipsoid and is here called the *spherical method*. Equation (33) is used as a starting point for many iterative solutions (e.g., Bomford 1971, Borkowski 1989). Due to its simplicity, the spherical method is the most efficient transformation method, but its accuracy for points away from the ellipsoidal surface is poor.

A more accurate transformation is the well-known method of Bowring (1976). In Bowring's method, geodetic latitude is computed through the approximate expression

$$\phi = \arctan\frac{Z + e'^2 b\sin^3 u}{W - e^2 a\cos^3 u} \qquad (34)$$

where

$$u = \arctan\frac{aZ}{bW} \qquad (35)$$

and e'^2 is the second numerical eccentricity of the ellipse. Once latitude is known, the ellipsoidal height can be computed through (Bowring 1985)

$$h = W\cos\phi + Z\sin\phi - \frac{a^2}{N} \qquad (36)$$

Two other geodetic methods have been selected for comparison. The method of Fukushima (2006), variation (f), has been selected, as it is one of the most efficient geodetic methods known (e.g. Zeng 2013). Fukushima's method has, for example, been implemented in the International Earth Rotation and Reference Systems Service (IERS) Conventions software collection. The first method of Pollard (2002) has also been selected for comparison, because it is one of the few geodetic methods in which, like in Sampson's and Uteshev's methods, ellipsoidal height is computed first, and geodetic latitude second based on the ellipsoidal height. Both Fukushima (2006) and Pollard (2002) state that their methods have a faster computation speed than Bowring's (1976) method. The algorithms of both methods can be found in Appendix A.

6. ACCURACY OF METHODS

The accuracy of Sampson's and Uteshev's methods, in unmodified and modified form, is compared here to the geodetic methods by Bowring (1976, 1985), Pollard (2002) and Fukushima (2006). These geodetic methods have been selected as they are among the simplest and computationally most efficient of the geodetic methods.

The main aim of this section is to test the methods for use on or near the Earth's surface. A numerical closed-loop experiment is conducted for heights in the range from -11,000 m to +15,000 m and latitudes from the equator to the North Pole. Results on the southern hemisphere are identical but with opposite sign, and are therefore not shown. A regular, equidistant grid of geodetic latitudes and heights was created with a resolution of 10′ in latitude and 50 m in height. This grid was then transformed to geocentric Cartesian coordinates using Eq. (1) and the parameters of the GRS80 reference ellipsoid (Moritz 2000). Subsequently, the geocentric Cartesian coordinates were transformed back into geodetic coordinates using the various methods. Longitude does not significantly affect the accuracy of the recovered latitude and height (λ was set to 0°).

Approximation errors are the differences between the original and transformed geodetic coordinates. For the latitudes, the approximation error in radians was converted to an equivalent approximation error in metres through multiplication by the distance to the origin (r). In order to properly assess the approximation error in each method without the influence of numerical rounding errors, extended precision arithmetic (variable precision arithmetic) was used. The results are shown visually in Figures 1 and 2, and the maximum error for each method in the test area is shown in Table 1.

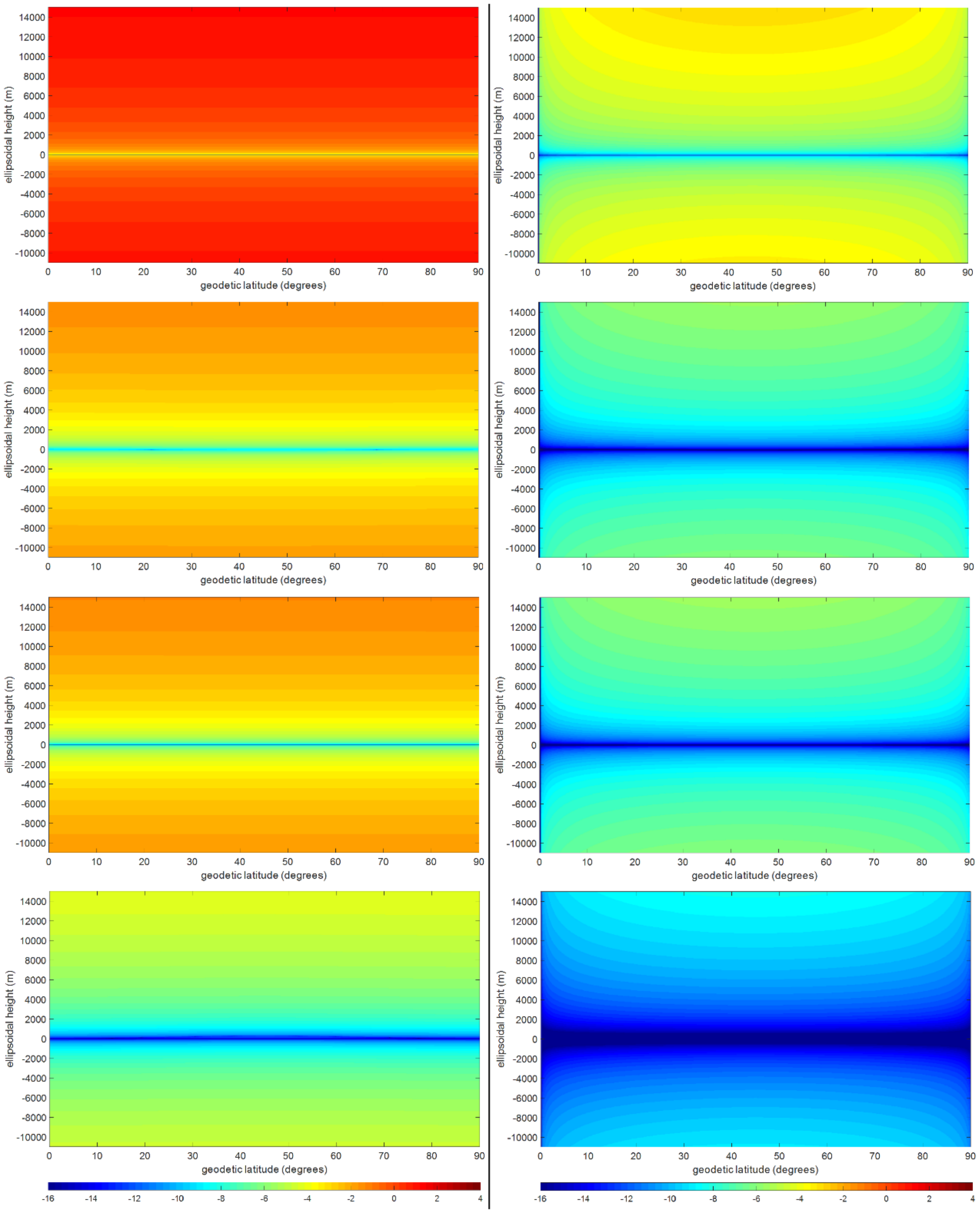

Figure 1. Approximation error in the computation of ellipsoidal height (left) and geodetic latitude (right) using the unmodified Sampson method (top row), modified Sampson method (second row), unmodified Uteshev method (third row), and modified Uteshev method (bottom row) (units: m; scale bars show the logarithm of the error; errors in latitude were converted from radians to metres through multiplication by r)

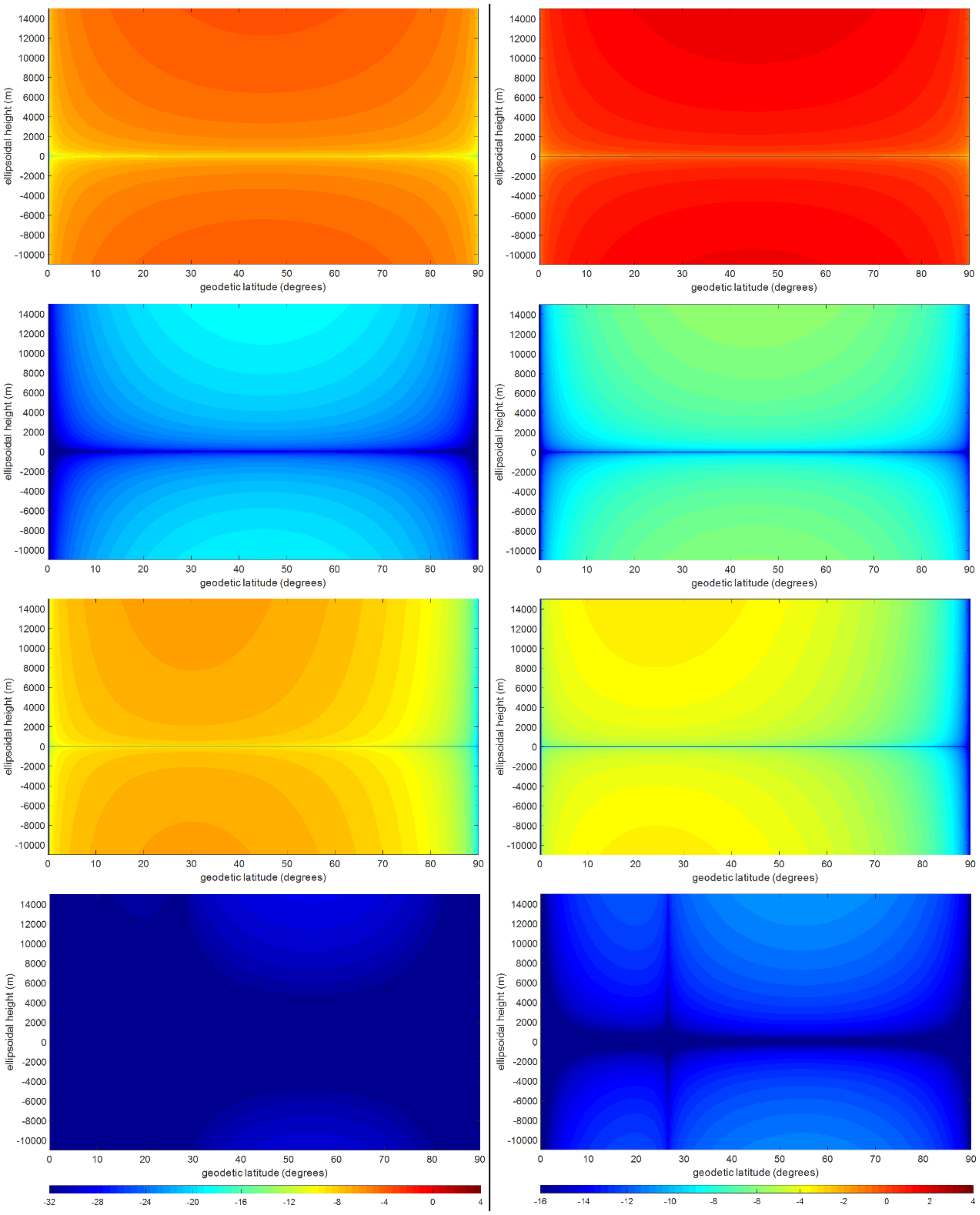

Figure 2. Approximation error in the computation of ellipsoidal height (left) and geodetic latitude (right) using the spherical method (top row), Bowring method (second row), Pollard method (third row) and Fukushima method (bottom row) (units: m; scale bars show the logarithm of the error; note that the scale bar for the left figures has a larger range to properly indicate the accuracy of all methods; errors in latitude were converted from radians to metres through multiplication by r)

	Error in height direction (m)	Error in latitudinal direction (m)
Sampson	1.77E+01	9.73E-04
Modified Sampson	4.17E-02	2.30E-06
Uteshev	5.18E-02	2.86E-06
Modified Uteshev	7.27E-05	3.92E-09
Spherical	1.98E-04	5.03E+01
Bowring	3.13E-19	2.00E-06
Pollard	3.96E-07	5.83E-04
Fukushima	3.31E-29	2.05E-11

Table 1. Maximum absolute error in height and latitudinal direction for the transformation methods listed in the test range covering all latitudes and heights heights in the range from -11,000 m to +15,000 m

Figure 1 shows that the unmodified Sampson and Uteshev methods for the computation of ellipsoidal height are not of sufficient accuracy for most geodetic applications. However, the modified Uteshev method produces sub-millimetre accuracy in the whole test area. In fact, while it is not shown in Figure 1 and Table 1, the modified Sampson method can yield a comparable accuracy to the modified Uteshev method if the correction term (Eq. 25) is applied twice instead of once. In the computation of latitude, both Sampson's and Uteshev's methods produce sub-millimetre accuracy. This shows that Eq. (20), which in both methods computes geodetic latitude when ellipsoidal height is known, is insensitive to approximation errors in the ellipsoidal height. Equation (20) is therefore a very useful formula, however it appears to be (almost) completely unknown within the geodetic community.

Figure 2 shows that the spherical method should not be used for any points that are not on the surface of the ellipsoid. Even at a height of only 50 m, the error in latitudinal direction reaches 0.168 m. All other geodetic methods produce a level of accuracy that is sufficient for any practical application in the test range. The method of Fukushima (2006) is the most precise of the methods tested.

It is also interesting that in the geodetic methods the accuracy of the ellipsoidal height is higher than the equivalent accuracy of the latitude, whereas for Sampson's and Uteshev's method the opposite holds. This is thanks to the fact that Eq. (36), which computes geodetic latitude when ellipsoidal height is known in Bowring's and Fukushima's methods, is insensitive to approximation errors in the geodetic latitude. A final observation is that the modified Uteshev method produces more precise geodetic latitudes over most of the Earth's surface than any of the tested geodetic methods.

The numerical stability of the methods in regions near singularities, for example close to the poles, has not been studied here. A discussion on this can be found in many other publications (e.g., Bowring 1985, Borkowski 1989, Fukushima 1999).

7. NUMERICAL EFFICIENCY OF METHODS

Many researchers have compared computation times of various methods for the inverse geodetic transformation problem (e.g., Laskowski 1991, Gerdan and Deakin 1999, Seemkooei 2002, Fok and Iz 2003, Bajorek et al. 2014). However, studies do often not agree on the relative computation speed of different methods. The main reason for this is that computation time is highly dependent on various aspects, including hardware specifications, programming language, compiler, and implementation of the method. Therefore, the fastest method in one test setup will not necessarily be the fastest in another.

Fukushima (1999) has suggested comparing the various methods by an operation count instead. Methods that limits the use of computationally expensive operations such as divisions, square roots, and trigonometric functions, are generally computationally efficient. Fukushima (1999) provides relative computation times required for various operations. However, these also vary across different platforms and depend heavily on floating point precision. An operation count can give an indication of the computational efficiency of a method, but it can't definitively and reliably rank methods based on their efficiency under all circumstances. Nevertheless, it is the best method available for providing an indication of computational efficiency.

No matter whether methods are compared through a test of computational speed or through an operation count, it is important that each of the methods is implemented in an optimal sense. This is best illustrated using Bowring's (1976) method (Eqs. 34-35) as an example. It can be implemented naively as, for example, in the following snippet of code:

```
u=atan(a*Z/(b*W))
phi=atan((Z+ep2*b*sin(u))/(W-e2*a*cos(u))
```

This implementation requires two calls of the `atan` function, one of the `sin` function and one of the `cos` function, which is generally computationally expensive. Bowring's original implementation instead made use of the fact that the variable u does not need to be computed, because $\sin u$ and $\cos u$ can be computed directly from $\tan u$ using trigonometric identities. Additional minor savings can be made by avoiding on-the-fly use of operations between constants. For example, in the snippet of code above, the values of `ep2*b` and `e2*a` could have been stored in memory, avoiding two multiplications. A more efficient implementation of Bowring's method is (cf. Fukushima 1999, Appendix C)

```
T=c1*Z/W                    %T=tan(u)
C=1/sqrt(1+T*T)             %C=cos(u)
S=C*T                       %S=sin(u)
phi=atan((Z+c2*S*S*S)/(W-c3*C*C*C))
```

where `c1=a/b`, `c2=ep2*b` and `c3=e2*a`. This avoids one `atan`, one `sin` and one `cos` at the expense of one `sqrt`, one extra division, three extra multiplications and one extra addition.

We can optimise the implementation of Bowring's (1976) method even further. Using Pythagoras's theorem, alternative expressions for the sine and cosine of the auxiliary parameter u can be found

$$\sin u = \frac{aZ}{\sqrt{a^2Z^2 + b^2W^2}} \quad \text{and} \quad \cos u = \frac{bW}{\sqrt{a^2Z^2 + b^2W^2}} \tag{37}$$

Inserting these equations into Eq. (34) gives an alternative form of Bowring's formula

$$\phi = \arctan \frac{Z + a^4 b^{-4} L Z^3}{W - LW^3} \tag{38}$$

where

$$L = e^2 a^4 \left(\frac{W^2}{a^2} + \frac{Z^2}{b^2} \right)^{-\frac{3}{2}} \tag{39}$$

This can be implemented as follows:

```
W2=W*W
Z2=Z*Z
K=W2+c1*Z2
L=c2/(K*sqrt(K))
phi=atan((Z+c3*Z2*Z*L)/(W-W2*W*L))
```

where `c1=a*a/(b*b)`, `c2=e2*a` and `c3=c1*c1`. Compared to the previous implementation above, this saves one division while the number of all other operations is identical. In most situations, this will be the most efficient implementation.

Likewise, the computation of height (Eq. 36) can be optimised by removing the need to compute the sine and cosine of latitude, using the following equation instead

$$h = \frac{W + |Z| \tan\phi - a\sqrt{1 + (1 - e^2)\tan^2\phi}}{\sqrt{1 + \tan^2\phi}} \quad (40)$$

The implementation of all methods used in this study is summarised in Appendix A. Table 2 shows the operation count in the computation of geodetic latitude and height from Cartesian coordinates. All methods necessarily require one arctangent operation to compute the latitude, but do not require the evaluation of any other trigonometric functions. The most efficient method is then in theory the method that minimises the number of operations, but in particular the computationally expensive square root and division operations (cf. Fukushima 1999, Appendix C). Table 3 shows the same for the computation of ellipsoidal height only. It can, for example, be concluded from Table 3 that for the computation of ellipsoidal height only, Sampson's method would be expected to be the fastest method in any test because it has the lowest operation count for each type of operation.

	Spherical	Bowring (conventional implementation)	Bowring (new implementation)	Sampson	Modified Sampson	Uteshev	Modified Uteshev	Fukushima	Pollard
Addition/subtraction	5	8	8	12	13	14	15	11	12
Multiplication	7	15	14	16	18	23	26	31	14
Division	2	4	3	3	4	3	4	2	5
Square root	3	4	4	2	3	2	2	4	4
Arctangent	1	1	1	1	1	1	1	1	1

Table 2. Operation count for the computation of geodetic latitude and height from 3D Cartesian coordinates in various inverse transformation methods

	Spherical	Bowring (conventional implementation)	Bowring (new implementation)	Sampson	Modified Sampson	Uteshev	Modified Uteshev	Fukushima	Pollard
Addition/subtraction	5	8	8	4	6	6	8	11	10
Multiplication	7	15	14	7	9	14	17	31	12
Division	2	4	3	1	2	1	2	1	4
Square root	3	4	4	1	2	1	1	4	3
Arctangent	0	0	0	0	0	0	0	0	0

Table 3. Operation count for the computation of ellipsoidal height only from 3D Cartesian coordinates in various inverse transformation methods

It appears from Tables 2 and 3 that the methods of Sampson and Uteshev are computationally very efficient especially in the computation of ellipsoidal height. This is confirmed in a numerical test of computation times. The average computation time for each method was measured by performing more than 10^8 transformations of points regularly distributed in the test area. All methods were coded in Fortran95 with double precision arithmetic (`selected_real_kind(15,307)`). To test the variability in computation time, the code was compiled with different compilers (with and without code optimisation), and run on four different machines with different hardware specifications and operating systems. The specifications of the four machines used are shown in Table 4.

Machine	Operating system	Processor	RAM
M1	Red Hat Linux 6.10	40 Intel Xeon E5-2690 CPUs @ 3.00 GHz	378 GB
M2	Red Hat Linux 3.10	32 Intel Xeon E5-2690 CPUs @ 2.90 GHz	251 GB
M3	Windows 10 Enterprise	Intel Core i7-7700 CPU @ 3.60 GHz	16.0 GB
M4	Windows 10 Pro	Intel Core i5-6200U CPU @ 2.30 GHz	8.00 GB

Table 4. Hardware specifications of four machines (herein named M1, M2, M3, M4) used for computational speed tests

Table 5 shows the difference in relative computation time between the different transformation methods. The computation times are normalised relative to the conventional implementation of Bowring's method for the computation of latitude and height. In all cases, the code was compiled using the GNU compiler gfortran with optimisation flag O3. It can be seen that, in the computation of latitude and height, the spherical method is the fastest, but as seen in section 6 it is not sufficiently precise for most applications. The new implementation of Bowring's method (Eqs. 37-39) provides a significant advantage over the conventional implementation and is the fastest of the other methods tested across all machines used, but only marginally faster than Fukushima's method. Sampson's and Uteshev's method do not improve on the speed of Fukushima's method for the computation of latitude and height. However, when only the ellipsoidal height is of interest, Sampson's and Uteshev's methods, in unmodified or modified form, are faster than all other methods tested.

	Latitude and height				Height only			
	M1	M2	M3	M4	M1	M2	M3	M4
Spherical	0.60	0.64	0.77	0.76	0.32	0.35	0.08	0.07
Bowring (conv.)	1.00	1.00	1.00	1.00	0.52	0.62	0.16	0.19
Bowring (new)	0.84	0.85	0.87	0.88	0.44	0.53	0.13	0.13
Sampson	0.87	0.90	0.98	0.97	0.12	0.08	0.02	0.02
Modified Sampson	1.06	1.10	1.08	1.05	0.25	0.17	0.03	0.04
Uteshev	0.97	0.98	1.04	1.02	0.13	0.14	0.07	0.07
Modified Uteshev	1.15	1.14	1.13	1.10	0.21	0.17	0.07	0.07
Fukushima	0.86	0.91	0.98	0.96	0.37	0.40	0.18	0.20
Pollard	1.23	1.27	1.16	1.15	0.44	0.57	0.14	0.16

Table 5. Computation time of various methods for the inverse geodetic transformation, relative to the time required for Bowring's method in the conventional implementation, on four different machines (M1, M2, M3, M4) with different hardware specifications and operating systems

To test the influence of the compiler, the code was also compiled with the Intel compiler ifort, and with different optimisation flags. The result obtained in these tests on machine M3 (see Table 4) are shown in Table 6. It can be seen that the choice of compiler and optimisation has a significant influence on the test results. With the ifort compiler, the improvement of the new implementation of Bowring's method is more pronounced than with the gfortran compiler. However, regardless of the method of compilation, it can be concluded that 1) the new implementation of Bowring's method is the fastest method for the inverse geodetic transformation under all tests performed (apart from the imprecise spherical method), and 2) Sampson's and Uteshev's method do not provide a speed advantage for the complete inverse geodetic transformation, but are the fastest methods for the computation of ellipsoidal height only.

	Latitude and height				Height only			
	gfortran		ifort		gfortran		ifort	
	O0	O3	O0	O3	O0	O3	O0	O3
Spherical	0.76	0.77	0.49	0.49	0.10	0.08	0.32	0.13
Bowring (conv.)	1.00	1.00	1.00	1.00	0.24	0.16	0.61	0.31
Bowring (new)	0.88	0.87	0.74	0.73	0.18	0.13	0.51	0.23
Sampson	0.98	0.98	0.78	0.96	0.07	0.02	0.17	0.06
Modified Sampson	1.06	1.08	0.93	1.16	0.09	0.03	0.27	0.19
Uteshev	1.06	1.04	0.91	1.09	0.10	0.07	0.28	0.09
Modified Uteshev	1.16	1.13	1.10	1.36	0.14	0.07	0.39	0.17
Fukushima	1.02	0.98	1.12	0.92	0.29	0.18	0.80	0.34
Pollard	1.19	1.16	1.10	1.38	0.20	0.14	0.55	0.29

Table 6. Computation time of various methods for the inverse geodetic transformation, relative to the time required for Bowring's method in the conventional implementation, using the gfortran and ifort compilers with optimisation flags O0 and O3 (results from machine M3)

Finally, it is important to note from Tables 5 and 6 that tests for computation speed show great variability based on compiler and hardware, and results may be different for a setup not tested here.

The test results also depend on the programming language and floating point precision applied (only Fortran95 with double precision arithmetic was used here). Different floating point precision will affect methods differently depending on hardware specifications, choice of compiler and compiler settings. Results from a single test are not ever sufficient to draw definitive conclusions about the "optimal" transformation method for all situations.

8. CONCLUSIONS AND RECOMMENDATIONS

The methods of Sampson (1982) and Uteshev and Goncharova (2018) have been applied to the inverse geodetic transformation problem. Both methods are not sufficiently accurate for most geodetic applications, but a minor modification increases the accuracy of the ellipsoidal height by ~3 orders of magnitude. In the common region of application near the Earth's surface bounded by heights from -11,000 m to +15,000 m, the maximum error in the modified Sampson method is 42 mm, and the maximum error in the modified Uteshev method is 0.073 mm. In both methods, the modification consists of a simple additive correction to the height that is a function of the approximate height and the distance of the point to the geocentre.

One difference between the (modified) Sampson's and Uteshev's methods compared to most geodetic methods is that ellipsoidal height is estimated first, and geodetic latitude second using the computed height. In most geodetic methods, geodetic latitude is calculated first and ellipsoidal height second using the computed latitude. If only the ellipsoidal height is required, Sampson's and Uteshev's methods are therefore computationally more efficient than any of the existing geodetic methods. If a complete conversion from Cartesian to geodetic coordinates is required, the ranking of methods in terms of computation time is dependent on hardware, language, floating point precision, choice of compiler and compiler settings. The main advantage of Sampson's and

Uteshev's method is that they require less calls of the expensive square root operation than any other method.

A new formulation of Bowring's formula has also been presented here. It provides a significant advantage over the conventional formulation, giving between 12% and 27% saving in computation time in our numerical tests. Based on operation count, the new formulation of Bowring's method is also expected to be computationally more efficient than both Pollard's and Fukushima's method in (almost) any situation. However, Fukushima's (2006) method was only marginally slower in all tests performed here, and may perform better than the new implementation of Bowring's method in some situations. The main advantage of Fukushima's method is its impressive accuracy, which is superior to all other methods tested, while still being very computationally efficient.

REFERENCES

Bajorek, M., Kulczycki, M., Ligas, M., 2014. A comparison of iterative methods of the cubic rate convergence in the problem of transformation between Cartesian and geodetic coordinates. *Geomatics and Environmental Engineering*, 8(2): 15-25.

Bomford, G., 1971. *Geodesy*. 3rd edition. Oxford University Press, UK. 731 pages.

Borkowski, K. M., 1989. Accurate algorithms to transform geocentric to geodetic coordinates. *Bulletin Géodésique*. 63(1): 50-56.

Bowring, B. R., 1976. Transformation from spatial to geographical coordinates. *Survey Review*, 23(181): 323-327.

Bowring, B. R., 1985. The accuracy of geodetic latitude and height equations. *Survey Review,* 28(218): 202-206.

Civicioglu, P., 2012. Transforming geocentric Cartesian coordinates to geodetic coordinates by using differential search algorithm. *Computers and Geosciences*, 46: 229-247.

Featherstone, W. E., Claessens, S. J., 2008. Closed-form transformation between geodetic and ellipsoidal coordinates. *Studia Geophysica et Geodaetica*, 52: 1-18.

Fok, H.S., Iz, H. B., 2003. A comparative analysis of the performance of iterative and non-iterative solutions to the Cartesian to geodetic coordinate transformation. *Journal of Geospatial Engineering*. 5(2): 61-74.

Fukushima, T., 1999. Fast transform from geocentric to geodetic coordinates. *Journal of Geodesy*. 73(11), 603-610.

Fukushima, T. 2006. Transformation from Cartesian to geodetic coordinates accelerated by Halley's method. *Journal of Geodesy*. 79(12): 689-693.

Gerdan, G. P., Deakin, R. E., 1999. Transforming Cartesian coordinates X,Y,Z to geographical coordinates φ, λ, h. *The Australian Surveyor,* 44(1): 55-63.

Harker, M., O'Leary, P., 2006. First order geometric distance (the myth of Sampsonus). *BMVC,* doi: 10.5244/C.20.10

Heiskanen, W. A., Moritz, H., 1967. *Physical Geodesy*. W. H. Freeman & Co., San Fransisco, USA. 364 pages.

Laskowski, P., 1991. Is Newton's iteration faster than simple iteration for transformation between geocentric and geodetic coordinates? *Bulletin Géodésique*. 65(1): 14-17.

Ligas, M., 2012. Cartesian to geodetic coordinates conversion on a triaxial ellipsoid. *Journal of Geodesy*, 86: 249-256.

Moritz, H., 2000. Geodetic Reference System 1980. *Journal of Geodesy* 74(1): 128–162.

Paul, M. K., 1973. A note on computation of geodetic coordinates from geocentric (Cartesian) coordinates. *Bulletin Géodésique*. 108(2): 135-139.

Pollard, J., 2002. Iterative vector methods for computing geodetic latitude and height from rectangular coordinates. *Journal of Geodesy,* 76: 36-40.

Sampson, P. D., 1982. Fitting conic sections to "very scattered" data: an iterative refinement of the Bookstein algorithm. *Computer graphics and image processing*, 18: 97-108.

Seemkooei, A. A., 2002. Comparison of different algorithms to transform geocentric to geodetic coordinates. *Survey Review*. 36(286): 627-633.

Shu, C., Li, F., 2010. An iterative algorithm to compute geodetic coordinates. *Computers and Geosciences*, 36: 1145-1149.

Soler, T., Han, J. Y., Weston, N. D., 2012. Alternative transformation from Cartesian to geodetic coordinates by least-squares for GPS georeferencing applications. *Computers and Geosciences*, 42: 100-109.

Turner, J. D., 2009. A non-iterative and non-singular perturbation solution for transforming Cartesian to geodetic coordinates. *Journal of Geodesy,* 83(2): 139-145.

Uteshev, A. Yu., Goncharova, M. V., 2018. Point-to-ellipse and point-to-ellipsoid distance equation analysis. *Journal of Computational and Applied Mathematics,* 328: 232-251.

Uteshev, A. Yu., Yashina, M. V., 2015. Metric problems for quadrics in multidimensional space. *Journal of Symbolic Computation,* 68: 287-315.

Vermeille, H., 2004. Computing geodetic coordinates from geocentric coordinates. *Journal of Geodesy*. 78(1-2): 94-95.

Vermeille, H., 2011. An analytical method to transform geocentric into geodetic coordinates. *Journal of Geodesy*, 85(2): 105-117.

Zeng, H., 2013. Explicitly computing geodetic coordinates from Cartesian coordinates. *Earth, Planets and Space*, 65(4): 291-298.

APPENDIX A: CODE FOR TRANSFORMATION METHODS

This appendix shows how each of the methods discussed in this paper was implemented for the transformation from Cartesian coordinates to geodetic latitude and height ($(X, Y, Z) \rightarrow (\phi, h)$) over the study area. Constants to be stored in memory are named `c1`, `c2`, etc., and the formulas for their computation from the semi-major axis a and semi-minor axis b of the reference ellipsoid are shown underlined at the top of the code. Note that these codes do not include special cases to avoid singularities and are only applicable to the northern hemisphere; slight modifications would be required to make them more generally useable.

Sampson

```
c1=a*a,  c2=b*b,  c3=1/c1,  c4=1/c2,  c5=4*c3*c3,  c6=4*c4*c4,  c7=c1+c2,
c8=0.5*c1*c2, c9=c3+c4, c10=-4.5*c8
W2=X*X+Y*Y
Z2=Z*Z
G=c3*W2+c4*Z2-1
h=G/sqrt(c5*W2+c6*Z2)
h2=h*h
A1=W2+Z2-h2-c7
hA2=c8*(c9*h2-G)
mu=(c10*h2-A1*hA2)/(A1*A1-6*hA2)
phi=atan((c1-mu)*Z/((c2-mu)*sqrt(W2)))
```

Modified Sampson

```
c1=a*a,  c2=b*b,  c3=1/c1,  c4=1/c2,  c5=4*c3*c3,  c6=4*c4*c4,  c7=c1+c2,
c8=0.5*c1*c2, c9=c3+c4, c10=-4.5*c8
W2=X*X+Y*Y
Z2=Z*Z
G=c3*W2+c4*Z2-1
h0=G/sqrt(c5*W2+c6*Z2)
r2=W2+Z2
h=h0+h0*h0/(2*sqrt(r2))
```

```
h2=h*h
A1=r2-h2-c7
hA2=c8*(c9*h2-G)
mu=(c10*h2-A1*hA2)/(A1*A1-6*hA2)
phi=atan((c1-mu)*Z/((c2-mu)*sqrt(W2)))
```

Uteshev

```
c1=a*a, c2=b*b, c3=1/c1, c4=1/c2, c5=2*c3*c3, c6=2*c4*c4, c7=0.5*c3*c5,
c8=0.5*c4*c6, c9=c1+c2, c10=0.5*c1*c2, c11=c3+c4, c12=-9*c10
W2=X*X+Y*Y
Z2=Z*Z
S4=c5*W2+c6*Z2
S42=S4*S4
G=c3*W2+c4*Z2-1
h=G*sqrt((0.5*S42+(c7*W2+c8*Z2)*G)/(S42*S4))
h2=h*h
A1=W2+Z2-h2-c9
hA2=c10*(c11*h2-G)
mu=(c12*h2-A1*hA2)/(A1*A1-6*hA2)
phi=atan((c1-mu)*Z/((c2-mu)*sqrt(W2)))
```

Modified Uteshev

```
c1=a*a, c2=b*b, c3=1/c1, c4=1/c2, c5=2*c3*c3, c6=2*c4*c4, c7=0.5*c3*c5,
c8=0.5*c4*c6, c9=c1+c2, c10=0.5*c1*c2, c11=c3+c4, c12=-9*c10
W2=X*X+Y*Y
Z2=Z*Z
```

```
S4=c5*W2+c6*Z2
S42=S4*S4
G=c3*W2+c4*Z2-1
h0=G*sqrt((0.5*S42+(c7*W2+c8*Z2)*G)/(S42*S4))
r2=W2+Z2
h=h0+0.625*h0*h0*h0/r2
h2=h*h
A1=r2-h2-c9
hA2=c10*(c11*h2-G)
mu=(c12*h2-A1*hA2)/(A1*A1-6*hA2)
phi=atan((c1-mu)*Z/((c2-mu)*sqrt(W2)))
```

Spherical

```
c1=a*a/(b*b), c2=1/c1, c3=a
W=sqrt(X*X+Y*Y)
tau=c1*Z/W
phi=atan(tau)
tau2=tau*tau
h=(W+Z*tau-c3*sqrt(1+c2*tau2))/sqrt(1+tau2)
```

Bowring (conventional implementation)

```
c1=b*b/(a*a), c2=sqrt(c1), c3=a*(1-c1), c4=a
W=sqrt(X*X+Y*Y)
T= c1*Z/W
C=1/sqrt(1+T*T)
S=C*T
```

```
tau=(Z+c2*S*S*S)/(W-c3*C*C*C)
phi=atan(tau)
tau2=tau*tau
h=(W+Z*tau-c4*sqrt(1+c1*tau2))/sqrt(1+tau2)
```

Bowring (new implementation)

```
c1=a*a/(b*b), c2=1/c1, c3=c1*c1, c4=a*(1-c2), c5=a
W2=X*X+Y*Y
W=sqrt(W2)
Z2=Z*Z
K=W2+c1*Z2
L=c4/(K*sqrt(K))
tau=(Z+c3*Z2*Z*L)/(W-W2*W*L)
phi=atan(tau)
tau2=tau*tau
h=(W+Z*tau-c5*sqrt(1+c2*tau2))/sqrt(1+tau2)
```

Fukushima

```
c1=1/a, c2=b*b/(a*a), c3=1-c2, c4=sqrt(c2), c5=1.5*c3*c3, c6=a
W=sqrt(X*X+Y*Y)
s0=c1*Z
Wn=c1*W
c0=c4*Wn
c02=c0*c0
s02=s0*s0
a02=c02+s02
```

```
a0=sqrt(a02)
a03=a02*a0
f0=Wn*a03-c3*c02*c0
b0=c5*s02*c02*Wn*(a0-c4)
s1=(c4*s0*a03+c3*s02*s0)*f0-b0*s0
cc=c4*(f0*f0-b0*c0)
phi=atan(s1/cc)
s12=s1*s1
cc2=cc*cc
h=(W*cc+Z*s1-c6*sqrt(c2*s12+cc2))/sqrt(s12+cc2)
```

Pollard

```
c1=a*a, c2=c1/(b*b), c3=c2-1, c4=b*c3
W2=X*X+Y*Y
Z2=Z*Z
Zp=Z+c4*Z/sqrt(W2+Z2)
PhN=sqrt(W2+Zp*Zp)
n=Zp/PhN
r=1+c3*n*n
s=W2/PhN+c2*n*Z
t=W2+c2*Z2-c1
h=(s-sqrt(s*s-r*t))/r
Z02=Z-n*h
phi=atan((Z+c3*Z02)/sqrt(W2))
```