C. C. TSCHERNING - K. PODER

☆

Some Geodetic Applications of Clenshaw Summation

ESTRATTO DAL « *BOLLETTINO DI GEODESIA E SCIENZE AFFINI* »
RIVISTA DELL' ISTITUTO GEOGRAFICO MILITARE
ANNO XLI - N. 4 - OTTOBRE - NOVEMBRE - DICEMBRE 1982

Some Geodetic Applications of Clenshaw Summation (*)

C.C. TSCHERNING - K. PODER

Geodaetisk Institut, Charlottenlund, Danmark

Summary. — The Clenshaw summation algorithm may be used for a numerical stable and fast summation of series of polynomials or functions, which fulfil a recurrence relationship. It is show how the algorithm may be modified in order to enable the evaluation of integrals or derivatives of the series.

It is described how the algorithm may be applied in a number of cases where numerical problems exist such as (1) the evaluation of certain map-projection transformations especially at very high latitudes (e.g. in North Greenland), (2) the evaluation of integrals or derivatives of a high degree and order spherical harmonic expansion of the earths gravity potential, (3) the evaluation of Molodensky type modifications to Stokes or Vening Meinesz kernels or (4) the evaluation of covariances between the derivatives of the anomalous potential to be used in a local gravity field determination.

(*) Presented at the 8th Symposium on Mathematical Geodesy — 5th Hotine Symposium — Como, Italy, September 7 - 9, 1981.

1. — INTRODUCTION

It is well known, cf. e.g. Davis (1963, Theorem 10.1.1), that real orthonormal polynomials, $P_n(x)$, satisfy a three-term recurrence relationship,

$$P_n(x) + a_n(x)\, P_{n-1}(x) + b_n\, P_{n-2}(x) = 0, \tag{1}$$

where $a_n(x) = a_n' x + a_n''$, and a_n', a_n'' and b_n are real numbers.

Also other polynomials, or functions (real, complex or of several variables) fulfil such relations. Furthermore we frequently find recurrence relations between the polynomials and their derivatives or integrals.

If P_n depends on several variables, $x = (x_1, ..., x_k)$, then both a_n and b_n in eq. (1) may depend (in a not necessarily linear manner) on one or more of the variables.

In geodesy polynomials or functions, which fulfil recurrence relations are in frequent use. Most important are ordinary real or complex polynomials, Legendre polynomials, associated Legendre functions, solid spherical harmonics, Chebychev polynomials and the trigonometric polynomials $\sin(mz)$, $\cos(mz)$, where z is real or complex and m an integer.

We may take advantage of recurrence relations like eq. (1), when computing sums, derivatives or integrals of sums of series. This technique is called Clenshaw summation and it is described in section 2. The technique has been extensively used in physical geodesy for the computation of sums or derivatives of sums of Legendre series ((Tscherning and Rapp, 1974), (Tscherning, 1976a)) and for the computation of double-sums of series of surface spherical harmonics (Gerstl, 1978). In section 3 we describe some new applications, namely for the summation of series of solid spherical harmonics and for the computation of integrals of (double) sums of surface spherical harmonics.

Clenshaw summation may also be applied in the computation of sums of (complex) trigonometric polynomials, which frequently are used in geometrical geodesy or in the evaluation of conformal mapprojections. Some of these applications are described in section 4.

The use of Clenshaw summation may result in considerable computational savings. However, even more important is the very favorable numerical properties of the method, see Clenshaw (1959), Smith (1967), Deuflhard (1976) and Gerstl

(1978). The fact, that the Clenshaw summation starts by summing the high order terms makes it especially well suited for most geodetic applications, because the higher order terms in general are small compared to the lower order terms.

Notation : i, j, k, n, m will be integers, letters like x, y, p, s will be vectors with elements (e.g. $x_1, x_2, \ldots, x_k$), $D_i^j = \partial^j / \partial_i^j$, A, C, D, E will be matrices.

2. — CLENSHAW SUMMATION

Let now $P_n(x) = P_n$ be a general polynomial or function of one or more real or complex variables. For a series with real or complex coefficients y_n, we put

$$S = S_m^N = \Sigma_{n=m}^N y_n P_n(x) = y^T p , \tag{2}$$

$$D_i^k S = \Sigma_{n=m}^N y_n D_i^k P_n(x) = y^T p_i^k , \tag{3}$$

$$SI = \Sigma_{n=m}^N y_n \cdot PI_n = y^T pi , \tag{4}$$

$$PI = \int_\Omega P_n(x)\, dx,$$

where Ω is the set of integration and

$$p = \begin{pmatrix} P_m \\ . \\ . \\ . \\ P_N \end{pmatrix}, \quad p_i^k = \begin{pmatrix} D_i^k P_m \\ . \\ . \\ . \\ D_i^k P_N \end{pmatrix}, \quad pi = \begin{pmatrix} PI_m \\ . \\ . \\ . \\ PI_N \end{pmatrix}, \quad D_i^1 = D_i ,$$

The very simple idea behind Clenshaw summation is best illustrated (cf. (Deuflhard, 1975)), if we express the evaluation of S on matrix form. Let us here for the sake of simplicity suppose, that for $n = m + 1$ we have $b_{m+1} = 0$ in eq. (1), i.e.

$$P_{m+1}(x) + a_{m+1}(x) P_m(x) = 0. \tag{5}$$

(The following is easily modified, if (5) does not hold). Then with

$$A = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ a_{m+1} & 1 & 0 & & 0 \\ b_{m+2} & a_{m+2} & 1 & . & 0 \\ . & . & . & & . \\ . & . & . & . & . \\ . & . & . & & . \\ . & . & . & & . \\ 0 & 0 & 0 & \ldots b_N & a_N \; 1 \end{pmatrix}, \quad p_0 = \begin{pmatrix} P_m \\ 0 \\ . \\ . \\ . \\ . \\ . \\ 0 \end{pmatrix},$$

we have

$$A\, p = p_0, \tag{6}$$

and from eq. (2), since A has determinant equal to 1 and hence is invertible,

$$S = y^T A^{-1} p_0 = p_0^T (A^T)^{-1} y. \tag{7}$$

A^T is an upper triangular matrix, for which the solution vector $(A^T)^{-1} y$ easily can be found. If we put $s^T = (s_m, \ldots, s_N)$, and

$$s = (A^T)^{-1} y\,,$$

then we have with $s_{N+1} = s_{N+2} = 0$,

$$s_n = - a_{n+1} \cdot s_{n+1} - b_{n+2} \cdot s_{n+2} + y_n \tag{8}$$

and

$$S = s_m \cdot P_m(x). \tag{9}$$

We get the first order derivative $D_i S$ by differentiating the matrix equation (7). However for the sake of simplicity we will now suppose that $a_n(x)$ is linear in x_i and that b_n is independent of x_i, i.e.

$$a_n(x) = a_n' x_i + a_n''\,.$$

(The following is easily modified if this is not true). Then

$$\begin{aligned} D_i S &= y^T \left(D_i(A^{-1})\, p_0 + A^{-1} D_i(p_0)\right) \\ &= y^T A^{-1} \left(- D_i(A)\, A^{-1} p_0 + D_i(p_0)\right) \\ &= s^T \left(- D_i(A)\, A^{-1} p_0 + D_i(p_0)\right). \end{aligned}$$

Because in this case

$$D_i(A) = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 \\ a'_{m+1} & 0 & 0 & & 0 \\ 0 & a'_{m+2} & 0 & & 0 \\ \cdot & & & \cdot & \cdot \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ 0 & 0 & 0 & \dots & a'_N \; 0 \end{pmatrix},$$

we have with

$$s' = -(s^T D_i(A) A^{-1})^T$$

and $s'_{N+1} = s'_{N+2} = 0$,

$$s'_n = -a_{n+1} \cdot s'_{n+1} - b_{n+2} s'_{n+2} - a'_{n+1} \cdot s_{n+1}, \tag{10}$$

and then

$$D_i S = s'_m P_m + s_m \cdot D_i P_m. \tag{11}$$

It is easily seen, that the recurrence formulae for higher order derivatives, D_i^k becomes

$$s_n^k = -a_{n+1} \cdot s_{n+1}^k - b_{n+2} \cdot s_{n+2}^k - a'_{n+1} \cdot s_{n+1}^{k-1} k, \tag{12}$$

$$D_i^k S = \Sigma_{j=0}^k \binom{k}{j} \cdot D_i^j P_m \cdot s_m^{k-j}, \qquad s_m^0 = s_m, \tag{13}$$

This equation is slightly more general that eq. (17) of Smith (1964).

Let us now suppose, that we for the integrals $PI_n = \int_a^b P_n(x)\, dx_1$, have a recurrence relationship of the following form,

$$C\, pi + E\,(p(b) - p(a)) = pi_0, \tag{14}$$

where C and E are matrices of the same simple form as the matrix A, so that

$$C = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ c_{m+1} & 1 & 0 & & 0 \\ d_{m+2} & c_{m+2} & 1 & & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix}, \quad E = \begin{pmatrix} e_m & 0 & 0 & \dots & 0 \\ f_{m+1} & e_{m+1} & 0 & & 0 \\ h_{m+2} & f_{m+2} & e_{m+2} & & 0 \\ \vdots & \vdots & \vdots & \cdot & \vdots \\ 0 & 0 & 0 & \dots & e_N \end{pmatrix}$$

and

$$p(b) = \begin{pmatrix} P_m(b) \\ P_{m+1}(b) \\ \cdot \\ \cdot \\ \cdot \\ P_N(b) \end{pmatrix}, \quad p(a) = \begin{pmatrix} P_m(a) \\ P_{m+1}(a) \\ \cdot \\ \cdot \\ \cdot \\ P_N(a) \end{pmatrix}, \quad pi_0 = \begin{pmatrix} PI_m \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{pmatrix}$$

Then

$$\begin{aligned} SI &= y^T pi = y^T C^{-1} (pi_0 - E A^{-1} (p_0(b) - p_0(a))) \\ &= si^T (pi_0 - E A^{-1} (p_0(b) - p_0(a))) \\ &= si^T pi_0 - g^T (p_0(b) - p_0(a)), \end{aligned} \tag{15}$$

with $si^T = (y^T C^{-1})$, $g^T = si^T E A^{-1}$.

Hence, if $si_{N+1} = si_{N+2} = g_{N+1} = g_{N+2} = 0$ and

$$si_n = -c_{n+1} \cdot si_{n+1} - d_{n+2}\, si_{n+2} + y_n, \tag{16}$$

$$g_n = -a_{n+1} \cdot g_{n+1} - b_{n+2} \cdot g_{n+2} + y_n + e_n \cdot si_n + f_{n+1} \cdot si_{n+1} + h_{n+2} \cdot si_{n+2}, \tag{17}$$

then

$$SI = si_m\, PI_m - g_m \,.\, (P_m(b) - P_m(a)). \tag{18}$$

Let us finally regard a series of complex polynomials, $P_n(z) = PR_n + i \cdot PJ_n$, where i is the imaginary unit, with coefficients $y_n = y_n^r + i \cdot y_n^j$. We must evaluate the real part of the sum $Re(S)$ and the imaginary part of the sum $Im(S)$ independently,

$$S = Re(s) + i\, Im(S) = (y^r + i\, y^j)^T (p^r + i\, p^j). \tag{19}$$

Let us put $a_n(z) = a_n^r + i \cdot a_n^j$, $b_n = b_n^r + i \cdot b_n^j$ in eq. (1) and $s_n = s_n^r + i \cdot s_n^j$. Then from eq. (8)

$$s_n^r = -a_{n+1}^r \cdot s_{n+1}^r + a_{n+1}^j \cdot s_{n+1}^j - b_{n+2}^r \cdot s_{n+2}^r - b_{n+2}^i \cdot s_{n+2}^i + y_n^r, \tag{20}$$

$$s_n^j = -a_{n+1}^j \cdot s_{n+1}^r - a_{n+1}^r \cdot s_{n+1}^j - b_{n+2}^r \cdot s_{n+2}^j - b_{n+2}^j \cdot s_{n+2}^r + y_i^n, \tag{21}$$

and

$$Re(S) = s^{r}_{m} \cdot Re(P_m) - s^{j}_{m} \cdot Im(P_m) \tag{22}$$

$$Im(S) = s^{r}_{m} \cdot Im(P_m) + s^{j}_{m} \cdot Re(P_m). \tag{23}$$

3. — CLENSHAW SUMMATION IN PHYSICAL GEODESY

3.1. – Evaluation of (local) isotropic covariance functions

A (nearly) standard technique for the representation of local isotropic covariance functions uses a closed expression, $F(r, r', \psi)$, which is modified by a sum of a finite Legendre series, see (Tscherning & Rapp, 1974, section 9),

$$C(P, Q) = F(r, r', \psi) - \Sigma_{n=0}^{N} \sigma_n \left(\frac{R^2}{r\, r'} \right)^{n+1} P_n(\cos \psi) ,$$

where P, Q are points in space, ψ is the spherical distance between the points, r, r' are the radial distances of P, respectively Q from the origin, σ_n are constants and P_n are the Legendre polynomials. With $s = R^2/(rr')$ and $t = \cos \psi$, the sum of the series can be expressed as

$$G(s, t) = \Sigma_{n=0}^{N} \sigma_n \, s^{n+1} \, P_n(t).$$

We put $p_n(t, s) = s^{n+1} P_n(t)$ and we have

$$p_n - (2n - 1)/n \; st \, p_{n-1} + (n - 1)/n \; s^2 \, p_{n-2} = 0, \; p_1 - st \, p_0 = 0, \tag{24},$$

i.e. $a_n = -(2n - 1) \; st/n$ and $b_n = (n - 1)s^2/n$. Eq. (8) and (9) can then be used for the evaluation of $G(s, t)$ and eq. (12) and (13) for the evaluation of derivatives with respect to t.

3.2. – Evaluation of a normal potential of the somigliana-Pizzetti type

A normal potential of the Somigliana-Pizzetti type (see Heiskanen & Moritz (1967, section 2-9)) may be expressed as the sum of a Legendre series plus the centrifugal potential. The sum and the derivatives of the sum can be calculated

as discussed above in 3.1 using $s = a/r$ and $t = \sin \bar{\varphi}$, where a is the semimajor axis of the reference ellipsoid and $\bar{\varphi}$ is the geocentric latitude, see Tscherning (1976a).

3.3. – Molodensky's improvement of Stokes formula

When computing gravimetric geoid undulations or deflections of the vertical using Stokes or Vening Meinesz integral formulae, the integration is in many cases only carried out over a spherical cap. In this case, the result may be improved in a least-squares sense by modifying the kernels by a function,

$$\tilde{S}(t) = \Sigma_{n=0}^{N} (2n + 1)/2 \; h_n \; P_n(t), \tag{25}$$

where h_n are constants, see e.g. Jekeli (1980, eq. (73)). $\tilde{S}(t)$ or its derivatives may easily be computed using $a_n = -(2n - 1)t/n$ and $b_n = (n - 1)/n$.

3.4. – Evaluation of sums of (external) solid spherical harmonic series

The external gravity potential of the earth, V, (or its corresponding anomalous potential, T) may be approximated by the sum of a series in solid spherical harmonics,

$$\tilde{V}(\bar{\varphi}, \lambda, r) = \Sigma_{n=0}^{N} \frac{GM}{r} \left(\frac{a}{r}\right)^n \Sigma_{m=0}^{n} P_n^m(\sin \bar{\varphi}) \; [\cos(m\lambda) \; C_n^m + \sin(m\lambda) \; S_n^m], \tag{26}$$

where λ is the longitude, GM the product of the gravitational constant and the mass ot the earth, P_n^m the associated Legendre functions and C_n^m, S_n^m the coefficients of the series. The computation of $\tilde{V}$ $(\bar{\varphi}, \lambda, r = a)$, i.e. for surface spherical harmonics, has been discussed by Gerstl (1978). We will here describe how Clenshaw summation may be applied for solid spherical harmonic series and for the derivatives of the series. Note, that similar results easily may be obtained for internal solid spherical harmonics.

It may of numerical reasons be of advantage to work with fully normalized or quasi-normalized associated Legendre functions. The latter will be denoted $\acute{P}_n^m$, with corresponding coefficients $\acute{C}_n^m$, $\acute{S}_n^m$, and we will here present results for these functions also. We now put $q = a/r$, $t = \sin \bar{\varphi}$, $u = \cos \bar{\varphi}$ and $w = GM/r$.

We first rearrange the double sum eq. (26),

$$\tilde{V}(\bar{\varphi}, \lambda, r) = \Sigma_{m=0}^{N} \frac{GM}{r} \Sigma_{n=0}^{N} P_n^m(t) \left[\cos(m\lambda)\, C_n^m + \sin(m\lambda)\, S_n^m\right] q^n , \tag{27}$$

$$= \Sigma_{n=0}^{N} \frac{GM}{r} \left[\Sigma_{n=m}^{N} P_n^m(t)\, q^n\, C_n^m \cos(m\lambda) + \Sigma_{n=m}^{N} P_n^m(t)\, q^n\, S_n^m \sin(m\lambda)\right] \tag{28}$$

Eq. (27) should be used when evaluating $\tilde{V}$ in points with varying latitude, while eq. (28) should be used if $\tilde{V}$ is evaluated in many points with identical latitude and r-value, because the sums

$$V_m^i = \Sigma_{n=m}^{N} P_n^m(t)\, q^n \begin{cases} C_n^m , & i = 1 , \\ S_n^m & i = 2 , \end{cases} \tag{29}$$

then only need to be evaluated one time. (This idea is e.g. used in Rizos (1979))

The associated Legendre functions fulfil the well known recurrence relations

$$P_n^m - \frac{(2n-1)}{n-m}\, t\, P_{n-1}^m + \frac{n+m-1}{n-m} P_{n-2}^m = 0,\ P_{m-1}^m = 0 , \tag{30}$$

$$P_m^m - (2m-1)\, u\, P_{m-1}^{m-1} = 0,\ P_0^0 = 1, \tag{31}$$

Correspondingly we have $p_n^m = P_n^m \cdot q^n$, $p_m = P_m^m \cdot q^m$,

$$p_n^m - \frac{2n-1}{n-1}\, tq\, p_{n-1}^m + \frac{n+m-1}{n-m}\, q^2\, p_{n-2}^m = 0 \tag{30a}$$

$$p_n^m - (2m-1)\, uq\, p_{m-1} = 0 \tag{31a}$$

The quasi-normalized Legendre functions are related to the un-normalized by

$$\acute{P}_n^m = P_n^m \left[\frac{(n-m)!}{(n+m)!}\right]^{1/2} , \acute{p}_n^m = \acute{P}_n^m\, q^n \tag{32}$$

hence

$$\acute{p}_n^m - \frac{2n-1}{((n+m)(n-m))^{1/2}} tq\, \acute{p}_{n-1}^m + \left[\frac{(n-m-1)(n+m-1)}{(n-m)(n+m)}\right]^{1/2} q^2\, \acute{p}_{n-2}^m = 0, \tag{30b}$$

$$\acute{p}_m - \left[\frac{2m-1}{2m}\right]^{1/2} uq\, \acute{p}_{m-1} = 0\,. \tag{31b}$$

The values of a_n and b_n derived from these equations must then be used, when series with quasi-normalized coefficients are used. The reader may easily derive the corresponding equations for fully-normalized coefficients. For un-normalized coefficients V_m^1 may be evaluated using eq. (8), (9) with

$$a_n = -\frac{2n-1}{n-m} tq,\ b_n = \frac{n+m-1}{n-m} q^2,\ v_m^1 = C_n^m,\ v_m^2 = S_n^m\,,$$

$$s_n^i = -a_{n+1} \cdot s_{n+1}^i - b_{n+2} \cdot s_{n+2}^i + v_n^i\,, \tag{33}$$

and

$$V_m^i = s_m^i \cdot p_m^m\,. \tag{34}$$

The sum

$$\tilde{V} = \Sigma_{m=0}^N w(V_m^1 \cos(m\lambda) + V_m^2 \sin(m\lambda)) = \Sigma_{m=0}^N w(s_m^1 \cos(m\lambda) + s_m^2 \sin(m\lambda))\, p_m,$$

can then be evaluated again using eq. (8), (9) with

$$a_m = -(2m-1)\, uq,\ b_m = 0,\ v_m = s_m^1 \cos(m\lambda) + s_m^2 \sin(m\lambda). \tag{35}$$

(Clenshaw summation degenerates here to the well-known Horner's schema for the computation of the value of a usual polynomial).

However, when evaluating $\tilde{V}$, we may take advantage of the recurrence relations

$$\cos(m\lambda) = 2\cos\lambda \cdot \cos((m-1)\lambda) - \cos((m-2)\lambda) \tag{36}$$

$$\sin(m\lambda) = 2\cos\lambda \cdot \sin((m-1)\lambda) - \sin((m-2)\lambda)\,, \tag{37}$$

because

$$p_m \cos(m\lambda) = (2m - 1)\, uq \cdot p_{m-1} \cdot 2\cos\lambda \cdot \cos((m-1)\lambda) - (2m-1)(2m-3)\, u^2 q^2 p_{m-2} \cdot \cos((m-2)\lambda) \tag{38}$$

$$p \sin(m\lambda) = (2m - 1)\, uq \cdot p_{m-1} \cdot 2\cos\lambda \cdot \sin((m-1)\lambda) - (2m-1)(2m-3)\, u^2 q^2 p_{m-2} \cdot \sin((m-2)\lambda)\,. \tag{39}$$

In this case

$$a_m^i = -(2m-1)\, uq\, 2\cos\lambda,\ b_m^i = (2m-1)(2m-3)\, u^2 q^2,$$

and v_m^i is given by eq. (33).

However, it is in general more convenient to use eq. (36) and (37) for the direct computation of $\cos(m\lambda)$ and $\sin(m\lambda)$ and then eq. (35).

Derivatives with respect to r are most easily computed using for

$$\frac{\partial}{\partial r}\tilde{V} : v_m^i = -(n+1) \begin{cases} C_n^m & i = 1 \\ S_n^m & i = 2 \end{cases} \tag{40}$$

$$\frac{\partial^2}{\partial r^2}\tilde{V} : v_m^i = (n+1)(n+2) \begin{cases} C_n^m & i = 1 \\ S_n^m & i = 2 \end{cases} \tag{41}$$

The derivatives with respect to λ are found by computing V_m^i as in eq. (33), (34) and the using

$$\frac{\partial}{\partial\lambda}\tilde{V} = \Sigma_{m=0}^{N}\, m\, (V_m^2 \cos(m\lambda) - V_m^1 \sin(m\lambda))\, w\,, \tag{42}$$

$$\frac{\partial^2}{\partial\lambda^2}\tilde{V} = \Sigma_{m=0}^{N}\, m^2 (V_m^1 \cos(m\lambda) - V_m^2 \sin(m\lambda))\, w\,, \tag{43}$$

The first and second derivative of V_m^i with respect to t and the second order derivative of V_m^i with respect to t and r may be computed using eq. (33) combined with eq. (12) & (13) & (40). The mixed derivative with respect to λ and t may be computed using eq. (12), (13) & (42), and the mixed derivative with respect

to λ and r may be computed combining eq. (40) and (42). These derivatives with respect to t, λ and r can then be used for the computation of derivatives with respect to the coordinates of an arbitrary orthonormal frame as described in Tscherning (1976a).

Appendix 1 contains an algol procedure, which implements the above described computational method for un-normalized as well as quasi-normalized Legendre functions. The procedure has been designed so that it also will work for $\bar{\varphi} = 90^{\circ}$ or -90°.

This procedure permitted the numerical stable evaluation of $\tilde{V}$ with $N = 180$ on a (*RC* 8000) computer using $10\frac{1}{2}$ — digit arithmetic. The procedure published in Tscherning (1976b) did not work for $N > 90$.

3.5. – Computation of mean values of geoid undulations, gravity anomalies or deflections of the vertical

Let the anomalous gravity potential, T, be given by a series expansion like eq. (26). However, the summation will start from $n = 2$. We will here consider the computation of mean values over blocks where $\bar{\varphi}_0 < \bar{\varphi} < \bar{\varphi}_1$, $\lambda_0 < \lambda < \lambda_1$ and $r = R$, the mean radius of the Earth.

The integration with respect to λ is easily executed and the result is a new series independent of λ (and r), and with new coefficients, which we will denote H_n^m, i.e.,

$$\int_{\lambda_0}^{\lambda_1} V(\bar{\varphi}, \lambda, R)\, d\lambda = \Sigma_{m=0}^{N} \Sigma_{n=m}^{N} P_n^m(t)\, H_n^m \,.$$

Then

$$IV = \int_{\bar{\varphi}_0}^{\bar{\varphi}_1} \int_{\lambda_0}^{\lambda_1} V(\bar{\varphi}, \lambda, R) \cos\bar{\varphi}\, d\bar{\varphi}\, d\lambda = \Sigma_{m=0}^{N} \Sigma_{n=m}^{N} PI_n^m\, H_n^m, \text{ where}$$

$$PI_n^m = \int_{\bar{\varphi}_0}^{\bar{\varphi}_1} P_n^m(\sin\bar{\varphi}) \cos\bar{\varphi}\, d\bar{\varphi}$$

We have the following recurrence relations for PI_n^m, (cf. e.g. Gerstel (1980)),

$$PI_n^m = \frac{1}{n+1}\left[\frac{n+m-1}{n-m}(n-2)\, PI_{n-2}^m + \frac{2n-1}{n-m}\left(u_1^2 P_{n-1}^m(t_1) - u_0^2 P_{n-1}^m(t^0)\right)\right] \quad (44)$$

$$PI_m^m = \frac{1}{m+1}\left[(2m-1)(2m-3)m\, PI_{m-2}^{m-2} - (t_1 P_m^m(t_1) - t_0 P_m^m(t_0))\right], \quad (45)$$

with $PI^{m}_{m-1} = 0$, $PI^{0}_{0} = t_o - t_1$, $t_1 = \sin(\bar{\varphi}_1)$, $t_o = \sin(\bar{\varphi}_o)$, $u_1 = \cos(\bar{\varphi}_1)$, $u_o = \cos(\bar{\varphi}_o)$.

Hence, from eq. (14) we get

$$c^{m}_{n} = 0,\ d^{m}_{n} = -\frac{(n + m - 1)(n - 2)}{(n + 1)(n - m)}, \tag{46}$$

$$e^{m}_{n} = 0,\ f^{m}_{n} = -\frac{(2n - 1)\, u^2}{(n + 1)(n - m)},\ h^{n}_{m} = 0, \tag{46a}$$

which can be used for the first sum with respect to n. For the second sum, with respect to m we need

$$c_m = 0,\ d_m = -\frac{(2m - 1)(2m - 3)m}{m + 1}, \tag{47}$$

$$e_m = 0,\ f_m = t/(m + 1),\ h_m = 0. \tag{47a}$$

Hence, using eq. (16) — (18) two times, the integral of the series is computed. Mean geoid undulations are obtained using Bruns formula, i.e. the anomalous potential divided by the mean gravity. Mean free-air gravity anomalies are computed using coefficients multiplied by the factor $-(n - 1)/R$. Mean values of deflections of the vertical are computed using the fact, that they are (in linear approximation) equal to the derivatives of T with respect to φ and λ, (multiplied by $1/(r\gamma)$ and $1/(r\gamma u)$, respectively, where γ is the reference gravity).

4. — CLENSHAW SUMMATION IN GEOMETRICAL GEODESY AND MAP-PROJECTIONS

4.1. – Geometrical Geodesy

The fundamental work by König and Weise (1950) still forms the basis for the use of Clenshaw summation in connection with the ellipsoid and its mapping. This work, here referred to as KW, provides a wealth of series expansions, mostly in a standard form based upon the quantity $(a + b)/2$, the half sum of the two semiaxes of the ellipsoid and $n = (a - b)/(a + b)$, the « third flattening ». It is therefore very easy to compute the actual coefficients at «run time» from the actual ellipsoid used (instead of inserting numerical values in a programme or subroutine).

A single problem not handled by *KW* is the computing of the Gaussian curvature of the ellipsoid as a function of Gauss-Krüger (UTM) northings instead of the latitude (which not necessarily is available at run time).

It is a remarkable fact, that the curvature needed in arc-to-chord corrections or scale corrections (see e.g. Bomford (1971, section 2.66) on transverse mercator projections), should not be computed at the latitude of the point, but at the latitude giving a meridian arc length (from equator) equal to the northing on an ellipsoid scaled by the central scale of the projection — for UTM 0.9996.

Therefore, one can find a series giving the Gaussian curvature of the form

$$K = \Sigma_{i=0}^{m} q \, T_i(\dot{N}) ,$$

where T_i are the Chebychef polynomials, $\dot{N}$ is a parametric northing varying from -1 to $+1$, when the actual northing varies e.g. from equator to the pole. The coefficients may be found at run time by numerical Chebychef analysis. As

$$T_i(x) = 2x \, T_{i-1}(x) - T_{i-2}(x) ,$$

Clenshaw summation is applicable as discussed in section 2.

KW gives all usual curvature radii and curvatures as trigonometric series in cosine of multiples of the double latitude. These formulae may be (and are) used if one wants to take advantage of the fact that the « ellipsoidal » terms are 2 — 3 magnitudes smaller than the main (« spherical ») terms, but in many cases closed formulae based upon $V = sqrt\ (1 + (e')^2 \cos^2 B)$, where B is the geodetic latitude, are preferable.

The length of the meridian arc as a function of latitude and its inverse function may be evaluated with 12 — 14 digit computing precision from series of the form

$$\dot{G} = B + \Sigma_{k=1}^{5} p_k \sin (2kB) , \tag{48}$$

$$B = \dot{G} + \Sigma_{k=1}^{5} \dot{p}_k \sin (2k\dot{G}) \tag{49}$$

given in *KW* page 51 and 53, where $\dot{G}$ is the meridian arc length in units of the meridian quadrant.

These formulae extended to complex variables are the basis for a remarkably good set of formulae for transforming between geographical coordinates and transversal mercator coordinates as described in the next subsection.

4.2. – MAP-PROJECTIONS

The isometric latitude (defined by $dq = dB/\cos B$ for the sphere) and the longitude are frequently used as the basis for map-projections because of its simple form, but problems arize the nearer the one approaches the poles. Obviously a kind of isometric longitude $\cos B\, dL$ might cure the problem. Considering the fact, that eq. (48) and (49) have no singularity at the poles and taking $\dot{G}$ and B as complex variables,

$$\dot{G} = \dot{N} + i\dot{E} \text{ and } B = B_r + i\, B_i\,, \tag{50}$$

then we have for a sphere where $p_k \equiv 0$, that $B_r = \dot{N}$ and $B_i = \dot{E} \cdot \dot{N}$ and $\dot{E}$ become northing and easting, N and E, when they are scaled by the meridian quadrant length.

The formulae given in *KW* for transformations between (ellipsoidal) geographical coordinates and transverse mercator coordinates are based upon the generalization of eq. (48) and (49) to complex variables, thus avoiding the singularity of the isometric latitude at the poles. When using summation to the fourth order, 1 mm precision is obtained up to distances of 4000 km from the central meridian, and 1 — 2 mm from the pole. The undefined longitude at the pole is of course unavoidable.

A parametric sphere (not to be confused with the osculating sphere — Schmiegungskugel in the Gauss-Schreiber approach) is used to ease the problems with the imaginary latitude on the ellipsoid. The full set of transformations both ways are two pairs each of a real trigonometric series and a complex one. Appendix 2 lists a procedure (capable of transforming both ways) for this purpose and the initialization procedure for the constants of the transformation.

5. — CONCLUSIONS

We have here pointed out a number of applications of Clenshaw summation in geodesy. Its use gives many numerical as well as computational savings.

However, other applications are possible, because the only condition is the existence of an equation like (1). Note, that such an equation may involve more terms, which only will make the algorithm as given by eq. (8) and (9) slightly more complicated.

Acknowledgement: It was Torben Krarup, who many years ago pointed out to us the many advantages of Clenshaw summation.

REFERENCES

G. BOMFORD, *Geodesy*. Third Ed., Oxford at the Clarendon Press, 1971.

C.W. CLENSHAW, *A note on the summation of Chebyshev series*. MTAC. Vol. 9, pp. 118-120, 1955.

P.J. DAVIS, *Interpolation and Approximation*. Blaisdell, Waltham, Mass., 1963.

P. DEUFLHARD, *On Algorithms for the summation of Certain Special Functions*. Computing, Vol. 17, pp. 37-48, 1976.

M. GERSTL, *Vergleich von Algoritmen zur Summation von Kugelflächenfunktionen*. Veröff. d. Bayer. Komm. f.d. Int. Erdemssung, Heft Nr. 38, München, 1978.

M. GERSTL, *On the recursive computation of the integrals of the associated Legendre functions*. Manuscripta Geodaetica, Vol. 5, pp. 181-199, 1980.

W.A. HEISKANEN and H. MORITZ, *Physical Geodesy*. Freemann, St. Francisco, 1967.

C. JEKELI, *Reducing the error of geoid undulation computations by modifying Stokes' function*. Rep. Dep. Goedetic Science no. 301, Columbus, 1980.

R. KÖNIG und K.H. WEISE, *Mathematische Grundlagen der Höheren Geodäsie und Kartographie*. Erster Band. Springer-Verlag, 1951.

C. RIZOS, *An efficient computer technique for the evaluation of geopotential from spherical harmonics*. Austr. J. Geod. Surv., No. 31, pp. 161-169, 1979.

F.J. SMITH, *An algorithm for summing orthogonal polynomial series and their derivatives with applications to curve-fitting and interpolation*. Math. Comp., Vol. 21, pp. 629-638, 1967.

C.C. TSCHERNING, *Computation of second-order derivatives of the normal potential based on the representation by a Legendre series*. Manuscripta Geodaetica, Vol. 1 pp. 71-92, 1976a.

C.C. TSCHERNING, *On the Chain-Rule Method for Computing Potential Derivatives*. Manuscripta Geodaetica, Vol. 1, pp. 125-141, 1976b.

C.C. TSCHERNING and R.H. RAPP, *Closed covariance expressions for vravity anomalies, geoid undulations, and deflections of the vertical implied by anomaly degree variance models*. Rep. Dep. Geodetic Science no. 208, Columbus, 1974.

```
appendix_1              * page 1   17 08 81,  9.58;


external real procedure gpotdr(po, C, su, N, order, G);
value N, order;  integer N, order;  array po, G, C, su;

comment GI reg.no. 81013, programmed by C.C.Tscherning, july 1981.

References:
(1) Tscherning, C.C.: On the Chain-rule method for Computing Po-
    tential Derivatives. Manuscripte Geodaetica, Vol.1,
    pp. 125-141, 1976.
(2) This paper.

The procedure computes the value and up to the second order de-
rivatives of the potential of the Earth (W) or of its correspon-
ding anomalous potential (T).

The potential is represented by a series in solid spherical har-
monics, with un-normalised or quasi-normalized coefficients.
The chain-rule is used combined with the Clenshaw algorithm.
The array C must hold the coefficients, C(0, 0) = 1.0 for W and
0.0 for T, C(1) = C(1, 0), C(2) = C(1, 1), C(3) = S(1, 1) etc. up
to C((N+1)**2-1) = S(N, N).

Parameters:
(a) Call values:
N    The absolute value of N is equal to the maximal degree and
     order of the series.  N negative indicates, that the coeffi-
     cients are quasinormalized.
order The maximal order of the derivatives (< 3 p.t.).
po   Array holding position information. Bounds (1:6).
     po(1) = p, the distance from the Z (rotation) axis,
     po(2) = r, the distance from the origin,
     po(3), po(4) cos and sin of the geocentric polar angle,
     po(5), po(6) sin and cos of the longitude.
C    Must be declared with bounds (-3: (N+1)**2-1) when the coef-
     ficients are un-normalised and with bounds (-3:(N+3)**2-2)
     when the coefficients are quasi-normalized. C(1) to
     C((N+1)**2-1) contains the coefficients and we must have
     C(-3) = GM (the product of the mass and the gravitational
             constant,
     C(-2) = a, the semi-major axis of the used ref. ellipsoid,
     C(-1) = the angular velocity (= 0, when dealing with T).
     C((N+1)**2+k) = sqrt(k), k < 2*(N+1), when N < 0.
(b) Return values:
G    The result is stored in G as follows:
     G(0, 1) = dW/dx, G(0, 2) = dW/dy, G(0, 3) = dW/dz,
     G(1, 1) = ddW/ddx, G(1, 2) = G(2, 1) = ddW/dxdy,
     G(1, 3) = G(3, 1) = ddW/dxdz, G(2, 2) = ddW/ddy,
     G(2, 3) = G(3, 2) = ddW/dydz and G(3, 3) = ddW/ddz.
     where W may be interchanged with T and the variables
     x, y, z are the Cartesian coordinates in a local (fixed)
     frame with origin in the point of evaluation, x posi-
     tive, North, y positive East and z positive in the direc-
     tion of the radius-vector, (cf. ref.(1), eq. (4) and (5)).
The values of W or T will be returned in gpotdr.
;
```

```
comment appendix_1              * page 2   17 08 81,  9.58;

comment continued:
(c) Call and return values:
sm  array of dimension (0:K*(N+1)-1), where K = case order of
-   (2,  6, 10). Here are stored the partial sums, cf. ref.(2), eq.
-   (29), of P(n, m)*(a/r)**(n+1-m)/P(m, m)*(C(n, m) or S(n, m)
-   from n=m to n=N, and the derivatives of these sums. This makes
-   it unnecessary to recalculate these quantities, if the pro-
-   cedure is called subsequently with the same value of t and r,
-   and the same order;

begin
  integer n, m, i, j, k, km, n21, N21, n1, n2, m1, m2, max,
  max2, nm1, nm2, npm1; boolean quasi, deriv1, deriv2, pole;
  array sml, cml(0:abs(N));
  real vc, vs, vc1, vs1, v2, vxs, vxc, vxc1, vxs1, vzc, vzs,
  vzc1, vzs1, vxxc, vxxs, vxxc1, vxxs1, vxzc, vxzs, vxzc1, vxzs1,
  vzzc, vzzs, vzzc1, vzzs1, vzzm, vxxm, vyym, vxzm, vxym, vyzm,
  s2, sq1, m21, vm, vxm, vym, vzm, ck, ck1, ck2, ck12, u0, t2, a1u,
  sqnm1, sqnm2, sqnpm1, sqnpm2, m21t, m21u, m21u0,
  cm, sm, cl, sl, a1, a1t, b2, t, u, c, d, r, p, s, om2;
  own boolean first, new; own real oldt, oldr;
  own integer oldord, i1, i2, i3, i4, i5, i6, i7, i8, i9;

  comment we use now, that own booleans are initialised with the
  value false, first time the procedure is called in a program;

  if -, first then
  begin
    first:= true; oldt:= 2.0;
    j:= abs(N); i:= j+1; i1:= i; i2:= 2*i; i3:= 3*i; i4:= 4*i;
    i5:= 5*i; i6:= 6*i; i7:= 7*i; i8:= 8*i; i9:= 9*i;
  end;

  r:= po(2); p:= po(1); t:= po(3); u:= po(4); sl:= po(5);
  cl:= po(6); t2:= 2*t; pole:= abs(u) <= '-9;

  new:= abs(oldr-r) > '-3 or abs(oldt-t) > '-9 or oldord <> order
  -     or pole;
  if new then
  begin
    oldr:= r; oldt:= t; oldord:= order;
  end;

  quasi:= N < 0; if quasi then N:= -N; max:= (N+1)**2;
  s:= C(-2)/r; s2:= s*s; cml(0):= 1.0; sml(0):= 0.0; m1:= 0;
  deriv1:= order > 0; deriv2:= order > 1;

  comment sml(m), cml(m) are the sin and cos of m*(longitude);
  for m:= 1 step 1 until N do
  begin
    sml(m):= sml(m1)*cl+cml(m1)*sl;
    cml(m):= cml(m1)*cl-sml(m1)*sl; m1:= m;
  end;

  N21:= 2*N+1; vm:= vxm:= vym:= vzm:= 0.0; sqnm1:= sqnpm1:= 1.0;
  if deriv2 then vxxm:= vyym:= vzzm:= vxym:= vxzm:= vyzm:= 0.0;
  km:= max; max:= if quasi then N+max else N;
  max2:= max+N+1;
```

```
comment appendix_1              * page 3   17 08 81,  9.58;

  comment we now use the Clenshaw algorithm, cf. ref.(2), eq.(8
  - (13), modified in an obvious way following ref.(1);

  for m:= N step -1 until 0 do
  begin
    k:= km:= km-(if m=0 then 1 else 2);  n21:= N21;
    vs:= vc:= vs1:= vc1:= vxs1:= vxc1:= vzs:= vzc:= vzs1:= vzc1:=
    vxc:= vxs:= 0.0;
    if deriv2  then vxxc:= vxxs:= vxxc1:= vxxs1:= vzzc:=
    vzzs:= vzzc1:= vzzs1:= vxzc:= vxzs:= vxzc1:= vxzs1:= 0.0;
    cm:= cml(m);  sm:= sml(m);
    nm1:= max-m+2;  n1:= N+1;  npm1:= max+m+2;
    if deriv2 then m2:= m*m;

    if new then
    for n:= N step -1 until m do
    begin
      comment note that if -,quasi then nm2:=n-m+2, nm1:=n-m+1,
      npm1:= n+m+1, else the same+max. Furthermore sqnpm2:=
      sqrt(n+m+2), etc.. Also note, that the values at the end of the
      loop are used later on (npm1:= 2*m+1, sqnpm1:= sqrt(2*m+1),
      for example);

      nm2:= nm1;  nm1:= nm1-1;  npm1:= npm1-1;

      comment cf. ref.(2), eq.(40);
      if quasi then
      begin
        comment cf. ref.(2), eq.(30b);
        sqnm2:= sqnm1;  sqnm1:= C(nm1);
        sqnpm2:= sqnpm1;  sqnpm1:= C(npm1);
        sq1:= sqnm1*sqnpm1;  a1:= s*n21/sq1;
        b2:= -s2*sq1/(sqnm2*sqnpm2);
      end else
      begin
        comment cf. ref.(2), eq.(30);
        a1:= s*n21/nm1;  b2:= -s2*npm1/nm2;
      end;

      a1t:= a1*t;  a1u:= a1*u;
      n21:= n21-2;  ck:= C(k);  ck1:= C(k+1);  k:= k-n21;

      comment cf. ref.(2), eq.(33);

      v2:= vc1;  vc1:= vc;  vc:= vc1*a1t+v2*b2+ck;
      v2:= vs1;  vs1:= vs;  vs:= vs1*a1t+v2*b2+ck1;
      if deriv1 then
      begin
        ckz:= ck*n1;  ck1z:= ck1*n1;

        comment cf. ref.(2), eq.(10);
        v2:= vxc1;  vxc1:= vxc;  vxc:= vxc1*a1t+vc1*a1u+v2*b2;
        v2:= vxs1;  vxs1:= vxs;  vxs:= vxs1*a1t+vs1*a1u+v2*b2;
        v2:= vzc1;  vzc1:= vzc;  vzc:= vzc1*a1t+v2*b2-ckz;
        v2:= vzs1;  vzs1:= vzs;  vzs:= vzs1*a1t+v2*b2-ck1z;  n1:= n;
```

```
comment appendix_1              * page 4   17 08 81,  9.58;

        if deriv2 then
        begin  n2:= n+2;
          comment cf. ref.(2), eq.(41);
          v2:= vzzc1;  vzzc1:= vzzc;
          vzzc:= vzzc1*a1t+v2*b2+n2*ckz;
          v2:= vzzs1;  vzzs1:= vzzs;
          vzzs:= vzzs1*a1t+v2*b2+n2*ck1z;

          if pole then
          begin
            comment cf. ref.(2), eq.(12). The second order deriva-
            tive with respect to latitude;
            v2:= vxxc1;  vxxc1:= vxxc;
            vxxc:= a1t*(vxxc1-vc1)+2*a1u*vxc1+v2*b2;
            v2:= vxxs1;  vxxs1:= vxxs;
            vxxs:= a1t*(vxxs1-vs1)+2*a1u*vxs1+v2*b2;
          end;

          comment cf. ref.(2), eq.(10) and eq.(40), the deriva-
          tive with respect to r and the latitude;
          v2:= vxzc1;  vxzc1:= vxzc;
          vxzc:= vxzc1*a1t+vzc1*a1u+v2*b2;
          v2:= vxzs1;  vxzs1:= vxzs;
          vxzs:= vxzs1*a1t+vzs1*a1u+v2*b2;
        end second order deriv.;
      end deriv1;  .
    end;

    if new then
    begin
      su(m):= vc;  su(m+i1):= vs;
      if deriv1 then
      begin
        su(m+i2):= vxc;  su(m+i3):= vxs;
        su(m+i4):= vzc;  su(m+i5):= vzs;
        if deriv2 then
        begin
          su(m+i6):= vzzc;  su(m+i7):= vzzs;
          su(m+i8):= vxzc;  su(m+i9):= vxzs;
        end;
      end;
    end else
    begin
      vc:= su(m);  vs:= su(m+i1);
      if quasi then
      begin
        sqnpm1:= C(max2);  sqnpm2:= C(max2+1);
      end;
      npm1:= max2;  max2:= max2-2;
      if deriv1 then
      begin
        vxc:= su(m+i2);  vxs:= su(m+i3);
        vzc:= su(m+i4);  vzs:= su(m+i5);
        if deriv2 then
        begin
          vzzc:= su(m+i6);  vzzs:= su(m+i7);
          vxzc:= su(m+i8);  vxzs:= su(m+i9);
        end;
      end;
    end;
```

```
;       appendix_2              * page 1   17 08 81, 12.20;

;  References for all procedures are:
;  (1) Poder, Knud and Frede Madsen: Status of the new adjustment
;  in Greenland and the adjustment system of the Danish Geodetic
;  Institute. Proceedings 2'Int. Symp. on Probl. Rel. Redefi-
;  nition of North American Geodetic Networks, pp. 9-19, 1978.
;  (2) This paper.
;  (3) Jinsen Hansen, Hans (Ed.): ALGOL 6. Users Manual, RCSL NO:
;  31-D522, 1.ed, 1974.

;  Field variables are used in the procedures. They are described
;  in ref.(3), section 5.7.  In short, they act as pointers on
;  elements of an array, which then may be treated as booleans,
;  integers, long (integers) reals or the first elements of an array.
;  - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

external long procedure u_t_q
_               (N, E, B, L, sa, direct);
value            N, E,           direct;
array sa;  long N, E, B, L;  boolean direct;

comment GI reg.no. 81014, programmed by K.Poder, 4 nov. 1977.

function and parameters

u_t_q           (return)            long
Dual autochecking transformation procedure for transformation
between geographical coordinates (geo) and transversal mercator
coordinates. Here two types are permitted, namely ordinary utm
coordinates and transversal mercator coordinates with an 'inter-
mediate' longitude, itm. The transformation method used is de-
scribed in ref.(2) section 4.2. The procedure transforms utm/itm
-> geo when direct is true and the reverse otherwise.
An alarm is produced when the check by the inverse transforma-
tion exceeds the tolerance of 1.2 mm or an other value set by
the user in sa(19) for utm->geo or sa(20) for geo->utm. The
alarm action depends on the value of tr_status, a system variable.

tr_status >= 0  -> alarm exit, program termination
tr_status, leftmost bit=1 -> tr_status := tr_status + 1
tr_status, 2 leftm. bit=1 -> report on diff. on curr. outp.
and tr_status := tr_status + 1

The value of the procedure is the transformed N or latitude
in geotype units, (see ref.(1), p.15),

N, E                (call)              long
The utm- or geographical coordinates input for trans-
formation in geotype units.

B, L                (return)            long
The geographical or utm-coordinates output from the procedure
as transformed and checked coordinates in geotype units.

sa                  (call)              array
Transformation constants for direct and inverse transf.
See fields in sa or set_utm_const for a description

direct              (call)              boolean
direct = true => transformation utm, itm, sb -> geogr.
direct = false => transformation geogr -> utm, itm,

external constants and procedures:
clen_sin, clen_csin, tr_status;
```

```
comment appendix_2              * page 2   17 08 81, 12.20;

begin
  integer                       i, s, h;
  long                          Bg, Lg, N_check, E_check, N_dif, E_dif,
  -                             E_dcos, tol;
  real                          Np, Ep, Bg_r, Lg_r, cos_BN, dN, dE;
  long          field   Q_n, L0, E0, tol_f;
  array         field   bg_f, gb_f, utg_f, gtu_f;
  boolean               utm, tod, neg_utm;
  boolean       field   utm_f;
  real procedure arc_tan_h(x);
  value x;  real x;
  arc_tanh := if abs x < 0.95 then
  -           (ln((1 + x)/(1 - x))/2.0)
  -           else '300;

  comment fields in sa, see set_utm_const for details;

  Q_n   :=          4;  <*norm. mer. arc*>
  E0    := Q_n    + 4;  <*east. at central mer.*>
  L0    := E0     + 4;  <*central longitude*>
  bg_f  := L0         ;  <*ell. geo -> sph. geo*>
  gb_f  := bg_f   + 16;  <*sph. geo -> ell. geo*>
  gtu_f := gb_f   + 16;  <*sph. N,E -> ell. N,E*>
  utg_f := gtu_f + 16;  <*ell. N,E -> sph. N,E*>
  tol_f := utg_f
  -        + (if direct then 20 <*utm*> else 24 <*geo*>);
  utm_f := utg_f + 28;  <*true => utm, false => itm*>

  comment transformation sequence;

  i := if direct then 1 else 3;
  h := 4 - i;  s := 2 - i;

  comment check-values;
  utm     := sa.utm_f;
  N_check := N;  E_check := E;
  tol     := sa.tol_f;

  comment transformation cases;

  for i := i step s until h do case i of
  begin
    comment case 1, utm,itm -> geo;
    begin
      comment normalize N, E;
      if utm and N > 10 000 000 000 000 then
      -      N := N - 20 000 000 000 000;
      Np       := N/sa.Q_n;
      Ep       := (E - sa.E0)/sa.Q_n;

      comment ellip. N, E -> sph. N, E;
      Np := Np + clen_csin(sa.utg_f, 4, 2*Np, 2*Ep, dN, dE);
      Ep := Ep + dE;

      comment sph. N, E = compl. sph. lat -> sph lat, lng;
      cos_BN := cos(Np);
      Lg_r   := arg(cos_BN, sinh(Ep));
      Bg_r   := arg(cos_BN, sin(Np)*cos(Lg_r));

      comment sph. lat, lng -> ell. lat, lng;
      Bg_r := Bg_r + clen_sin(sa.gb_f, 4, 2*Bg_r);
      Bg   := Bg_r/rg;
      Lg   := Lg_r/rg;  Lg   := Lg + sa.L0;
    end case 1;
```

```
comment appendix_2              * page 3   17 08 81, 12.20;

    comment case 2, transf results;

    begin
      u_t_g :=  B := N := Bg;
      _         L := E := Lg;
    end case 2;

    comment case 3, geo -> utm, itm;

    begin

      <* NB NB NB: B,L refers to northing and easting,
      _            N,E refers to latitude and longitude*>

      comment ell. lat, lng -> sph. lat, lng;

      neg_utm := N < 0 and utm;
      Bg_r    := N*rg;
      Bg_r    := Bg_r + clen_sin(sa.bg_f, 4, 2*Bg_r);
      Lg_r    := (E - sa.L0)*rg;

      comment sph. lat, lng -> compl. sph. lat = sph N, E;

      cos_BN := cos(Bg_r);
      Np     := arg(cos(Lg_r)*cos_BN, sin(Bg_r));
      Ep     := arc_tanh(sin(Lg_r)*cos_BN);

      comment sph. normalized N, E -> ell. N, E;

      Np := Np + clen_csin(sa.gtu_f, 4, 2*Np, 2*Ep, dN, dE);
      Ep := Ep + dE;
      Bg := sa.Q_n*Np;  <*northing*>
      Lg := sa.Q_n*Ep + sa.E0;  <*easting*>
      if neg_utm then
      Bg := Bg + 20 000 000 000 000;

    end case 3;

  end transf cases;

  comment in/rev-dif for check;

  N_dif  := Bg - N_check;
  E_dif  :=
  E_dcos := Lg - E_check;
  if -, direct then  E_dcos := E_dcos*cos(rg*N_check);

  _
  comment error actions;

  if abs N_dif > tol or abs E_dcos > tol then
  begin
    comment output of N_check, E_check, N_dif, E_dif
    and the error-exit not shown here;
    tr_status := tr_status + 1;

  end error actions;
end u_t_g;
end;
```