Wide Zone Transverse Mercator Projection

N. Stuifbergen

Ocean Sciences Division
Maritimes Region
Fisheries and Oceans Canada

Canadian Hydrographic Service
Bedford Institute of Oceanography
P.O. Box 1006
DARTMOUTH, Nova Scotia
Canada B2Y 4A2

2009

Canadian Technical Report of Hydrography and Ocean Sciences 262

Canadian Technical Report of Hydrography and Ocean Sciences

Technical reports contain scientific and technical information of a type that represents a contribution to existing knowledge but which is not normally found in the primary literature. The subject matter is generally related to programs and interests of the Oceans and Science sectors of Fisheries and Oceans Canada.

Technical reports may be cited as full publications. The correct citation appears above the abstract of each report. Each report is abstracted in the data base *Aquatic Sciences and Fisheries Abstracts*.

Technical reports are produced regionally but are numbered nationally. Requests for individual reports will be filled by the issuing establishment listed on the front cover and title page.

Regional and headquarters establishments of Ocean Science and Surveys ceased publication of their various report series as of December 1981. A complete listing of these publications and the last number issued under each title are published in the *Canadian Journal of Fisheries and Aquatic Sciences*, Volume 38: Index to Publications 1981. The current series began with Report Number 1 in January 1982.

Rapport technique canadien sur l'hydrographie et les sciences océaniques

Les rapports techniques contiennent des renseignements scientifiques et techniques qui constituent une contribution aux connaissances actuelles mais que l'on ne trouve pas normalement dans les revues scientifiques. Le sujet est généralement rattaché aux programmes et intérêts des secteurs des Océans et des Sciences de Pêches et Océans Canada.

Les rapports techniques peuvent être cités comme des publications à part entière. Le titre exact figure au-dessus du résumé de chaque rapport. Les rapports techniques sont résumés dans la base de données *Résumés des sciences aquatiques et halieutiques*.

Les rapports techniques sont produits à l'échelon régional, mais numérotés à l'échelon national. Les demandes de rapports seront satisfaites par l'établissement auteur dont le nom figure sur la couverture et la page de titre.

Les établissements de l'ancien secteur des Sciences et Levés océaniques dans les régions et à l'administration centrale ont cessé de publier leurs diverses séries de rapports en décembre 1981. Vous trouverez dans l'index des publications du volume 38 du *Journal canadien des sciences halieutiques et aquatiques*, la liste de ces publications ainsi que le dernier numéro paru dans chaque catégorie. La nouvelle série a commencé avec la publication du rapport numéro 1 en janvier 1982.

Canadian Technical Report of
Hydrography and Ocean Science No. 262

2009

Wide Zone Transverse Mercator Projection

by

N. Stuifbergen

Ocean Sciences Division
Maritimes Region
Fisheries and Oceans Canada

Canadian Hydrographic Service
Bedford Institute of Oceanography
1 Challenger Drive, P.O. Box 1006
DARTMOUTH, Nova Scotia
Canada B2Y 4A2

0.1 Acknowledgements

This report amounts to a free translation of the German-language article by Jürgen Klotz [Klotz, 1993] in which this analytical solution is developed.
Author's permission to make full use of the article content is much appreciated.

Valuable insights were included from ideas in the article by Egon Dorrer [Dorrer, 1999].

Acknowledged also are contributions of numerous co-workers, including BIO library staff and CHS Marine Geomatics support.

Cat No. FS97-18/262E ISSN:(Print) 0711-6764

Correct citation for this publication:

Stuifbergen, N., 2009. *Wide Zone Transverse Mercator Projection.*
Can. Tech. Rep. Hydrogr. Ocean Sci. No. 262: iv + 50 pp.

0.2 Abstract

ABSTRACT

A description is provided of the conversions of geodetic latitude and longitude into transverse Mercator grid co-ordinates, and vice-versa, for coverage of wide extent, with sub-millimetre accuracy tested up to 80 degrees from the central meridian. (160-degree zone-width).

It is useful as a tool to evaluate existing algorithms based on Taylor series expansions, which begin to degrade beyond the standard 6-degree zone-width of UTM.

Mathematical formulae with derivations, numerical examples and Fortran code are included.

RÉSUMÉ

Une description est fournie pour les conversions des latitudes et des longitudes géodésiques en coordonnées de quadrillage dans la projection de Mercator transverse, et inversement, pour une couverture de grande étendue, avec une exactitude submillimétrique vérifié jusqu'à 80 degrés du meridien central (largeur de zone de 160 degrés).

Cet outil s'avère utile pour évaluer des algorithmes existants, basés sur des expansions de la série de Taylor, qui commencent à se dégrader au delà d'une zone UTM d'une largeur normalisee de 6 degrés.

Des formules mathématiques avec leur dérivation, des exemples numériques et le programme Fortran sont inclus.

0.3 Disclaimer and Caution Note

Statutory Disclaimer

Neither the author, nor the Canadian Hydrographic Service, nor the Crown, accept any legal liability for the accuracy of this software, or legal liability for the results obtained by the use and/or application of this software.

Caution Note

This software has not been thoroughly tested for applications in the Southern hemisphere, nor for applications in the vicinity of longitude 180 degrees, around the International dateline.

Contents

APPENDIX

1.1 Introduction

This report describes a new analytical solution for conversions of geodetic latitude and longitude to/from Transverse Mercator grid co-ordinates. Sub-millimetre accuracy is achieved, tested up to 80 degrees from the central meridian (160-degree zone width)

Unlike other closed-form solutions, this analytical solution has the advantage of the mathematics being within the grasp of undergraduate engineers.

Existing solutions, based on voluminous Taylor series expansions, are designed for a 6-degree UTM zone width, beyond which their accuracy begins to degrade. [Redfearn,1948] [Thomas, 1968]

The contents of this report are arranged as follows:

Chapter 1 Front matter with preliminary information.
Chapter 2 has components of the process in detail, in a topic-by-topic sequence.
Chapter 3 covers the conversion process between Geodetic and Gauss-Kruger.
Chapter 4 has conclusion and other findings.
Chapter 5 contains literature sources.
Appendix A covers lesser related topics, along with numerical test data.
Appendix B contains listings of the Fortran code.

This algorithm uses complex variables, which by the principle of analytic continuation, extends the meridian arc formula into Gauss-Kruger co-ordinates, done simply by a change from real to complex data type of certain variables.
To enable this process, the geodetic co-ordinates must first be transformed into "complex intermediate latitude" co-ordinates w, with isometric properties.
Isometric properties for w are created by mapping geodetic coordinates ϕ, λ into isometric Mercator variables q, λ with the real `eLam` function for q, then back into complex w by the complex inverse Lambertian `(eGud)` function.
Then an integral E_3 is run on complex w to yield the Gauss-Kruger coordinates in unitary form z=x + iy. This integral is the extension of the meridian arc integral into complex variables.("analytic continuation")

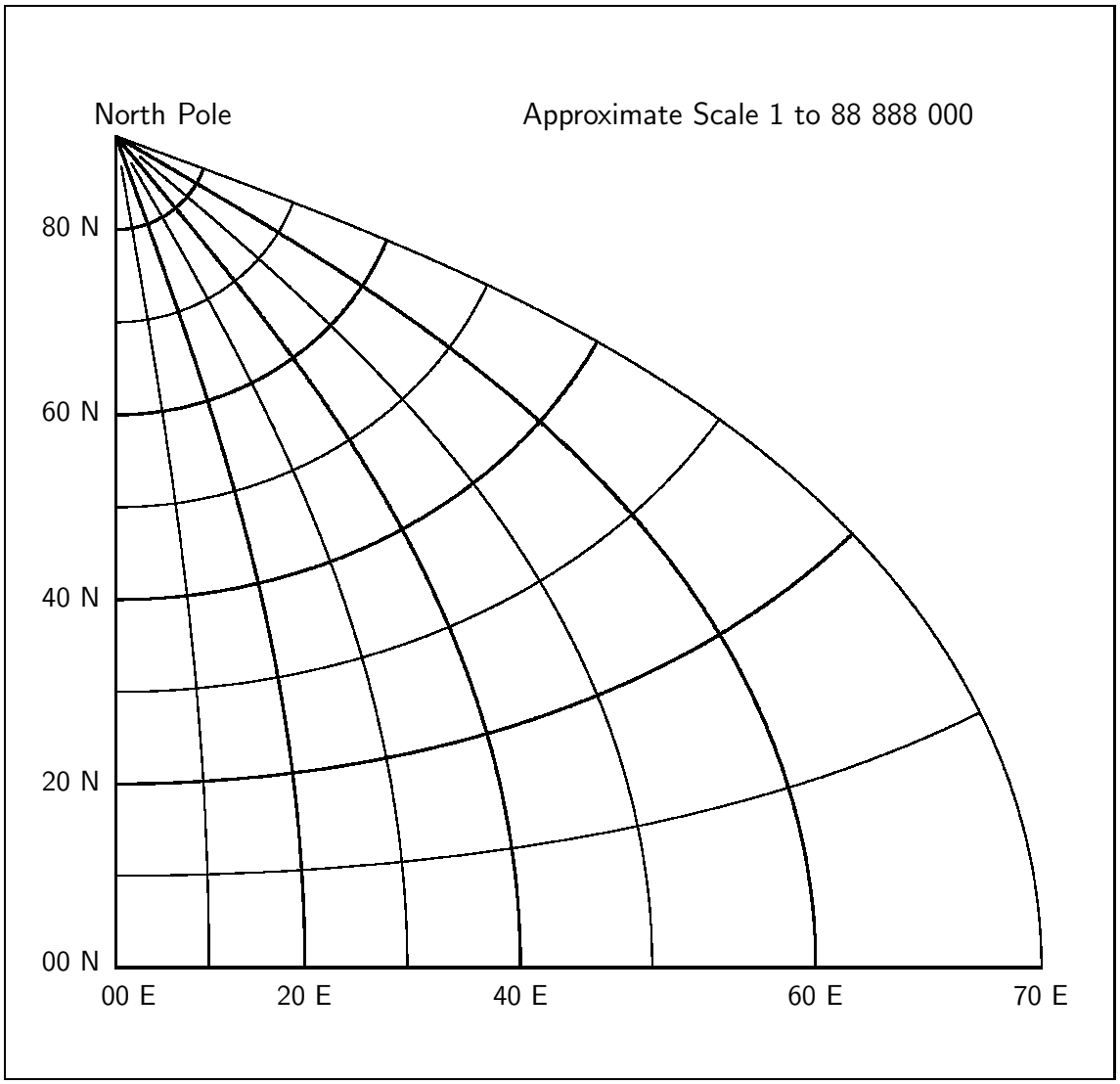
Test points are included for numerical verification, to full machine precision.
Recurrence formulae yield numerous simple expressions in the Fortran code.

The content and substance of this report is almost entirely based on a free translation of a German-language article that describes the original solution developed by Jürgen Klotz, of Potsdam, Germany. [Klotz, 1993]
Some ideas also, of encapsulating the mapping functions, were derived from an article by Egon Dorrer, of Munich, Germany.[Dorrer, 1999]

Readers are assumed to be familiar with UTM system of survey grids, some basics of calculus, with a refresher on complex variables and their functions.

It is possible that this report could become a useful contribution to the English technical literature on this special topic of practical geodesy.

1.2 Meridians and Parallels Diagram on Transverse Mercator



The graticule drawing reveals that this analytic solution is well-behaved in polar regions. Except for a slight curvature in the meridians, it closely resembles the polar stereographic projection. Beyond 80 degrees longitude, the mapping is greatly distorted (Scale expansion 15X)

It is widely believed that the transverse Mercator grid is unusable near the pole.

An untested speculation is that the problem is due to an artifact of the conventional power series method, in which the purely spherical terms (powers of t^2) are inadequate to model the high curvature of the parallels in polar latitudes.

Also the spherical transverse Mercator has a numerical break-down point at the pole, while the ellipsoidal solution does not.

1.3 Notation, Basic Formulae and Sign Conventions

1.3.1 Notation

Symbol	Description	Fortran	Units
a	major semi-axis	`Ae`	metres
b	minor semi-axis	`Be`	metres
f	ellipsoid flattening	`FL`	
n	2nd flattening	`FL2`	
e^2	1st excentricity-squared	`E2`	
e'^2	2nd excentricity-squared	`Ep2`	
ϕ	geodetic latitude	`Phi`	radians
λ	geodetic longitude	`Dlon`	radians
R_M	meridian radius of curvature	`Rm`	metres
R_N	lateral radius of curvature	`Rn`	metres
R_P	polar radius of curvature	`Rp`	metres
S_ϕ	meridian arc length	`ArcS`	metres
q	isometric latitude (Mercator)	`Q`	
ψ	complex Mercator variable	`Psi`	
w	complex intermediate latitude	`W`	
z	complex unitary coords	`Z`	
γ	meridian convergence	`Conv`	radians
k	point scale factor	`Psf`	
k_0	central scale factor	`Csf`	
!!	double factorial		
qatn(x,y)	quadrantal arctan (full-circle)	`ATAN2(X,Y)`	
-	Gauss-Kruger Northing	`Xn`	metres
-	Gauss-Kruger Easting	`Ye`	metres
-	UTM Northing	`Un`	metres
-	UTM Easting	`Ue`	metres
-	False Northing	`Fn`	metres
-	False Easting	`Fe`	metres

Note the convention, of using x for northing, and y for easting, which differs from customary American practice. It enables a more familiar notation for complex variables.

Function Np(n,p) = n(n-1)(n-2) $\cdots\cdots$ (n-p+1) is notation to capture the chain of n-factors of the binomial series.

Function wsin(ϕ, p) is notation to represent a Wallis sine integral:

$$\text{wsin}(\phi, p) = \int_0^\phi (\sin\,\varphi)^p \;\; d\varphi$$

1.3.2 Basic Formulae

f	$= (a-b)/a$	$\cdots$	ellipsoid flattening
n	$= (a-b)/(a+b) = f/(2-f)$	$\cdots$	2nd flattening
e^2	$= (a^2-b^2)/a^2 = (2-f)f$	$\cdots$	1st excentricity-squared
e'^2	$= (a^2-b^2)/b^2 = e^2/(1-e^2)$	$\cdots$	2nd excentricity-squared
W	$= \sqrt{1-e^2\,\sin^2\phi}$		
R_N	$= a/W$	$\cdots$	radius of curvature normal to meridian
R_M	$= R_N\,(1-e^2)/W^2$	$\cdots$	radius of curvature in meridian
R_P	$= a/(1-f)$	$\cdots$	radius of curvature at the pole
ψ	$= q + i\lambda$	$\cdots$	complex isometric latitude-Mercator
w	$= u + iv$	$\cdots$	complex intermediate latitude
z	$= x + iy$	$\cdots$	complex unitary GK coordinate
Xn	$= a\,(1-e^2)\,x$	$\cdots$	Gauss-Kruger Northing (metres)
Ye	$= a\,(1-e^2)\,y$	$\cdots$	Gauss-Kruger Easting (metres)

Double factorial notation: n !!

0 !! = 1

n !! = 2.4.6.8 $\cdots$ n – for n even

n !! = 1.3.5.7 $\cdots$ n – for n odd

1.3.3 Sign Conventions

Azimuths reckoned positive clock-wise from North-up.
Angles taken positive clockwise.

ϕ – North latitudes positive
λ – East longitudes positive
γ – Meridian convergence with same sign as longitude

X_n – Grid Northing (metres),positive northwards from Equator.
Y_e – Grid Easting (metres), positive eastwards from central meridian

Some authors use an opposite convention for meridian convergence.(Caution)

In the western hemisphere, it is common practice to enter and display longitudes as positive quantities, and convert to negative values in the computer to conform with mathematical convention.
This saves print of numerous minus signs in data processing.

The X-Y convention used here differs from usual practice (of X east, Y north), for the advantage that the numerous complex variables, z = x + iy, have the familiar appearance seen in standard textbooks.

Presentation of Angular Units

1. Inside the software, all angles and azimuths are in radian units.
2. In geodetic work, angular units shown in degrees, minutes and seconds,
   DDD MM-SS.ffff
3. For checks with hand calculators, decimal degrees are convenient. DDD.ffff ffff
4. Marine navigators prefer degrees and decimal minutes, DDD MM.ff,
   standard practice for convenience in plotting positions on the nautical chart.
   Having to convert every displayed value in degrees, minutes and seconds
   before plotting is a nuisance on the ships' bridge.
5. Full format display example: Lat 45 20.12 N, Long 63 32.15 E

Offsets - "False Easting" and "False Northing"

Published map grids have additional offsets applied, to ensure that all coordinate values are printed as positive numbers.

For UTM the offsets are:
Fe = + 500 000 metres added, the "'False Easting"
Fn = 0 in the northern hemisphere;
south of the Equator the added offset is 10 000 000 metres,
the "False Northing"(e.g. as in Australia and New Zealand)

2.1 Common Mercator Projection

The Mercator projection is a conformal projection of the earth ellipsoid, in which the longitude meridians are vertical parallels, and the latitude scale expands with northing for a conformal presentation.
Known as the Common Mercator, it is the projection traditionally used for marine navigation charts.
For a distinction from the Transverse Mercator, the term Normal Mercator is also seen. (Normal aspect vs. Transverse aspect)

Since geodetic coordinates (ϕ , λ) are not isometric, an isometric co-ordinate system of Mercator variables (q,λ) is introduced in order to enable conformal mapping by functions of complex variables.

$$q = \int \frac{R_m}{R_n \cos\varphi}\, d\varphi \qquad = \int_0^{\phi} \frac{1-e^2}{(1-e^2\sin^2\varphi)\cos\varphi}\, d\varphi$$

Solving the integral yields: $\boxed{\text{q = atanh(sin}\,\phi\text{) - e atanh(e sin}\,\phi\text{)}}$

Mercator Northing: Xn = a q Easting: Ye = a λ

The inverse relation, ϕ as a function of q, is found by iteration.(successive approximation)

$$\boxed{\sin\phi_{i+1} = \tanh\,[\ \text{q} + \text{e atanh(e sin}\,\phi_i\text{)}\ ]}$$

Since e is small (e $\approx$ 0.08), convergence is rapid.
$\psi = q + i\lambda$ $\cdots$ ψ denotes the complex isometric latitude, (Mercator)

In the subject of map projections, isometric latitude q arises so often that it is convenient to encapsulate it in a latitude function, named the Lambertian function:
q = Lam(ϕ) ... for a spherical earth,(e = 0)
q = eLam(e,ϕ) ... modified for the ellipsoid.

The inverse Lambertian functions:
ϕ = Lam^{-1}(q) = Gud(q) ... for a spherical earth,(e = 0)
ϕ = eLam^{-1}(e, q) = eGud(e,q) ... for the ellipsoid.

The inverse Lambertian is less well known by the term Gudermannian.

The functions eLam and eGud are also valid for complex variables.
In the Fortran software `eLam` and `eGud` are coded as subroutines so that the output can include derivatives also.

Sources: [Klotz, 1993 , p 107, eqn 1] [Dorrer, 1999]

2.2 Meridian Arc Computation

The computation of the meridian arc length s is by integrating radius of curvature in the meridian R_m over the latitude range.

$$S_\phi = \int_0^\phi R_M \, d\varphi \qquad \text{where: } R_M = \frac{a(1-e^2)}{W^3} \qquad \text{with } W = \sqrt{1 - e^2 \sin^2 \phi}$$

Inserting the radius of curvature R_M into the arc length formula for S_ϕ yields:

$$\boxed{S_\phi = a(1-e^2) \int_0^\phi (1 - e^2 \sin^2 \varphi)^{-\frac{3}{2}} \, d\varphi}$$

The integral is an elliptical integral of the 3rd kind, for which a closed-form solution is not known. It is solved by expanding the integrand, the term under the integral sign, into a binomial series and integrating term-by-term, for a summation sequence of Wallis integrals.

Two methods of computing meridian arc lengths are:

1. By Wallis integrals (see appendix A.4, page 24)

The solution by Wallis integrals can achieve a precision limited only by the number of significant digits available in floating point arithmetic of the computer.
A practical break-out tolerance is about 0.1 mm.
By design the meridian arc is exactly equal to Gauss-kruger Northing Xn along the central meridian ($\lambda = 0$).

2. By trigonometric series (see appendix A.4.2, page 25)

The conventional meridian arc length solution by series is of the form:

$$\boxed{S_\phi = a\,[\,(1 - A_0)\,\phi - A_2 \sin 2\phi + A_4 \sin 4\phi - A_6 \sin 6\phi + A_8 \sin 8\phi - A_{10} \sin 10\phi\,]}$$

Coefficients A_i are formed by terms in powers of e^2.
Schödlbauer uses terms in powers of e'^2 .
DMA technical manual uses terms in powers of n, the second flattening,
n=(a-b)/(a+b) $\approx 1/600$.

Comparison Tests of Meridian Arc Formulae

Using the solution by Wallis integrals as the standard, test data comparisons reveal that terms in e^{10} are insignificant. Terms carried up to e^6 only, are perhaps inadequate for survey grid accuracy, maximum error about 0.95 mm. Terms carried up to e^8 will yield a maximum error in computed meridian arc of 0.07 mm.
For a tabulation of comparisons (see appendix A.4.6, page 28)

2.3 Wallis Integrals by Recurrence Formula

Wallis sine integral: $\int_0^{\phi} (\sin\,\varphi)^p \; d\varphi = \; \mathrm{wsin}(\phi, p)$

... expressed by the function wsin

The recurrence formula is:

$$\boxed{\mathrm{wsin}(\phi, p) = [\, -\cos\phi\,(\sin\phi)^{p-1} + (p-1)\,\mathrm{wsin}(\phi, p-2)\,]/p}$$

Starter values: $\mathrm{wsin}(\,\phi,\,0) = \phi$ $\mathrm{wsin}(\,\phi,\,1) = -\cos\phi$

This recurrence formula is hard to find in the literature. A derivation is provided in appendix A.3 23, based on the method of integration by parts.

In this application we only need the even powers 2p in: $\int_0^{\phi} (\sin\,\varphi)^{2p} \; d\varphi$

Hence: $\boxed{\mathrm{wsin}(\phi, 2p) = [\, -\cos\phi\,(\sin\phi)^{2p-1} + (2p-1)\,\mathrm{wsin}(\phi, 2p-2)\,]\,/\,2p}$

Also valid for complex ϕ

In pseudo-code, with variables declared complex:

```
C     -- sequence of 10 complex Wallis integrals by recurrence formula:
      REAL*8  P2
      COMPLEX*16  CDCOS, CDSIN
      COMPLEX*16  Phi, CosPhi,SinPhi,  CSterm, Wsin2p(0:9)

      CosPhi = CDCOS(Phi);  SinPhi = CDSIN(Phi)
      CSterm = CosPhi* SinPhi;  P2 = 0;  Wsin2p(0) = Phi
      DO 100 I = 1,9
         P2 := P2+2
         Wsin2p(I) = (( -CSterm + ( P2-1)* Wsin2p(I-1) )) / P2
         CSterm := CSterm * SinPhi*SinPhi
  100 CONTINUE
```

Source: [Klotz, 1993 , p 108, Eqn 5]

2.4 Elliptic Integral Evaluation

The elliptical integral of the 3rd kind: E_3

$$\boxed{E_3 = \int_0^{\phi} \frac{\mathrm{d}\varphi}{\sqrt{(1-e^2\sin^2\varphi)^3}} = \int_0^{\phi} (1-e^2\sin^2\varphi)^{-\frac{3}{2}}\,\mathrm{d}\varphi}$$

occurs in the calculation of meridian arc length, and with complex w
in place of ϕ it is used to find Gauss-Kruger coordinates by analytic continuation.

It is evaluated by expanding the integrand into a binomial series,
and integrating the sequence of Wallis integrals term-by-term.

Binomial series:

$$(1+x)^n = 1 + \frac{x\,n}{1!} + \frac{x^2\,n(n-1)}{2!} + \frac{x^3\,n(n-1)(n-2)}{3!} \;\cdots\; \frac{x^p\;Np\,(n,\,p)}{p\,!}$$

Function Np(n,p) is used to capture the chain of n-factors.

$Np\,(n,0) = 1; \quad Np\,(n,1) = n; \quad Np\,(n,2) = n(n-1); \quad Np\,(n,3) = n(n-1)(n-2)$

$Np\,(n,p+1) = Np\,(n,p\,)\,(n-p) \qquad \cdots \qquad$ by recurrence formula

$$\boxed{(1+x)^n = 1 + \sum_{p=1}^{\infty} \frac{x^p}{p\,!}\;Np\,(n,p)}$$

2 substitutions for: $(1+x)^n = (1-e^2\sin^2\phi)^{-\frac{3}{2}}$ are: $\quad x = -e^2\sin^2\phi: \quad n = -3/2$

Then integrand: $(1-e^2\sin^2\phi)^{-\frac{3}{2}} = 1 + \sum_{p=1}^{6}(-1)^p\;(e^{2p})\;(\sin\phi)^{2p}\;Np\,(-\frac{3}{2},\,p)\,/\,p\,!$

where, for n = - 3/2:

$Np\,(-\frac{3}{2},0) = +1\,; \qquad Np\,(-\frac{3}{2},1) = -\frac{3}{2}\,; \qquad Np\,(-\frac{3}{2},2) = +\frac{3.5}{2.2}$

$Np\,(-\frac{3}{2},3) = -\frac{(3.5.7)}{2^3}\,; \quad Np\,(-\frac{3}{2},4) = +\frac{(9\;!!)}{2^4}\,;$

$Np\,(-\frac{3}{2},p) = (-1)^p\,\frac{(2p+1)\,!!}{2^p}$

Integration term-by-term:

$$E_3 = \int_0^{\phi} (1 - e^2 \sin^2\varphi)^{-\frac{3}{2}}\, \mathrm{d}\varphi = \phi + \sum_{p=1}^{\infty} \left[\underbrace{\frac{e^{2p}}{2^p}\, \frac{(2p+1)\,!\,!}{p\,!}}_{F_p} \quad \underbrace{\int^{\phi} (\sin\varphi)^{2p}\, \mathrm{d}\varphi}_{\mathrm{wsin}(\phi, 2p)} \right]$$

Factors F_p: $\quad F_0 = 1; \quad F_1 = (\frac{2p+1}{2\,p})\; e^2 = \frac{3}{2}\, e^2\,; \quad F_2 = \frac{15}{8}\, e^4 \quad \cdots$ etc.

$$F_p = F_{p-1} \left(\frac{2p+1}{2\,p} \right) e^2 \qquad \cdots \quad \text{by recurrence formula}$$

$$\boxed{E_3 = \phi + \sum_{p=1}^{6} F_p\, \mathrm{wsin}\,(\phi, 2p)}$$

The summation sequence converges quickly due to the small value of $e^2 \approx 0.0067$.
The Landen transformation, to accelerate convergence of the elliptic integral, is not needed.

Pseudo-code to evaluate complex integral E_3

```
     REAL*8      E2, P2, Fp, Tolerance
     COMPLEX*16  CDCOS, CDSIN
     COMPLEX*16  Phi, CosPhi,SinPhi, Wsin2p
     COMPLEX*16  CSterm, Term, Sum, E3

     CosPhi = CDCOS(Phi);  SinPhi = CDSIN(Phi)
     CSterm = CosPhi* SinPhi;  Fp=1;  P2 = 0;  Wsin2p = Phi

     Tolerance= 1d-14
     Sum=Phi
 100 CONTINUE
        P2 := P2+2
        Fp := Fp*E2*(P2+1)/P2
        Wsin2p := (( -CSterm + ( P2-1)* Wsin2p )) / P2
        Term = Fp*Wsin2p
        Sum := Sum + Term
        CSterm := CSterm*SinPhi*SinPhi
        IF ( CDABS(Term) .gt. Tolerance) GOTO 100 ! repeat until convergence
     E3=Sum
```

With recurrence formulae here in Fortran code, this evaluation of integral E_3 boils down to a relatively simple task.

3.1 Gauss-Kruger Projection

The Gauss-Kruger projection, also known as the Gauss Conformal, is the one conformal projection of the earth ellipsoid, in which the central meridian of the projection is held to have the same length and scale as the meridian arc of the ellipsoid.
The central meridian, also known as the "principal meridian", is the central axis of the projection.
In its standard form, the central meridian is taken to be at longitude 0 degrees.
The central scale factor set at 1.0000
Coordinate units in metres, for Northings Xn and Eastings Ye.
Xn is negative for latitudes south of equator.
For a conformal projection, the source image and its projection must consist of complex coordinates with isometric properties.
Then any analytic function on complex variables will preserve isometric properties,
by the Riemann-Cauchy conditions:

$$\boxed{\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \qquad \frac{\partial v}{\partial x} = -\frac{\partial u}{\partial y}}$$

The conversion, of geodetic coordinates (ϕ, λ) into Gauss-Kruger plane grid coordinates (Xn,Ye) is accomplished in three stages:

1. To convert geodetic latitude ϕ into isometric latitude q (a Mercator variable).
   The transformation of (ϕ , λ) into (q, λ) creates a mapping of geodetic coordinates into Mercator variables, isometric coordinate pairs.
   q is known as the "isometric latitude".
   (see section 2.1, page 6)

2. To transform complex isometric latitude ψ= q + iλ into the "complex intermediate latitude": w= u+iv, by the inverse Lambertian `eGud( ψ,w)`.

3. To evaluate the integral E_3 on w, to find the unitary coordinates: z = x + iy, and convert these to Gauss-Kruger metric coordinates Xn and Ye.
   (see section 2.4, page 9)

3.2 Process Outlines

3.2.1 Geodetic to UTM

Steps for geodetic to UTM conversion - forward solution

1. Convert geodetic latitude ϕ to isometric latitude q ,
   with the real elliptical Lambertian function: ($\lambda = 0$)
   $q = \text{eLam}(\, e, \phi \,)$

2. Shift geodetic longitude of UTM zone to the local longitude
   relative to central meridian of the zone:
   Shifted $\lambda := \lambda - \lambda_{CM}$

3. Insert local longitude λ into complex $\psi = q + i\lambda$

4. Transform complex isometric latitude ψ into complex w;
   w is the "complex intermediate latitude",
   found with the complex inverse Lambertian function (Gudermannian):
   $w = \text{eGud}(\, e, \psi \,)$

5. Run the elliptical integral of 3rd kind E_3 on w,
   to obtain unitary co-ordinates z = x + iy, by subroutine:
   `CALL Eint3M( e,w,dzw, z )`

6. Convert unitary x,y to Gauss-Kruger Xn, Ye (metres):
   Xn = a $(1 - e^2)$ x Ye = a $(1 - e^2)$ y a = major semi-axis

7. Convert generic Gauss-Kruger: Xn, Ye to UTM: Un, Ue
   Northing: Un = Fn + k_0 Xn Easting : Ue $\doteq$ Fe + k_0 Ye
   $k_0 = 0.9996$ UTM central scale factor.

8. Thus UTM northing Un and easting Ue are found.

3.2.2 UTM to Geodetic

Steps for UTM to geodetic conversion - inverse solution

1. Convert UTM to generic Gauss-Kruger Xn, Ye
   Northing: Xn = (Un-Fn)/k_0 Easting : Ye = (Ue-Fe)/k_0
   Fn = False Northing (metres) Fe = False Easting (metres)
   k_0 =0.9996 ... UTM central scale factor

2. Gauss-Kruger to complex unitary z = x + iy
   `Z= DCMPLX( Xn/Ae, Ye/Ae) /` ($1 - e^2$)

3. Perform inverse of elliptical integral, using Newton-Raphson iteration, given complex z to find w, the "complex intermediate latitude"
   `CALL InvEint3M( E, Z, W, dzw )`

4. Transform complex w into ψ, the complex isometric latitude,(Mercator variables)
   ψ= eLam(e, w) $\psi = q + i\lambda$

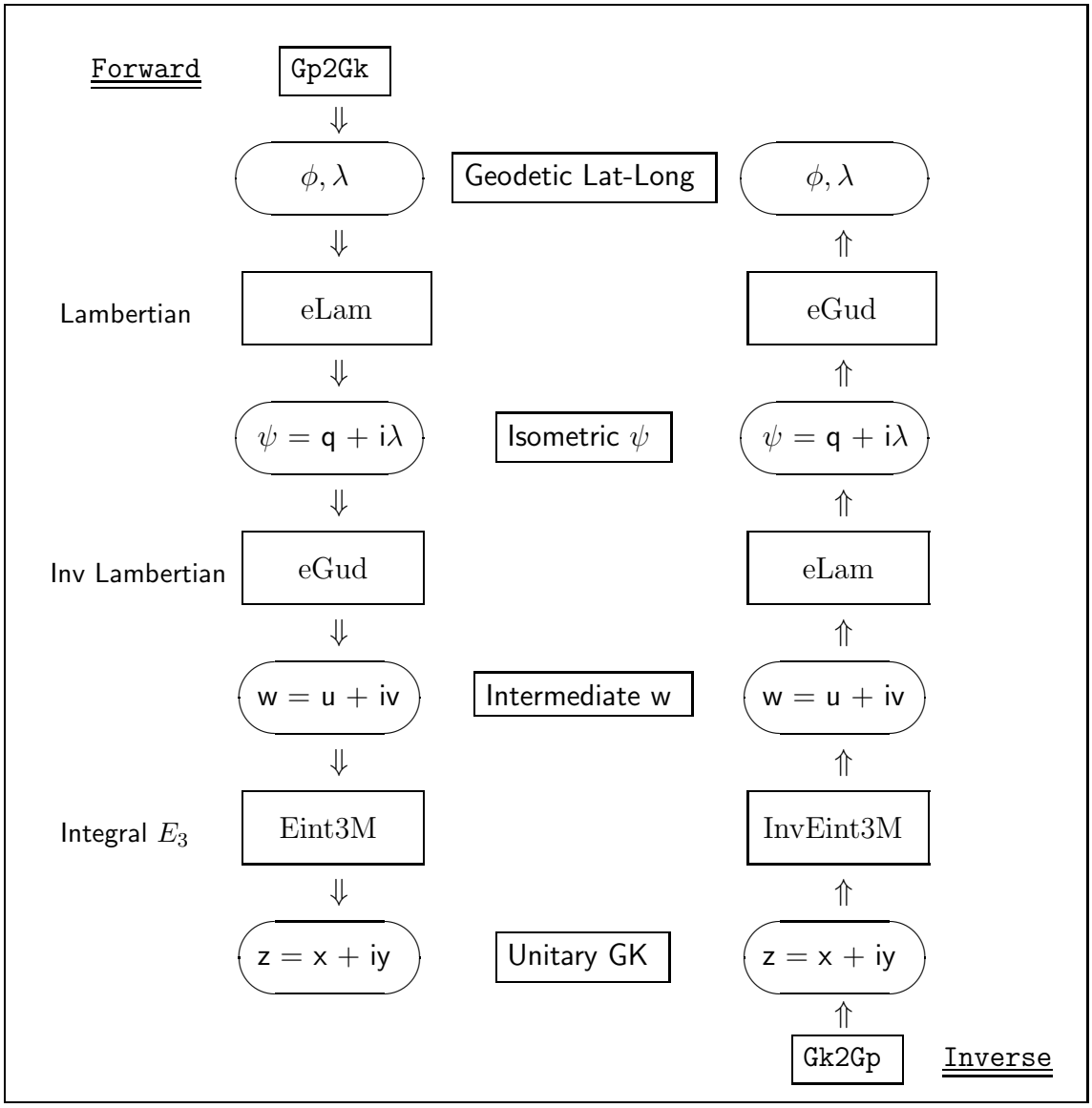
5. Separate ψ = q + iλ into real q , and λ the imaginary component.
   q = `DREAL` (ψ) λ = `DIMAG` (ψ)

6. Transform real isometric latitude q into geodetic latitude ϕ , by real inverse elliptical Lambertian function. (eGud)
   ϕ = eGud(e, q+i λ) , with $\lambda = 0$

7. Shift λ to the longitude of the zone central meridian λ_{CM}
   Shifted $\lambda := \lambda + \lambda_{CM}$

8. Thus the geodetic latitude ϕ and geodetic longitude λ are found.

3.3 Data Flow Schematic



The Fortran module names are, in subroutine form:

`eLam`	– Lambertian function geodetic to isometric
`eGud`	– Inverse Lambertian, (Gudermannian)
`Eint3M`	– Elliptical integral evaluation
`InvEint3M`	– Inverse of Elliptical integral evaluation

4.1 Conclusion and Findings

When the need arises to operate beyond the standard limitations of UTM, of millimetre accuracy within the 6-degree zonewidth, this new solution is helpful as a standard for a tool to evaluate the accuracy of existing conventional methods of conversion between Transverse Mercator and geodetic coordinates, conversions which are based on voluminous series expansion formulae.

This algorithm will yield sub-millimetre accuracy, tested within a zonewidth of up to 160 degrees.

It is useful also for very accurate zone-to-zone transformation, or for conversions between different Transverse Mercator grid coordinate systems. (e.g. UTM to MTM)

It could be a useful tool to evaluate the idea of wider zone configurations of UTM, in the Canadian Arctic, where 6-degree zone-widths are narrowed by meridian convergence. For example by selecting only the even-numbered zones to be implemented, with wider extent (12-degree zone-width) to cover the regions of odd-numbered zones. Thus there would be fewer zone boundaries. For UTM in the Svalbard-Spitsbergen region, a very similar practice was implemented by the Norwegian mapping agency.

The Gauss-Kruger projection may be regarded as the generic solution, from which any of the established Transverse Mercator grid systems of plane coordinates can be derived.

Findings

1. The Landen transformation, used by some authors to enhance the rate of convergence of the elliptical integral E_3, was found to be not necessary.
Due to the small value of $e^2 \approx 0.0067$, this analytic solution converges to full machine precision, in 5 recurrence cycles.
2. The meridian arc length by the conventional series solution of the form:

$$\boxed{S_\phi = a\,[\,(1 - A_0)\,\phi - A_2 \sin 2\phi + A_4 \sin 4\phi - A_6 \sin 6\phi + A_8 \sin 8\phi - A_{10} \sin 10\phi\,]}$$

was seen to have a maximum error of 0.95 mm at latitude 71 degrees, when terms are carried up to A_6 (to powers of e^6) only .
Carried to A_8 (to powers of e^8) yields a maximum error of 0.07 mm.
Carried to A_{10} (topowers of e^{10}) reveals no significant improvement.

3. A number of authors consider only the mathematics of coordinate conversions between Geodetic and Gauss-Kruger, and leave aside consideration of the computation of meridian convergence and scale expansion factor, very important to surveyors working on point-to-point computations in plane grid coordinates.

4. The graticule drawing reveals that this analytic solution is well-behaved in polar regions. Except for a slight curvature in the meridians, it closely resembles the polar stereographic projection.

Bibliography

5.1 Bibliography

[Klotz, 1993] Klotz,Jürgen. *Eine analytische Lösung der Gauß-Krüger Abbildung.* ("An analytical Solution of the Gauss-Kruger Projection") Zeitschrift für Vermessungswesen (ZfV), No 3, 1993, Potsdam, Germany. pp 106-116.

[Dorrer, 1999] Dorrer, Egon. *From Elliptic Arc Length to Gauss-Krüger Coordinates by Analytic Continuation.* Quo vadis geodesia? Anniversary Festschrift dedicated to Erik W. Grafarend, Schriftenreihe des Studiengang Geodäsie & Geoinformatik. Nr 6, Stuttgart, 1999, 9 pages.
www.uni-stuttgart.de/gi/research/schriftenreihe/quo_vadis/pdf/dorrer.pdf
Web search keywords "egon dorrer analytic continuation"

[DMA, 1990] Defense Mapping Agency. *DMA Technical Manual 8358.1* Datums,Ellipsoids,Grids and Grid Reference Systems. Defense Mapping Mapping Agency, (Hydrographic/Topographic) Fairfax, Va, USA. Sept 20 1990
http://earth-info.nga.mil/GandG/publications/tm8358.1/tr83581a.html
Web search keywords: "DMA 8358.1 Grids"

[DMA, 1989] Defense Mapping Agency. *DMA Technical Manual 8358.2* The Universal Grids: Universal Transverse Mercator (UTM) and Universal Polar Stereographic (UPS) Defense Mapping Mapping Agency (Hydrographic/Topographic), Fairfax, Va, USA. Sept 18 1989
http://earth-info.nga.mil/GandG/publications/tm8358.2/TM8358_2.pdf
Web search keywords: "DMA 8358.2 UTM UPS"

[Thomas, 1968] Thomas, Paul D. *Conformal Projections in Geodesy and Cartography.* Special Publication No. 251, United States Government Printing Office, Washington D.C. (1968)

[Snyder, 1987] Snyder, John P. *Map Projections - A Working Manual*
Professional Paper No. 1395, U.S. Geological Survey, 394 p. (1987)
http://pubs.er.usgs.gov/djvu/PP/PP-1395.pdf
Web search keywords: "map projections snyder 1395"

[Bomford, 1971] Bomford G., *Geodesy.*
Clarendon Press, Oxford, U.K., 731 p.

[Redfearn,1948] Redfearn, J.C.B.,(1948) *Transverse Mercator Formulae.*
Empire Survey Review, Vol 9, No. 69, pp 318-322.

[Bakker, 1995] Bakker,G. (1995) *Radio Positioning at Sea:*
Geodetic Survey Computations: Least Squares Adjustment.
G.Bakker, J.C. de Munck, G.L. Strang van Hees, University Press,
Delft University of Technology, the Netherlands.
ISBN 90-6275-537-2, 271 pages.

[Richardus, 1972] Richardus, P. *Map Projections.*
P.Richardus, R.Adler, American Elsevier Publishing Company, N.Y.,
ISBN 0-444-10362-7, 174 pages.

[Lee, 1976] Lee, L.P.(1976) *Conformal Projections based on Elliptic Functions.*
Cartographica, Monograph No. 16, York University, Toronto,
Canada, 128 pages

[Dozier, 1980] Dozier, Jeff. (1980) *Improved Algorithm for Calculation of UTM and Geodetic Coordinates.*
NOAA Technical Report NESS 81, U.S. National Environmental Satellite Service Series, Washington DC, 19 pages.

[Gerstl, 1984] Gerstl, M. (1984) *Die Gauß-Krügersche Abbildung des Erdesellipsoides mit direkter Berechnung der elliptischen Integrale durch Landentransformation.*
("The Gauss-Kruger Projection of the Earth Ellipsoid with Direct Computation of Elliptical Integrals by Landen Transformations")
DGK Reihe C, Heft Nr 296, Munich, Germany.
("German Geodetic Commission Series C, Issue No. 296")

[Wallis, 1992] Wallis,D.E.(Ph.D) *Transverse Mercator Projection via Elliptic Integrals* JPL Technical Report No. NPO-17996,
Jet Propulsion Laboratory (NASA),Pasadena, California,USA. (December 1992)
Search keywords: "Wallis transverse mercator elliptic integrals "

[Hooijberg, 2008] Hooijberg, M. (2008) *Geometrical Geodesy*
Springer-Verlag Berlin Heidelberg New York, 439 p.,
ISBN 978-3-340-25449-2

[Forssell, 2008] Forssell, Börje (2008) *Radionavigation Systems*
Artech House, 685 Canton St, Norwood, MA 02062, USA.
ISBN 1-59693-354-2, 392 p., US$ 79.

[Schödlbauer, 1981] Schödlbauer,A. (1981) *Gaußsche konforme Abbildung von Bezugsellipsoiden in die Ebene auf der Grundlage des transversalen Mercatorentwurfs.*
("Gauss Conformal Projection of Reference Ellipsoids
based on Transverse Mercator designs")
Algemeine Vermessungs-Nachrichten (AVN), 88. Jahrgang, Heft 5,
(" 88th Annual Volume, Issue no. 5")
Karlsruhe, Germany, May 1981. pp 165-173

5.2 Literature Sources Annotated

[Klotz, 1993] The original solution in Klotz' article is the primary source for this report. Main advantage is that the underlying mathematics is readily understood, and the method is entirely adequate for practical applications. This method overcomes the limitations of the power series solution.

[Dorrer, 1999] contributes the idea of using functions, modified for the ellipsoid, to encapsulate the mapping functions, `eLam` for the Lambertian and its inverse `eGud`.

[Redfearn,1948] presents the power series solution for geodetic to transverse Mercator and vice-versa.

[Thomas, 1968] provides similar power series solutions, along with a body of geodetic theory.

[DMA, 1990] DMA Technical Manual 8358.1 contains a wealth of tutorial information on the topic of grids and datums.

[DMA, 1989] DMA Technical Manual 8358.2 has a complete set of formulae for coordinate conversion by the power series methods, as used formerly in tables of latitude functions for use with mechanical calculators.
Tables TM 5-241-nnn Series for use by Army personnel.

[Schödlbauer, 1981] article has an interesting method of extending the validity of the power series solution, by incorporating the spherical solution.

[Lee, 1976] has developed a complete elegant closed-form solution, based on Jacobian elliptic functions, for mapping the entire spheriod.

[Dozier, 1980] provides a complicated C-code implementation of Lee's algorithm, using unfamiliar "theta functions". No test data for numerical verification.

[Bakker, 1995] contains the meridian arc computation by Wallis integrals. By a simple change into complex variables, the essential part of Gauss-Kruger coordinate computation is achieved.

[Bomford, 1971] contains the meridian arc length by trigonometric series up to terms in e^6.

[Gerstl, 1984] contains a closed-form solution, using the polar stereographic projection to create isometric coordinates. Includes Landen transformations to evaluate the elliptic integral. (A doctoral thesis)

[Wallis, 1992] contains a similar closed-form solution, also based on the polar stereographic projection, to form isometric coordinates, for mapping by complex variables. It has deep math.

[Hooijberg, 2008] Hooijberg has Gauss-Schreiber mapping equations, very similar to Gauss-Kruger, with a very different method of computation.

[Snyder, 1987] has formulae for a wide variety of map projections, mainly for computer graphics.

[Richardus, 1972] contains much general theory of map projections.

[Forssell, 2008] describes the Norwegian modifications to UTM, for their Arctic territory.

A.1 Various Forms of Transverse Mercator

There are a number of versions of the Transverse Mercator projection in existence, plane coordinate grid systems used for surveying and mapping, as instituted by each country's mapping agency.

In Canada provincial control surveys use 3-degree zones, also known as MTM, for "Modified Transverse Mercator". Central scale factor = 0.9999

Australia uses UTM in standard 6-degree zones, with a false Northing of 10 000 000 metres, for positive Northings in the southern hemisphere.

New Zealand used a similar UTM, with a particular central meridian shifted to 173 degrees East, chosen to cover the country entirely in a single zone.

South Africa uses narrow zonewidths of 2 degrees, central scale factor = 1.000
X coordinate for "Southings", Y coordinate for "Westings".
An advantage of narrow zones is that ground-to-grid corrections are very small, and can often be omitted.

Finland uses a dual system, a "Basic Grid" in 3-degree zones to benefit surveyors, and a "Uniform grid" in a single wide zone to cover the entire country, for mapping purposes.

In Norway, UTM zone 32V, in latitude band V (56 to 64 degs N, CM at 9 degs E) the zone is extended by 3 degrees farther west to longitude 3 deg E, so that the southwestern part of the country is covered within a single zone.
The neighbouring zone 31V to the westward is accordingly trimmed by 3 degrees so that its coverage then is all over-water only.

In the Svalbard (Spitsbergen) region, latitude band X (72 degs N to 84 degs N), where the zones are narrow due to meridian convergence, only the odd-numbered UTM zones 31X, 33X, 35X and 37X are implemented. These zones are widened to cover the omitted zones 32X, 34X and 36X. [Forssell, 2008]

In the USA a number of states use Transverse Mercator state plane grids, in diverse zone-widths to provide their "State Plane Coordinate System". [DMA, 1990]
The U.S. Army Map Service uses transverse Mercator for world-wide systematic coverage in UTM. [DMA, 1989]

The transverse Mercator system is used officially in the United Kingdom, Ireland, Sweden, Norway, Finland, Germany, Poland, Russia, China, Bulgaria, former Yugoslavia, Portugal, Egypt, former British African colonies, Southern Africa, Australia and New Zealand.[Thomas, 1968]

All of these TM grids are modified forms of the Gauss-Kruger projection, which could be regarded as the basic source algorithm from which to derive any one of the particular survey grids based on a transverse Mercator projection.

There is also a similar Gauss-Schreiber projection, which differs slightly from Gauss-Kruger, and is computed by a very different method.
Details can be found in [Hooijberg, 2008 , pp 190-200]

A.2 Transverse Mercator on the Sphere

On the sphere the Transverse Mercator formulae are simple.

x = Xn/r	...	Xn northing (metres)
y = Ye/r	...	Ye easting (metres)
r =	...	radius of the sphere (metres)

To convert unitary x,y to spherical ϕ,λ :

$$\sin\phi = \sin x / \cosh y \qquad \tan\lambda = \sinh y / \cos x$$

To convert spherical lat & long ϕ,λ to unitary x,y:

$$\tan x = \tan\phi \cos\lambda \qquad \tanh y = \cos\phi \sin\lambda$$

Meridian convergence γ :

$$\tan\gamma = \sin\phi \tan\lambda$$
$$\tan\gamma = \tan x \ \sinh y \cosh y$$

Scale factor k :

$$1/\mathrm{k} = \sqrt{(1-\sin\theta)(1+\sin\theta)}$$

where $\sin\theta = \cos\phi \sin\lambda$

or: $\mathrm{k} = \sec\theta$

Spherical transverse mercator coordinates are useful for sketching a picture of the TM grid configuration of meridians and parallels, or to find a simple initial start value for an iterative refinement process towards the ellipsoidal solution.
This spherical solution has numerical failure at the Pole.

Numerical example

```
r = 6378 137.0 metres
φ = 52 N = 0.907 571 211 037 radians
λ = 30 E = 0.523 598 775 598 radians

x = 0.975 939 236 385
y = 0.318 147 296 863

Xn = 6224 674.153 338
Ye = 2029 187.045 570 metres

Convergence γ = 24.463 551 876 degrees
              =   0.426 969 526 963 radians

Scale factor k = 1.061 037 170
```

Sources: [Klotz, 1993 , p 113] [Thomas, 1968 , p 108]

A.3 Derivation of Recurrence Formula for Wallis Integrals

This topic included here, since it is hard to find in the literature.
Wallis sine integral expressed as a function wsin:

$$\boxed{\int_0^{\phi} (\sin\,\varphi)^p \; d\varphi = \; \mathrm{wsin}(\phi, p)}$$

The recurrence formula is:

$$\mathrm{wsin}(\phi, p) = [\, -\cos\phi\,(\sin\phi)^{p-1} \; + (p-1)\,\mathrm{wsin}(\phi, p-2) \;\;] \;/\; \mathrm{p}$$

Initial values: $\mathrm{wsin}(\phi, 0) = \phi$ $\mathrm{wsin}(\phi, 1) = -\cos\phi$

Detailed derivation

This derivation is based on the method of "integration by parts" :

$$\int u\,dv \;\; = u\,v - \int v\,du$$

$$\int (\sin\phi)^p \, d\phi = \int (\sin\phi)^{p-1} \sin\phi \, d\phi = \int (\sin\phi)^{p-1} \, d(-\cos\phi)$$

$$u = (\sin\phi)^{p-1} \qquad v = -\cos\phi \qquad du = (p-1)\,(\sin\phi)^{p-2} \cos\phi \; d\phi$$

$$\begin{aligned} \int (\sin\,\phi)^p \;\; d\phi &= -\cos\phi\,(\sin\phi)^{p-1} + \int \cos^2\phi\,(p-1)\,(\sin\phi)^{p-2}\,d\phi \\ &= -\cos\phi\,(\sin\phi)^{p-1} \; + (p-1) \int (1-\sin^2\phi)\,(\sin\phi)^{p-2}\,d\phi \\ &= -\cos\phi\,(\sin\phi)^{p-1} \; + (p-1) \int (\sin\phi)^{p-2}\,d\phi \;\; - \;\; (p-1) \int (\sin\phi)^p \, d\phi \end{aligned}$$

$$\begin{aligned} p \int (\sin\,\phi)^p \;\; d\phi \quad &= \quad -\cos\phi\,(\sin\phi)^{p-1} \; + \; (p-1) \int (\sin\phi)^{p-2}\,d\phi \\ \int (\sin\,\phi)^p \;\; d\phi \quad &= \quad [\, -\cos\phi\,(\sin\phi)^{p-1} \; + \; (p-1) \int (\sin\phi)^{p-2}\,d\phi\,] \,/p \\ \mathrm{wsin}(\phi, p) \quad &= \quad [\, -\cos\phi\,(\sin\phi)^{p-1} \; + (p-1) \; \mathrm{wsin}(\phi, p-2) \;]/\,p \end{aligned}$$

In this application we only need the even powers 2p for: $\int_0^{\phi} (\sin\,\varphi)^{2p} \; d\varphi$

Hence: $\boxed{\mathrm{wsin}\;(\phi, 2p) = [\, -\cos\phi\,(\sin\phi)^{2p-1} + (2\mathrm{p}\text{-}1)\;\mathrm{wsin}(\;\phi,\; 2\mathrm{p}\text{-}2\;)\,] \;/\; 2\mathrm{p}}$

This recurrence formula is valid also for complex ϕ,
and for complex w as in the integral for E_3

A.4 Gauss-Kruger by Power Series Formulae

A.4.1 Gauss-Kruger Northing & Easting

Additional Notation:

$t = \tan\phi \qquad S_\phi =$ meridian arc metres $\qquad \eta^2 = \cos^2\phi\left(\frac{e^2}{1-e^2}\right)$

Easting: Y_E

$$Y_E = R_N\left[\, C_1\lambda\cos\phi + C_3\lambda^3\cos^3\phi + C_5\lambda^5\cos^5\phi + C_7\lambda^7\cos^7\phi\,\right]$$

$$Y_E = R_N\sum_{p=1}^{4} C_{2p-1}\left(\lambda\cos\phi\right)^{2p-1}$$

$C_1 = 1$

$C_3 = [\,1 - t^2 + \eta^2\,]\,/\,6$

$C_5 = [\,5 - 18t^2 + t^4 + 14\eta^2 - 58t^2\eta^2$
$\quad +13\eta^4 - 64t^2\eta^4 + 4\eta^6 - 24t^2\eta^6\,]\,/\,120$

$C_7 = [\,61 - 479t^2 + 179t^4 - t^6 + 331\eta^2 - 3298\eta^2t^2 + 1771\eta^2t^4$
$\quad +715\eta^4 - 8655\eta^4t^2 + 6080\eta^4t^4 + 769\eta^6 - 10964\eta^6t^2 + 9480\eta^6t^4$
$\quad +412\eta^8 - 6760\eta^8t^2 + 6912\eta^8t^4 + 88\eta^{10} - 1632\eta^{10}t^2 + 1920\eta^{10}t^4\,]\,/\,5040$

Northing: X_N

$$X_N = S_\phi + R_N\sin\phi\left[\, C_2\lambda^2\cos\phi + C_4\lambda^4\cos^3\phi + C_6\lambda^6\cos^5\phi + C_8\lambda^8\cos^7\phi\,\right]$$

$$X_N = S_\phi + R_N\tan\phi\sum_{p=1}^{4} C_{2p}\left(\lambda\cos\phi\right)^{2p}$$

$C_2 = 1/2$

$C_4 = [\,5 - t^2 + 9\eta^2 + 4\eta^4\,]\,/\,24$

$C_6 = [\,61 - 58t^2 + t^4 + 270\eta^2 - 330t^2\eta^2$
$\quad +445\eta^4 - 680t^2\eta^4 + 324\eta^6 - 600t^2\eta^6 + 88\eta^8 - 192t^2\eta^8\,]\,/\,720$

$C_8 = [\,+1385 - 3111t^2 + 543t^4 - t^6 + 10899\eta^2 - 32802\eta^2t^2 + 9219\eta^2t^4$
$\quad +34419\eta^4 - 129087\eta^4t^2 + 49644\eta^4t^4 + 56385\eta^6 - 252084\eta^6t^2 + 121800\eta^6t^4$
$\quad +50856\eta^8 - 263088\eta^8t^2 + 151872\eta^8t^4 + 24048\eta^{10} - 140928\eta^{10}t^2 + 94080\eta^{10}t^4$
$\quad +4672\eta^{12} - 30528\eta^{12}t^2 + 23040\eta^{12}t^4\,]\,/\,40320$

Sources: [Thomas, 1968 , pp 95-99, Eqns 288-301] [Redfearn,1948]

A.4.2 Meridian Arc, Scale Factor & Meridian Convergence

Meridian Arc Length S_ϕ by Power Series Formulae

$$S_\phi = \int_0^\phi R_M \, d\varphi \quad = \quad a(1-e^2) \int_0^\phi (1 - e^2 \sin^2 \varphi)^{-\frac{3}{2}} \, d\varphi$$

$$\boxed{S_\phi = a\,[\,(1-A_0)\,\phi - A_2 \sin 2\phi + A_4 \sin 4\phi - A_6 \sin 6\phi + A_8 \sin 8\phi - A_{10} \sin 10\phi\,]}$$

$$S_\phi = a\left[\,(1-A_0)\,\phi + \textstyle\sum_{p=1}^{5} (-1)^p\, A_{2p} (\sin\, 2p\,\phi\,)\right]$$

$$A_0 = \frac{1}{4}\left[e^2 + \frac{3}{16}e^4 + \frac{5}{64}e^6 + \frac{175}{4096}e^8 + \frac{441}{16384}e^{10}\right]$$

$$A_2 = \frac{3}{8}\left[e^2 + \frac{1}{4}e^4 + \frac{15}{128}e^6 + \frac{35}{512}e^8 + \frac{735}{16384}e^{10}\right]$$

$$A_4 = \frac{15}{256}\left[e^4 + \frac{3}{4}e^6 + \frac{41}{64}e^8 + \frac{81}{256}e^{10}\right]$$

$$A_6 = \frac{35}{3072}\left[e^6 + \frac{5}{4}e^8 + \frac{315}{256}e^{10}\right]$$

$$A_8 = \frac{315}{131072}\left[e^8 + \frac{7}{4}e^{10}\right]$$

$$A_{10} = \frac{693}{1310720}\left[\,e^{10}\,\right]$$

Point Scale Factor k

$$k = 1 + D_2\lambda^2 \cos^2\phi + D_4\lambda^4 \cos^4\phi + D_6\lambda^6 \cos^6\phi$$

$$D_2 = [1+\eta^2]\,/\,2$$
$$D_4 = [\,5 - 4t^2 + 14\eta^2 + 13\eta^4 + 4\eta^6 - 28t^2\eta^2 - 48t^2\eta^4 - 24t^2\eta^6\,]\,/\,24$$
$$D_6 = [\,61 - 148t^2 + 16t^4\,]\,/\,720$$

Meridian Convergence: γ

$$\tan\gamma = \lambda \sin\phi\,\left[\,1 + D_3\lambda^2\cos^2\phi + D_5\lambda^4\cos^4\phi + D_7\lambda^6\cos^6\phi\,\right]$$

$$D_3 = [1 + t^2 + 3\eta^2 + 2\eta^4]\,/\,3$$
$$D_5 = [2 + 4t^2 + 2t^4 + 15\eta^2 + 35\eta^4 + 33\eta^6 - 40t^2\eta^4 - 60t^2\eta^6 + 11\eta^8 - 24t^2\eta^8\,]\,/\,15$$
$$D_7 = 17\left(1+t^2\right)^3 / 315$$

A.4.3 Schödlbauer's modification of Power Series Formulae

By Schödlbauer's principle the power series solution can be modified towards a wider range of validity.
The method consists of removing all purely spherical terms in powers of t^2, replacing these with the exact transverse Mercator solution on the sphere of radius R_N, and keeping only the smaller ellipsoidal terms containing powers of η^2.

Since the northing Xn found by the spherical transverse Mercator implies a meridian arc calculated with radius R_N, an adjustment is to be applied for the meridian arclength that should be computed with radius a, the major semi-axis.

The power series solution consists of:
Northing :X_n = (Spherical terms, in powers of t^2)
+ (Ellipsoidal terms, in powers of η^2)
+ (Meridian Arc Length based on major semi-axis a)

Easting: Y_e = (Spherical terms, in powers of t^2)
+ (Ellipsoidal terms, in powers of η^2)

The power series solution degrades with distance from the central meridian, mainly because the spherical terms in t^2 are inadequate to model the curvature of latitude parallels, more so in high latitudes. Schödlbauer's method is interesting, but not needed here with the new analytical solution of this report available.

A.4.4 Reference Ellipsoids

Ellipsoid	major semi-axis a	minor semi-axis b	flattening 1/f	Datum
GRS-80	6378 137.0	6356 752.3142	298.257 2221	WGS-84
	6378 135.0	6356 750.52	298.26	WGS-72
PZ90	6378 136.0	6356 751.362	298.257 8393	(Russia)
Clarke 1866	6378 206.4	6356 683.8	294.978 6982	Nad-27
Hayford	6378 388.0	6356 911.946	297.0	International 1924
Bessel 1841	6377 397.155	6356 078.963	299.152 81285	

Source: [Hooijberg, 2008 , p. 122, table 18]

A.4.5 Gauss-Kruger to Geodetic by Iteration

The inverse solution, Gauss-Kruger to Geodetic conversion, can also be accomplished by an iteration process based on the forward solution, Geodetic to Gauss-Kruger.
Then in power series solutions, extended beyond design limits, the systematic error will be the same, consistent in both forward and inverse conversion modes.
Then it will be simpler to apply corrections by re-calibration to remove systematic errors in a post-processing phase, an advantage in data handling.

The process below, in "'pseudo-code" leaves out syntax detail.

```
      SUBROUTINE Gk2GpIter(Ae,FL, Xn,Ye, Phi, Glon,Psf,Conv)
      ====================
      Xn, Ye       -- input GK Northing & Easting (metres)
      Phi, Glon    -- output Lat/Long (radians)
      PhiT, GlonT  -- Lat/Long iterated
      XnT, YeT     -- Northing & Easting iterated
      Psf, Conv    -- Point scale factor & Meridian convergence
      Dxn,Dye      -- update increment (metres)
      Dn, De       -- lat/long update ( metres)

      CALL GK2Sp( Ae,  Xn,Ye, PhiT, GlonT)  -- start with spherical GK
      Tolerance = 0.001     -- 0.001 metre breakout tolerance

  100 CONTINUE       -- iteration loop
      CALL Gp2GkSC( Ae, FL, PhiT, GlonT,XnT,YeT, Psf, Conv )
      Dxn = ( Xn-XnT)/Psf;   Dye = ( Ye-YeT)/Psf  -- shift north and east

c     -- rotation for alignment with meridian convergence angle
      Dn = Dxn*COS(Conv) - Dye*SIN(Conv)
      De = Dxn*SIN(Conv) + Dye*COS(Conv)

      CALL Radii( Ae,Fl, PhiT, Rm,Rn )   -- ellipsoid curvature radii
      PhiT = PhiT + Dn/Rm;  GlonT = GlonT + De/( Rn*COS(PhiT) )  -- metres to radians

      Test = ABS( Dxn) + ABS( Dxe)
      IF ( Test .gt. Tolerance) GOTO 100  -- repeat until break-out

      Phi = PhiT;  Glon = GlonT     -- fix converged
      RETURN;  END
```

A.4.6 Comparison of Series Formulae for Meridian Arc Length

```
 GRS-80 Ellipsoid  ( WGS-84 )
 Major semi-axis  a = 6378137.0 metres
 Minor semi-axis  b = 6356 752.3142
 Flattening       f = 1 / 298.257222933
 Central Scale Factor = 1.000 000 (Gauss-Kruger)

=========================================================
Lat   True Arc Dist S      e10          e8          e6
---------------------------------------------------------
 90   10001965.729277   65.729277   65.729278   65.729493
                         0.000000    0.000001    0.000216

 80    8885139.871894   39.871849   39.871853   39.872653
                        -0.000045   -0.000041    0.000759

 70    7768980.727721   80.727652   80.727656   80.728677
                 **=>   -0.000069   -0.000065    0.000956 <=**

 60    6654072.819437   72.819377   72.819380   72.820197
                        -0.000060   -0.000057    0.000760

 50    5540847.041631   47.041607   47.041608   47.042038
                        -0.000024   -0.000023    0.000407

 40    4429529.030301   29.030325   29.030326   29.030442
                         0.000024    0.000025    0.000141

 30    3320113.397899   13.397960   13.397960   13.397927
                         0.000061    0.000061    0.000028

 20    2212366.254142   66.254211   66.254211   66.254144
                   **=>  0.000069    0.000069    0.000002

 10    1105854.833219   54.833264   54.833264   54.833219
                         0.000045    0.000045    0.000000

          [Note 1]       [Note 2]    [Note 3]    [Note 4]
 ---------------------------------------------------------
```

Notes:

1. Computed by Wallis integrals to full machine precision.
2. Series terms up to e^{10}, max error = 0.07 mm $\approx$ 0.1 mm at Lat 22.5 & 67.5
3. Series terms up to e^{8}, max error = 0.070 mm $\approx$ 0.1 mm at Lat 22.5 & 67.5
4. Series terms up to e^{6}, max error = 0.956 mm $\approx$ 1 mm at Lat 71

Findings: e6 off by about 1 mm, No difference between e8 and e10 series.

A.5 Test Point Data

A.5.1 Test Points on Int 24 (Hayford)

```
   International 1924 Ellipsoid ( Hayford)
  ======================================
  Ellipsoid major semi-axis a =    6378388.0000000 metres
  Inverse Flattening  1/f =     297.00000000000
  Equator to Pole Quadrant=    10002288.2989894

  I  Forward solution:  Geodetic to Gauss-Kruger - Gp2GkSc
  --------------------
  Latitude  Phi = 52.00    Longitude   Dlon =  30.00

  Gauss-Kruger  Xn =  6200529.3551360   Ye =  2033568.7650943
  Point Scale Factor =      1.0511296998133
  Meridian Convergence =   24.469356395842  degrees

     Inverse solution:   Gauss-Kruger to Geodetic - Gk2Gp
     -----------------
 Xn =  6200529.3551360     Ye = 2033568.7650943

 Phi = 52.000000000000   Dlon = 30.000000000000



  II Forward solution:  Geodetic to Gauss-Kruger - Gp2GkSc
  --------------------
  Latitude  Phi = 52.0  Longitude Dlon = 3.0

  Gauss-Kruger: Xn = 5767715.3137183   Ye = 206021.24821416
  Point Scale Factor =     1.0005208365438
  Meridian Convergence =   2.3648574978736 degrees

  Inverse solution:    Gauss-Kruger to Geodetic - Gk2Gp
  -----------------
 Xn=  5767715.3137183  Ye=  206021.24821416 metres

 Phi =  52.000000000000  Dlon = 3.0000000000002
```

Test points agree with [Klotz, 1993, p. 111]

A.5.2 Test Points on WGS-84

```
Major semi-axis  a = 6378137.0 metres        1/f = 298.25722293287
Phi = 52.0   Lon= 03.0     Xn= 5767595.2929206  Ye= 206011.32347739
Phi = 52.0   Lon= 30.0     Xn= 6200388.1666517  Ye= 2033470.5811409
Equator to Pole Quadrant =   10001965.7292773   by Wallis Integrals
================================================================================
    Long:   00            03            30            50            80   degs
--------------------------------------------------------------------------------
Lat
 00         0.0000        0.0000        0.0000        0.0000        0.0000 = Xn
            0.0000   334112.2018  3504812.8613  6455393.1487 15914266.8015 = Ye

 26   2876834.5726  2880284.6474  3254544.4618  4126961.4204  7838075.1819 = Xn
            0.0000   300438.2501  3089240.7342  5395834.1389  8907862.4295 = Ye

 52   5763343.5500  5767595.2929  6200388.1667  7028098.8358  9140726.0741 = Xn
            0.0000   206011.3235  2033470.5811  3273375.1633  4492302.6980 = Ye

 78   8661834.3195  8663617.7911  8837145.4593  9133107.0552  9765899.3835 = Xn
            0.0000    69628.2312   667590.2393  1027862.5424  1328925.3157 = Ye

 89   9890271.8643  9890424.9059  9905233.5494  9930166.0187  9982568.3829 = Xn
            0.0000     5845.3101    55845.5147    85563.2187   110002.2322 = Ye

 90  10001965.7293  ... Xn at Pole
```

A.5.3 Rate of Convergence

```
 Int 1924 Ellipsoid:     a = 6378 388.0     f = 1/297.0
 ======================================================
  Forward solution:   Geodetic to Gauss-Kruger (metres) Gp2GK
 ------------------------------------------------------------
  e2p   Lat 52 N        Long 3 E            Lat 52 N         Long 30 E
Order     Xn              Ye                  Xn               Ye
 ---  ---------        --------            ---------        ---------
 p=1  5767670.7        206014.5            6200505.8        2033490.0
   2  5767715.1        206021.2            6200529.4        2033568.3
   3  5767715.3131     206021.2480         6200529.356      2033568.763
   4  5767715.31371    206021.248213       6200529.35514    2033568.76508

   5  5767715.3137183  206021.24821416     6200529.3551360  2033568.7650943
   6  5767715.3137183  206021.24821416     6200529.3551360  2033568.7650943
```

Comments:
Convergence to full machine precision reached in 5 recurrence cycles.
Landen transformations not needed for faster convergence.
A practical break-out tolerance would be about 0.1 mm.

A.6 Fortran Notes

The *Fortran compiler* utilized is version F77, that runs under the Unix operating system.
F77 supports complex data types.
Double precision complex data type is: `COMPLEX*16`

`Z=DCMPLX( X,Y )` ... to compose complex number z = x + iy
`X=DREAL(Z)  Y=DIMAG(Z)` ... to extract the real & imaginary components of z.

Computations were run close to full machine precision, with many more decimal places than practically required, so that numerical results can be compared and verified almost exactly.

In this report, all floating-point variables are double precision.
i.e. `REAL*8` or `COMPLEX*16`

Names of variables and functions are chosen to resemble mathematical notation, for improved clarity.

Items that are part of the Fortran language are in capital letters.
User-defined functions and variable names are in mixed-case ("camel-case")
e.g. `SinPhi = CDSIN( Phi)`

The *non-standard Fortran* statement `IMPLICIT NONE` imposes a requirement that all variable names are to be explicitly declared.
(e.g. as `INTEGER, REAL*8, COMPLEX*16`).
Then the Fortran compiler will catch variable names misspelled, misnamed, or not declared.

Generic functions in the Fortran library are a convenience when changing data types in the code.
For example: `R= SQRT(S)` in single precision `REAL*4`
and for `R=DSQRT(S)` in double precision `REAL*8`
we can use `R= SQRT(S)`; the Fortran compiler will recognize `S` as double precision, and internally switch to `DSQRT`, not seen by the user.

In other words, Fortran recognizes the data type as double precision and switches to the appropriate specific function `DSQRT`, for the given generic function name `SQRT`.

For another example: `ABS(V)` may become `IABS(V), ABS(V) or DABS(V)`, dependng on `V` having declared type of `INTEGER, REAL*4, or REAL*8`

Not all functions of complex variables are provided in the Fortran function library, so that some complex variable functions need Fortran code prepared by the user.

A.7 Computation of Complex Variable Functions

In Fortran version F77 we have type `COMPLEX*16` to declare double precision complex variables and functions. Then we can have statements like `C=A*B` and `C=A/B` for arithmetic in complex variables.
Unlike F77, not all computing systems support the complex data type.
Where complex data typing is not available, the same effect is accomplished by subroutines working in two-component arithmetic, with real and imaginary components separated, something like:

`CALL DCMUL( Ar,Ai, Br,Bi, Cr,Ci )` for `C=A*B`
with: `Cr=Ar*Br -Ai*Bi;  Ci=Ai*Br + Ar*Bi`
or:
`CALL DCDIV( Ar,Ai, Br,Bi, Cr,Ci )` for `C=A/B`
with: `Denom= Br*Br + Bi*Bi`
and: `Cr=(Ar*Br + Ai*Bi)/Denom;  Ci=(Ai*Br - Ar*Bi)/Denom`

It gives more bulk to the code, since every single arithmetic or function step requires a subroutine call, except for add or subtract by components.

The complex function $z = \sin w = (e^{wi} - e^{-wi})/2i$
and its inverse by complex natural logarithms:
$\arcsin z = -i \log(i\,z + \sqrt{1 - z^2}\,)$ can be evaluated in complex arithmetic.
But by separating complex w into real and imaginary components by:
`u = DREAL(w);  v=DIMAG(w)`
then z = x+iy = sin w = sin(u + iv)= sin u cosh v + i cos u sinh v
In two-component real arithmetic: x=sin u cosh v; y=cos u sinh v
Then by `z= DCMPLX(x,y)` to compose the complex value z.

Similarly for the inverse:
$\arcsin z = \arcsin(x + iy) = \arcsin(A - B) -$ i $\ln[\,A + B + \sqrt{(A+B)^2 - 1}\,]$
where: $A = \frac{1}{2}\sqrt{(x+1)^2 + y^2}$ $B = \frac{1}{2}\sqrt{(x-1)^2 + y^2}$

Thus we can use `DREAL` and `DIMAG` to split a complex variable, do the arithmetic in two-component form with real numbers, and recombine the result into a complex number by `DCMPLX`.
The advantage then is that the source code for functions of complex variables is more readily adaptable to systems that do not support complex data typing.

A.8 Complex Variable Functions - Collected Formulae

CDsqrt: Complex Square Root

$$\sqrt{w} = \sqrt{(u + i\,v)} \; = \sqrt{(|w| + u)/2} + i\,\sqrt{(|w| - u)/2}\,; \quad |w| = \sqrt{u^2 + v^2}$$

CDexp: Complex Exponential

$$z = e^w = e^{u+iv} = e^u \cos v + i\,e^u \sin v$$

CDlog: Complex Natural Logarithm

$$w = \ln z = \ln(x + iy) = \ln|z| + i\,\psi_o; \quad |z| = \sqrt{x^2 + y^2}\,; \quad \psi_o = \mathrm{qatn}(x.y)$$

CDsin: Complex Sine

$$z = \sin w = (e^{iw} - e^{-iw})/2\,i$$

$$z = \sin(u + iv) = \sin u \cosh v + i\,\cos u \sinh v$$

CDasin: Complex ArcSin

$$\mathrm{w} = \mathrm{asin}\ \mathrm{z} = -\,i \ln(i\,z + \sqrt{1 - z^2}\,)$$

$$\mathrm{w} = \mathrm{asin}\ \mathrm{z} = \mathrm{asin}(A - B) - i\,\ln(A + B + \sqrt{(A+B)^2 - 1}\,)$$

$$A = \frac{1}{2}\sqrt{(x+1)^2 + y^2} \qquad B = \frac{1}{2}\sqrt{(x-1)^2 + y^2}$$

CDcos: Complex Cosine

$$z = \cos w = (e^{iw} + e^{-iw})/2$$

$$z = \cos(u + iv) = \cos u \cosh v - i\,\sin u \sinh v$$

CDtan: Complex Tangent

$$z = \tan w = \sin w \,/\, \cos w = i\,\frac{1 - e^{2iw}}{1 + e^{2iw}}$$

$$z = \tan w = \left(\frac{\sin 2u}{\cos 2u + \cosh 2v}\right) + i\left(\frac{\sinh 2v}{\cos 2u + \cosh 2v}\right)$$

CDatan: Complex ArcTan

$$\mathrm{w} = \mathrm{u} + \mathrm{iv} = \mathrm{atan}(\ \mathrm{z}) = \mathrm{atan}(\ \mathrm{x} + \mathrm{iy})$$

$$\mathrm{u} = [\ \mathrm{qatn}(\mathrm{x},1{+}\mathrm{y}) + \mathrm{qatn}(\ \mathrm{x},1{-}\mathrm{y})\ ]\ /2$$

$$\mathrm{iv} = \mathrm{i}\ \ln\ [\ \ (x^2 + (1+y)^2\,)\,/\,(x^2 + (1-y)^2\,)\ \ ]\ /4$$

`CDsinh`: Complex Hyperbolic Sine

$$z = \sinh w = (e^w - e^{-w})/2$$

$$\sinh(u + iv) = \sinh u \cos v + i \ \cosh u \sin v$$

`CDcosh`: Complex Hyperbolic Cosine

$$\cosh w = (e^w + e^{-w})/2$$

$$\cosh(u + iv) = \cosh u \cos v + i \ \sinh u \sin v$$

`CDtanh`: Complex Hyperbolic Tangent

$$z = \tanh w = \sinh w / \cosh w = \frac{e^{2w} - 1}{e^{2w} + 1}$$

$$z = \tanh(u + iv) = \left(\frac{\sinh 2u}{\cosh 2u + \cos 2v}\right) + i \left(\frac{\sin 2v}{\cosh 2u + \cos 2v}\right)$$

`CDatanh`: Complex Hyperbolic ArcTanh

$$w = \text{atanh z} = \frac{1}{2}\ln\left(\frac{1+z}{1-z}\right)$$

`Newton-Raphson` for Complex Variable Functions

The Newton-Raphson method also works for complex variables.

For z = f(w):
To find w, given z, form the function F(w) = f(w) - z = 0

then iteration steps: $w_{i+1} = w_i$ - F(w) / F'(w) = $w_i - \Delta w_i$
repeated until $|\Delta w_i| \leq$ tolerance.

F'(w) = derivative of F(w).

To find a suitable start value for the iteration, it is sometimes necessary to map out an overall sketch of the function.

B.1 Code Modules List

```
c   -- Fortran Modules in hierarchy order
c...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
! TESTMAIN  .. program          ... main program to test the coded algorithms
!                                  ========================================
!    CDASIN    .. function          ... complex double arcsin
!    CDATAN    .. function          ... complex double arctan
!    CDTANH    .. function          ... complex double hyperbolic tangent
!    GK2GP     .. subroutine        ... Gauss-Kruger to Geodetic
!       EGUD      .. subroutine     ... inverse  complex eLambertian
!          ATANH     .. function    ... real hyperbolic arctan                        !          (
!          CDTANH    .. function    ... complex double hyperbolic tangent
!
!       ELAM      .. subroutine     ... complex eLambertian
!          ATANH     .. function    ... real hyperbolic arctan
!
!       INVEINT3M .. subroutine     ... inverse complex double elliptical integral
!          EINT3M    .. subroutine  ... complex double elliptical integral  3rd kind
!
!    GP2GKSC   .. subroutine        ... Geodetic to GK + Scale factor + Convergence
!       GP2GK     .. subroutine     ... Geodetic to Gauss-Kruger,
!          EGUD      .. subroutine  ... inverse  complex eLambertian
!             ATANH     .. function
!             CDASIN    .. function
!             CDTANH    .. function
!
!          EINT3M    .. subroutine
!          ELAM      .. subroutine
!             ATANH     .. function
!
!       RADII     .. subroutine     ... radii of curvature on ellipsoid
!
!    MERDISL   .. subroutine        ... meridian arc distance by Wallis integrals
!    MERDIST   .. subroutine        ... meridian arc distance by  trigonometric series
!
c...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
```

B.2 Fortran Code - Subroutines and Functions

B.2.1 Main Program - Test Module

```
       PROGRAM TESTMAIN
C     =================
C     -- test run on  Klotz' algorithm

         IMPLICIT NONE
      REAL*8  Phi,  Dlon, Xn,Ye, Psf,Conv
      REAL*8  Ae,Be,FL, E2, E
      REAL*8  Pi, D2R, R2D
      REAL*8  Quad

      Pi=4d0*ATAN(1d0)
      D2R=Pi/180d0
      R2D=180d0/Pi

C     -- International Hayford 1924
      Ae= 6378 388 d0
      FL= 1d0/297d0
      Be= Ae*(1d0-FL)
      Print *, " International 1924 Ellipsoid ( Hayford)"
      Print *

C     -- WGS-84
C     Ae= 6378137 d0
C     Be= 6356752.314 d0
C     FL= (Ae-Be)/Ae
C     Print *, " WGS-84 Ellipsoid"

      E2= FL*(2d0-FL)
      E = SQRT(E2)

      Print *, " Ellipsoid major semi-axis a =", Ae," metres"
      Print *, " Inverse Flattening  1/f = ", 1d0/FL

C     -- meridian arc quadrant equator to pole
      CALL MERDISL( Ae,FL, Pi/2d0, Quad )       ! by Wallis integrals
      Print *, " Equator to Pole Quadrant= ", Quad

      Phi=  52d0*D2R         ! test point
      Dlon= 30d0*D2R

      Print *
      Print *, " Forward solution: geodetic to Gauss-Kruger - Gp2GkSc "
      Print *
      Print *, " Latitude  Phi = ",  Phi*R2D
      Print *, " Longitude Dlon= ", Dlon*R2D
      Print *
```

```
CALL Gp2GkSc(Ae, FL, Phi,Dlon, Xn,Ye , Psf, Conv )
Print *, " Gauss-Kruger Xn= ", Xn,"  Ye= ", Ye
Print *, " Point Scale Factor = ", Psf
Print *, " Meridian Convergence = ", Conv*R2D, " degrees"

Print *
Print *, " Inverse solution: Gauss-Kruger to geodetic - Gk2Gp "
CALL Gk2Gp( Ae, FL, Xn,Ye, Phi,Dlon )

Print *, "Xn=", Xn, "     Ye=", Ye
Print *, "Phi =",  Phi*R2D
Print *, "Dlon=", Dlon*R2D

STOP   "‘End testmain"’
END
```

B.2.2 Gp2GkSc - Geodetic to Gauss-Kruger+Scale Factor+Convergence

```
      SUBROUTINE Gp2GkSc( Ae, FL, Phi,Dlon, Xn,Ye, Psf, Conv )
C     ==================
C     -- Geodetic to Gauss-Kruger generic Transverse Mercator
C     -- with point scale factor and meridian convergence

C     -- Phi  = input geodetic latitude  ( radians)
C     -- Dlon = input geodetic longitude ( radians)
C     -- Psi  =  complex isometric latitude
C     -- W    = "complex intermediate latitude"
C     -- Z    = unitary complex Gauss-Kruger, for Ae =1.0
C     -- GK   = complex Gauss-Kruger ( metres)
C     -- Xn,Ye= Gauss-Kruger Northing & Easting

         IMPLICIT NONE
      REAL*8      Ae,FL, E2, Pi, D2R, R2D
      REAL*8      Phi, Dlon,  Xn,Ye,  Psf,Conv
      REAL*8      Dp,Ds,Dm, Rm,Rn, Ta, Co, Si
      REAL*8      Xnp,Xnm,Dxn, Yep,Yem,Dyea, Qatnf
      COMPLEX*16  Grad

      Pi = 4d0*ATAN( 1d0)
      R2D = 180d0/Pi
      D2R = Pi/180d0
      E2 = ( 2d0-FL)*FL

C     --  finite differences Dxn Dye for scale factor
      DP = 0.00001d0*D2R
      CALL Gp2Gk( Ae,FL, Phi+DP, Dlon, Xnp, Yep, Grad )
      CALL Gp2Gk( Ae,FL, Phi-DP, Dlon, Xnm, Yem, Grad )
      Dxn = Xnp-Xnm
      Dye = Yep-Yem

 C    -- meridian convergence by derivatives - Grad
      CALL Gp2Gk( Ae, FL, Phi,Dlon, Xn,Ye, Grad )
      Conv =  -Qatnf( DIMAG(Grad), DREAL(Grad) )
      Ta =  DIMAG(Grad)/ DREAL(Grad)
      Co =  1d0/ Sqrt( 1d0 + Ta*Ta)
      Si =  Co*Ta

 C     -- scale factor by finite differences
      CALL RADII(Ae,FL,Phi, Rm,Rn)
      Ds = Dxn*Co + Dye*Si
      Dm = ( Dp+Dp)*Rm
      Psf = Ds/Dm

      RETURN
      END
```

B.2.3 Gp2Gk - Geodetic to Gauss-Kruger

```
      SUBROUTINE Gp2Gk( Ae,FL, Phi,Dlon, Xn,Ye, Grad )
C     ================
C     -- Geodetic to Gauss-Kruger generic Transverse Mercator

C     -- Phi  = input geodetic latitude  ( radians)
C     -- Dlon = input geodetic longitude ( radians)
C     -- Q    = isometric latitude
C     -- Psi  = complex isometric latitude
C     -- W    = complex intermediate latitude
C     -- Z    = unitary complex Gauss-Kruger
C     -- Xn,Ye = Gauss-Kruger Northing & Easting

         IMPLICIT NONE
      REAL*8      Phi, Dlon, Xn,Ye, Ae,FL, E,E2, Q
      COMPLEX*16  Psi, Dpsi, W, DW, DZW, Z, Grad

      E2 = FL*(2d0-FL)
      E = SQRT( E2)
      CALL eLam( E, DCMPLX( Phi, 0d0), Psi, Dpsi )
      Q = DREAL( Psi)

      Psi = DCMPLX( Q, Dlon)
      CALL eGud( E, Psi, W, Dw )    ! elliptical Gudermannian, inverse eLam
      CALL Eint3M( E, W, DZW, Z)    ! elliptical integral 3rd kind ( modified)
      Grad = (1d0-E2)*Dpsi*Dw*DZW   ! derivative

      Xn =  Ae*(1d0-E2)*DREAL(Z)    ! Gauss-Kruger Northing Xn
      Ye =  Ae*(1d0-E2)*DIMAG(Z)    !              Easting  Ye

c     Print *
c     Print *, "GK: Xn,Ye =", Xn,Ye
c     Print *, "Dpsi =", Dpsi
c     Print *, "Dw =", Dw
c     Print *, "DZw =", DZW
c     Print *, "Grad =", Grad, CDABS( Grad)

      RETURN
      END
```

B.2.4 Gk2Gp - Gauss-Kruger to Geodetic

```
       SUBROUTINE Gk2Gp( Ae, FL, Xn,Ye, Phi,Dlon )
C      ================
C      -- generic Gauss-Kruger to geodetic

C      -- Xn,Ye= input Gauss-Kruger Northing & Easting
C      -- Phi  = output geodetic latitude  ( radians)
C      -- Dlon = output geodetic longitude ( radians)
C      -- Q    =  isometric latitude
C      -- Psi  =  complex isometric latitude
C      -- W    = "complex intermediate latitude"
C      -- Z    = unitary complex Gauss-Kruger

         IMPLICIT NONE
       REAL*8  Phi,  Dlon,  Xn,Ye, Ae,FL, E,E2, Q
       COMPLEX*16  Psi, Dpsi, W,DZW,Z, Temp, Dphi

       E2 = ( 2d0-FL)*FL
       E = SQRT( E2)
       Z = DCMPLX( Xn/Ae,Ye/Ae) / (1d0-E2)

       CALL InvEint3M( E, Z, W, DZW)      ! inverse of elliptical integral

       CALL eLam( E, W, Psi, Dpsi )       ! elliptical Lambertian, eLam
       Q  =   DREAL( Psi)
       Dlon = DIMAG( Psi)

       CALL eGud( E, DCMPLX( Q, 0d0), Temp, Dphi ) ! inverse eLambertian
       Phi = DREAL( Temp)

c      Print *, " Gk2Gp.  Xn,Ye", Xn,Ye
c      Print *, " Gk2Gp: Phi,Dlon",  Phi, Dlon

       RETURN
       END
```

B.2.5 eLam - Complex Lambertian

```
      SUBROUTINE eLam( E, Phi,Psi, Dpsi )
C     ===============
C     -- Complex Lambertian function for complex isometric latitude Psi
C     -- modified for ellipticity e

         IMPLICIT NONE
      REAL*8  E, E2,Qr, Qi, Tol
      COMPLEX*16  Phi, SinPhi, CosPhi, Psi, Dpsi
      COMPLEX*16  CDATANH, CDSIN, CDCOS

c     Print *,"eLam.i: Phi = ", Phi
      SinPhi = CDSIN( Phi)
      CosPhi = CDCOS( Phi)
      Psi = CDATANH( SinPhi) - E*CDATANH( E*SinPhi)

C     -- hard limit to catch polar exception
      Qr  = DREAL( Psi)
      Qi  = DIMAG( Psi)
      Tol = 1d-15
      IF ( Qr .gt. 25d0) THEN
         Qr = 25d0
         Qi = Tol
      ENDIF
      Psi = DCMPLX( Qr,Qi)

C     -- derivative
      E2 = E*E
      Dpsi = (1d0-E2)/ ( CosPhi*(1d0-E2*SinPhi*SinPhi ))

c     Print *,"eLam: Psi = ",  Psi
c     Print *,"eLam Dpsi = ", Dpsi
      RETURN
      END        ! tested OK
```

B.2.6 eGud - Inverse Lambertian

```
      SUBROUTINE eGud( E,Psi, Phi, Dphi)
C     ==============
C     -- complex inverse Lambertian = Gudermann function
C     -- modified for ellipticity e

         IMPLICIT NONE
      REAL*8      E,E2, Test,Tolerance
      COMPLEX*16  Psi, SinPhi, CosPhi, Phi, Dphi
      COMPLEX*16  Pst, SinPhL
      COMPLEX*16  CDASIN, CDATANH, CDTANH

c     Print *," egud psi = ", Psi
      Tolerance = 1d-16
      E2 = E*E


      SinPhi = CDTANH( Psi)
  100 CONTINUE              ! iteration loop 100
      Pst = Psi + E*CDATANH( E*SinPhi)
      SinPhL = SinPhi
      SinPhi = CDTANH( Pst)
      Test =  CDABS( SinPhi-SinPhL)

c     Print *, "egud: SinPhi,test", SinPhi, Test
      IF ( Test .gt. Tolerance) GOTO 100

      Phi = CDASIN( SinPhi)

C     -- derivative
      CosPhi = CDCOS( Phi)
      Dphi = CosPhi*( 1d0-E2*SinPhi*SinPhi)/( 1d0-E2)

c     Print *," egud Phi = ",  Phi
c     Print *,"egud Dphi = ", Dphi
      RETURN
      END        ! tested Ok
```

B.2.7 Eint3M - Complex Integral E3

```
      SUBROUTINE Eint3m( E, W, DzW, Z)
C     =================
C     -- double precision complex meridian arc length on ellipsoid
C     -- based on Wallis integrals by recurrence formula
C     -- by principle of analytic continuation
C     -- W is intermediate complex latitude
C     -- complex arc Z = Gauss-Kruger  unitary X & Y

         IMPLICIT NONE
      REAL*8  E, E2,Tol, Rf,Fp, P2

      COMPLEX*16  W,DzW, CDRoot, Z
      COMPLEX*16  CosW,SinW, Sin2P, Sum,Term, Wsin2p

      E2 = E*E
      SinW = CDSIN(W)
      CosW = CDCOS(W)
      Rf = +1d0        ! for Third kind modified
      Fp = +1d0
      P2 = 0d0
      Sin2P = SinW
      Wsin2p = W
      Tol = 1d-16
      Sum = W
  100 CONTINUE       ! -- iteration loop 100
         P2 = P2+2d0
         Rf = Rf+2d0
         Fp = +Fp*Rf*E2/P2
         Wsin2p = ( (P2-1d0)*Wsin2p - CosW*Sin2P) / P2
         Term = Fp*Wsin2p
         Sum = Sum+Term
         Sin2P = Sin2P*SinW*SinW
         IF ( CDABS( Term) .gt. Tol ) GOTO 100
      Z = Sum

C     -- derivative Dzw
      SinW = CDSIN( W)
      CDRoot = CDSQRT( 1d0-E2*SinW*SinW )
      DzW =  1d0/(CDRoot*CDRoot*CDRoot)      ! E3 derivative
      RETURN
      END
```

B.2.8 InvEint3M - Inverse of Integral E3

```
      SUBROUTINE InvEint3M( E, Z, W,DZW )
C     ====================
C     -- inverse of elliptical integral 3rd kind E3

         IMPLICIT NONE
      INTEGER NIT
      REAL*8  E, Tol
      COMPLEX*16  Z,W, DW, DZW, Zt

      Tol= 1d-15
      W= Z

c     -- iterative refinement by Newton-Raphson
      NIT= 0
  100 NIT= NIT+1              ! -- iteration loop 100
      CALL Eint3M( E,W, DZW, Zt)
      DW=  ( Z-Zt)/DZW
      W= W+DW
      IF ( CDABS( DW) .gt. Tol ) GOTO 100

c     Print *, " W= ", W
c     Print *, "dw= ", DW
c     Print *, "InvEint3M  Nit= ", NIT

      RETURN
      END
```

B.2.9 Sp2Gk - Spherical Lat & Long to Gauss-Kruger

```
      SUBROUTINE Sp2Gk( R, Phi, Dlon, Xn,Ye, Psf,Conv)
C     ================
C     -- spherical lat/long to Gauss-Kruger TM

C     -- R         Sphere radius
C     -- Phi,Dlon  Spherical Lat & Long    ( radians)
C     -- Xn,Ye     GK Northing & Easting   { metres)
C     -- Psf       Point Scale factor
C     -- Conv      Meridian convergence (radians)

         IMPLICIT NONE
      REAL*8  R, Phi,Dlon, Xn,Ye, Psf,Conv
      REAL*8  CosPhi,SinPhi, CosDlon, SinDlon
      REAL*8  CpSdl, ATANH, Qatnf

      CosPhi = COS( Phi)
      SinPhi = SIN( Phi)
      CosDlon= COS( Dlon)
      SinDlon= SIN( Dlon)

      Xn = R*Qatnf( SinPhi, CosPhi*CosDlon )
      Ye = R*ATANH( CosPhi*SinDlon)

C     -- point scale factor
      CpSdl = CosPhi*SinDlon
      Psf  = 1d0 / SQRT( 1d0 - CpSdl*CpSdl)

C     --  meridian convergence
      Conv = Qatnf( SinDlon*SinPhi, CosDlon )

      RETURN
      END
```

B.2.10 Gk2Sp - Gauss-Kruger to Spherical Lat & Long

```
      SUBROUTINE Gk2Sp( R, Xn,Ye, Phi, Dlon)
C     ================
C     -- Gauss-Kruger to spherical Lat & Long

C     -- R         Sphere radius        ( metres)
C     -- Xn,Ye     Northing & Easting   { metres)
C     -- Phi,Dlon  Spherical Lat & Long ( radians)

         IMPLICIT NONE
      REAL*8  R, Xn,Ye, Phi,Dlon
      REAL*8  SinX, CosX, SinhY, CoshY, Qatnf

      IF ( Xn/R .ge. 1d0 ) Xn = R
      SinX  = SIN( Xn/R)
      CosX  = COS( Xn/R)
      SinhY = SINH( Ye/R)
      CoshY = COSH( Ye/R)

      Phi  =  ASIN( SinX / CoshY)
      Dlon = Qatnf( SinhY, CosX)

      RETURN
      END
```

B.2.11 Radii - Ellipsoid Curvatures Rm & Rn

```
       SUBROUTINE Radii(Ae,FL,Phi, Rm,Rn)
c      ================
c      -- ellipsoid radius of curvature, in meridian & normal

          IMPLICIT NONE
       REAL*8  Ae,FL,Phi, Rm,Rn
       REAL*8  E2,VV, Sp

       E2 = (2d0-FL)*FL
       Sp = SIN(Phi)
       VV = 1d0 - E2*Sp*Sp

       Rn = Ae/SQRT(VV)
       Rm = Rn*(1d0-E2)/VV

       RETURN
       END
```

B.2.12 MerDisL - Meridian Arc by Wallis Integrals

```
      SUBROUTINE MerDisL(Ae, FL, Phi, ArcS)
C     ==================
C     -- precise  meridian arc distance on ellipsoid
C     -- by  Wallis integrals Wsin2P

         IMPLICIT NONE
      REAL*8  Ae,FL, Phi,ArcS, Tol
      REAL*8  E2,SinPhi,CosPhi,Sin2P,F2n, Sum,Fp,Term
      REAL*8  Wsin2P

      E2 = (2d0-FL)*FL
      SinPhi = SIN(Phi)
      CosPhi = COS(Phi)
      Fp  = 1d0
      F2n = 0d0
      Wsin2P = Phi
      Sin2P  = SinPhi

      Tol = 1d-14
      Sum = Phi
  100 CONTINUE
         F2n = F2n+ 2d0
         Fp  = Fp*E2*(F2n+1d0)/(F2n)
         Wsin2P = (( (F2n-1d0)*Wsin2P - CosPhi*Sin2P )) / F2n
         Term = Fp*Wsin2P
         Sum  = Sum+Term
         Sin2P= Sin2P*SinPhi*SinPhi
         IF ( ABS( Term) .gt. Tol ) GOTO 100

      ArcS = Ae*(1d0-E2)*Sum
c     Print *, "merdisl", Phi, ArcS

      RETURN
      END         !  tested OK
```

B.2.13 MerDisT - Meridian Arc by Power Series

```
      SUBROUTINE MerDisT(Ae,FL, Phi, ArcS)
C     ==================
C     -- meridian arc distance on ellipsoid
C     -- by series formulae to power e**8

        IMPLICIT NONE
      REAL*8  Ae, FL, Phi, ArcS
      REAL*8      E2, E4, E6, E8
      REAL*8  A0, A2, A4, A6, A8
      REAL*8     Sp2,Sp4,Sp6,Sp8

      E2=(2d0-FL)*FL
      E4=E2*E2
      E6=E2*E4
      E8=E2*E6

      A0= E2*1d0/4d0+E4*3d0/64d0 +E6*5d0/256d0+E8*125d0/16384d0

      A2= E2*3d0/8d0+E4*3d0/32d0 +E6*3d0/64d0 +E8*105d0/4096d0

      A4= E4*15d0/256d0  +E6*45d0/1024d0  +E8*151d0/1024d0

      A6= E6*35d0/3072d0 +E8*175d0/12288d0

      A8= E8*315d0/131072d0

      Sp2= SIN( Phi*2d0 )
      Sp4= SIN( Phi*4d0 )
      Sp6= SIN( Phi*6d0 )
      Sp8= SIN( Phi*8d0 )

      ArcS= Ae*( (1d0-A0)*Phi - A2*Sp2 + A4*Sp4 - A6*Sp6 + A8*Sp8 )

      RETURN
      END
```

B.2.14 Atanh - Hyperbolic Arc Tan

```
       REAL*8 FUNCTION Atanh( X )
C      ====================
C      -- arctanh = inverse hyperbolic tanh

          IMPLICIT NONE
       REAL*8  X, T, ATANH

       T= ( 1d0+X)/(1d0-X)
       ATANH=LOG( T)/2d0
       RETURN
       END
```

B.2.15 CDasin - Complex Arc Sin

```
      COMPLEX*16 FUNCTION CDASIN(Z)
C     ==========================
C     -- double precision complex arcsin

         IMPLICIT NONE
      COMPLEX*16  Z, CDASIN
      REAL*8  X,Y, Xp,Xm,YY,DL,DR, Alfa, Beta, U,V

      X = DREAL(Z)
      Y = DIMAG(Z)
      YY = Y*Y
      Xp = (X+1d0)*(X+1d0)
      Xm = (X-1d0)*(X-1d0)

      DL = SQRT(Xp+YY)
      DR = SQRT(Xm+YY)
      Alfa = (DL+DR)/2d0
      Beta = (DL-DR)/2d0

      U = ASIN(Beta)
      V = LOG(( Alfa + DSQRT( (Alfa+1d0)*(Alfa-1d0) ) ))
      IF ( V .lt. 0d0) V =  -V

      CDASIN = DCMPLX(U,V)
      RETURN
      END
```

B.2.16 CDatan - Complex Arc Tan

```
      COMPLEX*16 FUNCTION CDATAN(Z)
C     ============================
C     -- complex double arctan

         IMPLICIT NONE
      COMPLEX*16  Z, CDATAN
      REAL*8  X,Y,XY2,Y2,Qatnf, U,V

      X = DREAL(Z)
      Y = DIMAG(Z)
      XY2 = X*X + Y*Y
      Y2 = Y + Y

      U = (( Qatnf( X, 1d0+Y) + Qatnf( X,1d0-Y) )) /2d0
      V = LOG(( (XY2 + Y2 + 1d0) / ( XY2 - Y2 + 1d0) )) /4d0

      CDATAN = DCMPLX(U,V)
      RETURN
      END
```

B.2.17 CDtanh - Complex Hyperbolic Tan

```
      COMPLEX*16 FUNCTION CDTANH(Z)
C     ============================
C     -- complex double hyperbolic tangent

         IMPLICIT NONE
      COMPLEX*16  Z, CDTANH
      REAL*8  XX,YY, U,V, Div

      XX = 2d0*DREAL(Z)
      YY = 2d0*DIMAG(Z)

      Div = COSH(XX)+COS(YY)
      U =  SINH(XX)/Div
      V =  SIN (YY)/Div

      CDTANH =  DCMPLX( U,V)
      RETURN
      END
```

End of Document LaTeX