XXX-TBD-TBD-YYY

Draft of 2014.10.26

ICE, CLOUD, and Land Elevation Satellite
(ICESat-2) Project

Algorithm Theoretical Basis Document
(ATBD)
for

Atmospheric delay correction
to laser altimetery ranges

Prepared By: L. Petrov /Code: 698

Goddard Space Flight Center
Greenbelt, Maryland

National Aeronautic and
Space Administration

CHECK https://icesatiimis.gsfc.nasa.gov
TO VERIFY THAT THIS IS THE CORRECT VERSION PRIOR TO USE

CM Foreword

This document is an Ice, Cloud, and Land Elevation (ICESat-2) Project Science Office controlled document. Changes to this document require prior approval of the Science Development Team ATBD Lead or designee. Proposed changes shall be submitted in the ICESat-II Management Information System (MIS) via a Signature Controlled Request (SCoRe), along with supportive material justifying the proposed change.

`In this document, a requirement is identified by "shall," a good practice by "should," permission by "may" or "can," expectation by "will," and descriptive material by "is."`

Questions or comments concerning this document should be addressed to:

ICESat-2 Project Science Office
Mail Stop 615
Goddard Space Flight Center
Greenbelt, Maryland 20771

Preface

This document is the Algorithm Theoretical Basis Document for the TBD processing to be implemented at the ICESat-2 Science Investigator-led Processing System (SIPS). The SIPS supports the ATLAS (Advance Topographic Laser Altimeter System) instrument on the ICESat- 2 Spacecraft and encompasses the ATLAS Science Algorithm Software (ASAS) and the Scheduling and Data Management System (SDMS). The science algorithm software will produce Level 0 through Level 4 standard data products as well as the associated product quality assessments and metadata information.

The ICESat-2 Science Development Team, in support of the ICESat-2 Project Science Office (PSO), assumes responsibility for this document and updates it, as required, as algorithms are refined or to meet the needs of the ICESat-2 SIPS. Reviews of this document are performed when appropriate and as needed updates to this document are made. Changes to this document will be made by complete revision.

Changes to this document require prior approval of the Change Authority listed on the signature page. Proposed changes shall be submitted to the ICESat-2 PSO, along with supportive material justifying the proposed change.

Questions or comments concerning this document should be addressed to:

Thorsten Markus, ICESat-2 Project Scientist
Mail Stop 615
Goddard Space Flight Center
Greenbelt, Maryland 20771

1. Introduction

This document covers calibration of laser altimeter for effects of propagation in the atmosphere. The input of the algorithm are 1) the output four-dimensional numerical weather model that contains air temperature, pressure, and air humidity; 2) footprint geolocation; and 3) azimuth and elevation of the spacecraft viewed from the footprint. The output of the algorithm is an increase of light propagation time through the atmosphere with respect to propagation in vacuum.

2. Overview and background information

The ICESat-II laser altimeter measures travel time of light that is emitted by the spacecraft, reflected by the surface and bounced back to the satellite. In the first approximation the travel time is the sum of distances from the emitter to the surface and from the surface to the receiver multiplied by the speed of light in vacuum. Since the satellite is moving with respect to the surface, the position of emitter and the receiver are different, but they can be easily computed from the parameters of the orbit. In the second approximation the effect of propagation of light in the atmosphere should be accounted for. The speed of light of the photon pulse, so-called group speed, depends on the density and composition of the gaseous medium: light travels slower in a denser gas. The relative reduction of speed of light v in the medium with respect to the speed of light in vacuum c is conveniently to express as $(c - v)/c$ and it will be called thereafter refractivity r. Gas density depends on temperature, pressure, and mixing ratios of gas components. The mixing ratios of all major gas component are fairly constant, except the mixing ratio of water vapor and ozone. Though, the contribution of ozone to refractivity does not exceed 10^{-6} of the total refractivity, which has a negligible effect on ICESat data.

Thus, in order to compute air refractivity, one needs to know the density of dry air and density of water vapor along the trajectory of light propagation. These densities depend on three parameters: temperature of moist air, total atmospheric pressure, and partial pressure of water vapor. In the presented algorithm these parameters will be computed using the output of numerical weather model GEOS-FPIT developed by the Global Modeling and Assimilation Office at the Goddard Space Flight Center. The model output can be used for deriving the state of the atmosphere at a four-dimensional grid over elevation level with respect to the geoid, longitude, geodetic latitude, and time. Air refractivity is computed for each element of the grid from the values of air temperature of moist air, total pressure, and the partial pressure of water vapor. Then, using the set of gridded *discrete* values of refractivity, the coefficients of the expansion of the refractivity field over a set of basic functions are computed. These interpolating functions determine the refractivity field as a *continuous* four-dimensional function in the bounded regions where the atmosphere is confined.

Using this expansion, the refractivity along the predicted light trajectory considered as a straight line is computed. It will be shown that we can neglect bending in the atmosphere for processing ICESat II data, since zenith angle of the light trajectory is supposed to be within 5 degrees. Integrating the refractivity along the trajectory gives us path delay in the atmosphere. Using this path delay, the estimate of the footprint altitude is corrected.

Presented algorithm has the internal precision better than 1 mm. Its accuracy is determined by the accuracy of numerical weather models.

2.1. Parameters that describe the footprint

The following parameters describe the footprint for the purpose of atmospheric path delay computation:

- h_{ell} — altitude above the WGS-84 reference ellipsoid in meters;
- φ — geodetic latitude in degrees;
- λ — longitude in degrees;
- t – time stamp of the pulse emitted by the spacecraft in TAI time scale;

- z — zenith angle of the spacecraft as it viewed from the footprint in degrees;
- A — azimuth of the spacecraft as it viewed from the footprint counted from north to east in degrees.

2.2. Parameters that describe state of the atmosphere

2.2.1. Grid parameters

- h_i — height of the the ith level with respect to the ground level in meters;
- φ_j — geodetic latitude in degrees;
- λ_k — longitude in degrees;
- t_l — UTC time tag.

2.2.2. Output variables of the numerical weather model that describes the state of the atmosphere

- Φ — Geopotential of the model surface in m^2/s^2;
- D_i — thickness of the ith layer in Pa: difference in pressure between the lower edge of the $i+1$ layer an the lower edge of the ith layer in Pa;
- T_i — air temperature at the ith layer in K;
- q_i — specific humidity of the ith layer.

NB: the height of the model layers depends on longitude and latitude and is not explicitly defined in the model.

2.2.3. Intermediate variables that describe the state of the atmosphere at a regular grid

- P_i — atmospheric pressure at ith level in Pa;
- T_i — air temperature at the ith level in K;
- P_{wi} — partial pressure of water vapor of the ith level in Pa.

NB: the height of the levels does not depend on longitude and latitude.

3. Algorithm theory

The algorithm processes the data in the following sequence:

1. The following parameters are extracted from the input files: pressure layer thickness, air temperature and specific humidity at the original, irregular, and non-uniform 3D grid. Height of vertical levels is not explicitely defined.
2. Each array of extracted pressure layer thickness, and specific humidity is transformed to total pressure and partial pressure of water vapor by solving the hypsometric differential equation. Total pressure, air temperature, and partial pressure of water vapor are projected to the *regular* vertical grid that stats at -1000 meters below the reference ellipsoid and ends at 90000 meter.
3. For a chunk of ICESat data to be processed (usually one day), a 4D subset pararameters of the state of the atmosphere (total pressure, air temperature, and partial pressure of water vapor) that covers all latitudes, all longitudes, all layers and a number of time epochs within the time interval of ICESat data, and additionally two epochs before and two epochs after the last epochs, is extracted. Air Refractivity is computed for each element of this 4D grid.
4. Coefficients of the expansion of the 4D subset of the refractivity fields over a tensor products of 4D B-splines of the 3rd degree are computed.
5. Using thes coefficients, the path delay in zenith direction is computed for each laser shot by integration air refractivity between the expected position of the footprint and the nominal top of the atmosphere set to the height 90,000 meters. The first partial derivative of the path delay with respect to the height of the predicted footprint is computed as well.
6. Using estimates of path delay and its partial derivative over time for the predicted position of the footprint and the result of range measurement, the final value of the range corrected for delay in the atmosphere is computed using an iterative procedure.

3.1. Physics of Problem

In order to compute path delay through the atmosphere, we need to know the state of the atmosphere, i.e. total pressure, partial pressure of water vapor, and air temperature as function of three components of coordinates and time. Output of numerical weather models does not provide the state of the atmosphere explicitely, but it can be deduced from these data.

3.1.1. Computation of the global field of refractivity from the output of a numerical weather model

The GEOS-FPIT model provides the output of a number of variables at 72 pressure layers that, among other parameters, include pressure thickness, specific humidity and air temperature at the middle of a layer. The upper edge of the highest pressure layer is fixed to 1 Pa. The bottom edge of the lowest pressure layer coincides with the Earth's surface. The pressure layers are indexed from top to bottom. The height of layers of the vertical grid is not fixed and depends on both latitude, longitude, and time.

3.1.2. Determination of the state of the atmosphere from the output of a numerical weather model

The hydrostatic equilibrium equation

$$\frac{\partial P}{\partial h} = -\rho(h)\, g(h) \tag{1}$$

relates the partial derivatives of the total pressure P over geometric height with air density ρ and gravity acceleration g. The state equation of moist air endorsed by the International Committee for Weights and Measures (CIPM-2007) (Picard et al., 2007) relates the air density with the total pressure, partial pressure of water vapor P_w with air absolute temperature T:

$$\rho_m = \left((P - P_w)\, M_d + P_w\, M_w \right) \frac{Z^{-1}(P, P_w, T)}{RT} \tag{2}$$

where M_d and M_w are molar mass of dry air and water vapor respectively, R is the universal gas constant, and Z^{-1} is the inverse compressibility of moist air. According to Picard et al. (2007), the compressibility depends on pressure and temperature the following way:

$$\begin{aligned} Z(P, P_w, T) \;=\; & 1 \\ & - \frac{P}{T} \left(a_0 + a_1 (T - T_0) + a_2 (T - T_0)^2 \right) \\ & + \frac{P_w}{T} \left(b_0 + b_1 (T - T_0) \right) \\ & + \frac{P_w^2}{PT} \left(c_0 + c_1 (T - T_0) \right) \\ & + \frac{P^2}{T^2}\, e_0 \\ & + \frac{P_w^2}{T^2}\, f_0 \end{aligned} \tag{3}$$

where $T_0 = 273.15\mathrm{K}$ is the freezing temperature. Numerical values of coefficients in equations 2–3, as well as numerical values of other constants can be found in section 7.

Combining equations 1 and 2, we write the differential state equation of the atmosphere in the hydrostatic equilibrium as

$$\frac{dh}{dP} = \frac{R\, T(P)}{g(P)\, Z^{-1}(P, P_w, T) \left(M_d\, (P - P_w) \;+\; M_w\, P_w \right)}. \tag{4}$$

For solving this differential equation, we need to define an initial value. The output of GEOS-FPIT model provides a 2D gridded quantity Φ called "surface geopotential". Then the surface height h_s can be computed as

$$h_s(\varphi, \lambda) = h_g(\varphi, \lambda) + \frac{\Phi(\varphi, \lambda)}{g_n}. \tag{5}$$

Here h_g is the geoid undulation (height of the geoid above the reference ellipsoid) and g_n is the nominal gravity acceleration used in the numerical weather model. Geoid height at a given point is found from the global field of undulations[1] computed from the EGM2008 gravity field (*Pavlis et al.*, 2012). This gridded undulation field is expanded into 2D B-spline basis. Since the geoid undulations do not depend on time, this computation is done only once and the interpolation coefficients are saved for re-use. Details of computations are provided in Appendix B. These B-spline expansion coefficients are used for interpolation of the geoid height.

In order to solve equation 4, we need to know function $g(P)$. It dependence on height is well known, for instance, *Li and Gotze* (2001):

$$\begin{aligned} g(\varphi,h) = \quad & g_{eq}\frac{1+k\sin^2\varphi}{\sqrt{1-(2f_\oplus - f_\oplus^2)\sin^2\varphi}}\times \\ & \left[1-\frac{2}{R_\oplus}\left(1+\frac{\Omega_\oplus^2 R_\oplus^3(1-f_\oplus)}{GM_\oplus}+f_\oplus\left(1-2\sin^2\varphi\right)\right)h+\frac{3}{R_\oplus^2}h^2\right]. \end{aligned} \tag{6}$$

Here $f_\oplus$ is the Earth's flattening, $R_\oplus$ is Earth's equatorial radius, φ — geodetic latitude, $\Omega_\oplus$ — Earth's angular velocity, GM — Earth's gravitational constants, and g_{eq}, k are constants.

The equation 4 is solved with two iterations. At each iteration we take approximation to $g(h)$ from the previous iteration. Before the first iteration we tabulate $P(h)$ and $g(h)$ using the ISO Standard atmosphere (*ISO*, 1975) as a reference within range [-1, 90] km and fit parameters of a linear regression between $\ln P$ and $g/g_{\rm geoid}$. This gives us an initial approximation of $g(P)$:

$$\begin{aligned} g_{\rm geoid}(\varphi) &= g_{eq}\frac{1+k\sin^2\varphi}{\sqrt{1-(2f_\oplus - f_\oplus^2)\sin^2\varphi}}\ . \\ g(\varphi,P) &= g_{\rm geoid}(\varphi)\cdot(g_0+g_1\ln(P)) \end{aligned} \tag{7}$$

Here $g_{\rm geoid}$ — is the gravity acceleration at the geoid, g_0 and g_1 are parameters of the linear regression of the dependence of geometric height above the geoid on atmospheric pressure. g_0 is the ratio of the gravity acceleration at pressure level 1 Pa to the gravity acceleration at the geoid at the same latitude. g_1 is the mean dependence of the Earth's gravity acceleration on logarithm of the total air pressure. Numerical values of parameters g_0 and g_1 can be found in Table 7.

Then, this expression for dependence of $g(P)$ is used in the first iteration. Solution of equation 4 is trivial:

$$h(P_k) = h_s + \int_{P_s}^{P_k} \frac{R\,T(P)}{g(P)\left(M_d\,(P-P_w)\ +\ M_w\,P_w\right)Z^{-1}(P,P_w,T)}\,dP \tag{8}$$

where P_s is the total pressure at the surface and P_k is the total pressure at layer k.

The GEOS-FPIT numerical weather model does not provide pressure at middle layer. It provides pressure thickness $\Delta P(k) = P_b(k) - P_t(k)$ where $P_l(k)$ is the atmospheric pressure at the

[1] 1 x 1-Minute Geoid Undulation Grid available from
http://earth-info.nga.mil/GandG/wgs84/gravitymod/egm2008/egm08_wgs84.html

bottom edge of the kth layer and $P_l(k)$ is the atmospheric at the top of the kth layer. Air temperature and specific humidity are provided for the middle of the layer. The upper edge of the highest pressure level is fixed: 1Pa. Atmospheric pressure at a given layer can by produced by summing the pressure at the layers from the top (See Figure 3.1.2).

The right hand-side of expression under integral 8 is not known at the surface, which corresponds to the bottom edge of the lowest layer. it can be computed by linear extrapolation:

$$\begin{aligned} \frac{dh}{dP}(s) &= \frac{dh}{dP}(l) + \Delta P(l)/2 \, \frac{\frac{dh}{dP}(l-1) - \frac{dh}{dP}(l)}{P(l-1) - P(l)} \\ P(s) &= P(l) + \Delta P(l)/2 \end{aligned} \tag{9}$$

where l is the index of the lowest layer. Remember, for GEOS-FPIT the lowest layer has the greatest index. We extend $\frac{dh}{dP}$ and P to the surface: $P(0)$ and $\frac{dh}{dP}(0)$ and use these extended functions for further analysis.

In order to compute integral 8, we expand the expression under the integral into B-spline basis of the 3rd degree: $f(P) = \sum_{1-m}^{n-1} f_i \, B_i^m(P)$, where $B_i^m(P)$ is the B-spline function of the m-th degree with the ith pivotal element with $n+1$ nodes that are indexed starting from 0. Then

$$h(P_k) = h_s + \sum_{i=1-m}^{k-1} f_i I_i^m(P), \tag{10}$$

where $I_i^m(x) = \int_\infty^x B_i^m(x)\,dx$ — a function that is expressed via B-splines of $m+1$ degree. If to introduce function $L_{i,k}^m(x) = I_i^m(x_k) - I_i^m(x_{k-1}) = \int_{x_{k-1}}^{x_k} B_i^m(x)\,dx$ and keeping in mind that $B_k^m(x)$ is zero outside interval $[x_{k-m}, x_{k+1}]$, expression 10 can be re-written as

$$h(P_0) = h_0 \tag{11}$$

$$h(P_k) = h(P_{k-1}) + \sum_{i=k-m-1}^{k-1} f_i \, L_{i,k}^m(P). \tag{12}$$

where index 0 corresponds to the surface and index $k > 0$ corresponds to the middle of a layer.

Then at each element of the grid we compute $g(h(P_k))$ using expression 5 and make the second iteration. As a result, we obtain height above the reference ellipsoid for each layer of total pressure, air temperature and specific humidity q. Finally, we transform specific humidity into partial pressure of water vapor:

$$P_w = \frac{q}{\frac{M_w}{M_d} + \left(1 - \frac{M_w}{M_d}\right) q} \, P. \tag{13}$$

3.1.3. Regridding the state of the atmosphere to the regular grid

The grid of parameters P, P_w, and T is irregular: the grid step along level axis depends on latitude, longitude and time. Interpolation at a multidimensional irregular grid is impractical. The

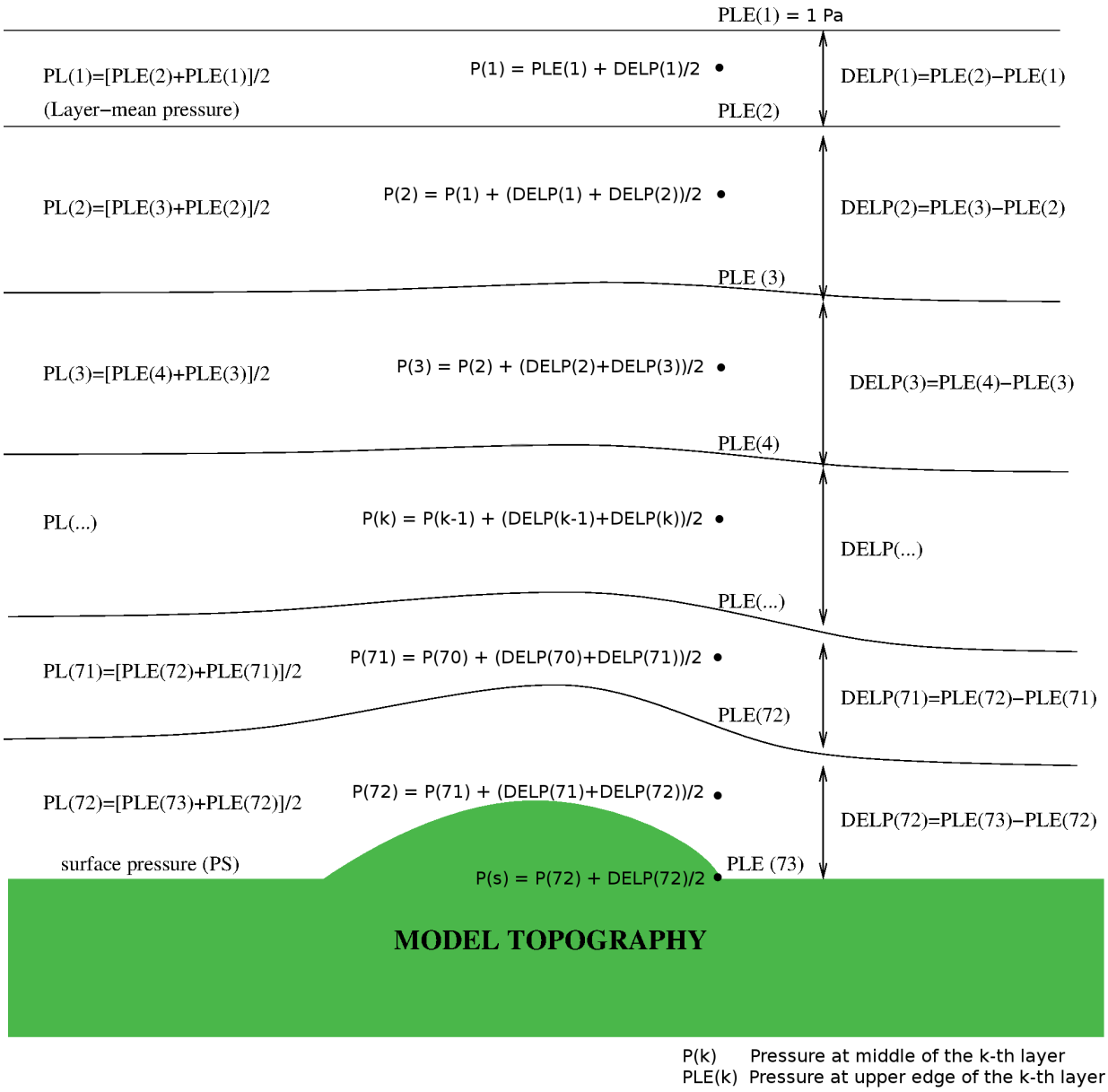

Figure 1. Structure of vertical GEOS-FPIT vertical grid.

next step of the data processing chain is to re-cast the atmosphere state parameters to a new regular grid that has fixed height of levels. It should be noted that for efficient interpolation the new grid does not need to be uniform, i.e. to have a constant step. Since at the first approximation, atmospheric pressure varies linearly with P, a grid with a uniform step over $\log(P)$ will introduce less artifacts than a grid with a uniform step over P.

The following grid is suggested:

$$h_i = \exp \frac{i - \mu_3}{\mu_1} - \mu_2 \tag{14}$$

where $\mu_1 = 20.25319, \mu_2 = 1200.0, \mu_3 = -169.30782$ for i $\in [-62, 62]$. The grid runs from -1000 to 90,000 meters above the reference ellipsoid. The new grid is wider and has 55% more nodes. Such a grid was selected to have density of nodes no less than the initial grid in order to avoid loss of accuracy during regridding.

The nodes of the uniform grid fall into three regions: 1) the region below the assumed surface, 2) the region within the range of height of the numerical weather model, 3) and the region above the modeled top of the atmosphere. Regions 1 and 3 require extrapolation and region 2 requires interpolation.

We expand total atmospheric pressure, partial pressure of water vapor, and air temperature over the B-spline basis against height variable. For points in region 2 we use the expansion coefficients for computing these quantities at nodes of the new uniform grid.

We assume the atmosphere above the highest grid level is isothermal and dry. The extrapolation is done this way:

$$\begin{aligned} P(h_i) &= P(h_u)\ \exp\{\ -g_u\, M_d\, \frac{h_i - h_u}{R\, T_u}\ \} \\ T(h_i) &= T_u \\ P_w(h_i) &= 0 \end{aligned} \tag{15}$$

where h_u, T_u, and g_u are height, air temperature and gravity acceleration at the upper pressure level of the output of the numerical weather model.

We assume the atmosphere obey the adiabatic law with a constant temperature derivatives $\frac{\partial T}{\partial h}$ (so-called lapse rate with the opposite sign) below the first pressure level. The lapse rate near the surface may differ significantly from the lapse rate in the middle of the troposphere. For extrapolation of the state of the atmosphere we prefer to use the representative lapse rate for the middle of the troposphere. We compute derivative $\frac{\partial T}{\partial h}$ with least squares over the range of heights from 1000 meters above the first level and 9000 meters. Then we compute air temperature and total pressure at the levels of the new grid as

$$\begin{aligned} T(h_i) &= T(h_1) + \frac{\partial T}{\partial h}\,(h_i - h_s) \\ P_w(h_i) &= P_w(h_1)\,(T(h_i)/T(h_1))^{-g(h_1)\,M_w/(R\,\frac{\partial T}{\partial h})} \\ P(h_i) &= P_w(h_i) + (P - P_w)(h_1)\,(T(h_i)/T(h_1))^{-g(h_1)\,M_d/(R\,\frac{\partial T}{\partial h})}, \end{aligned} \tag{16}$$

where h_1 is the height of the first level of the output of the numerical weather model.

Of course, accuracy of extrapolation beneath the surface is lower than the accuracy of the atmosphere state parameters within the range of heights of numerical weather models. We compute

atmosphere state parameters below the nominal surface for two reasons. First, we would like to keep continuity of the atmosphere over the height range [-1, 90] km in order to avoid large errors in interpolation near the surface. Second, due to the insufficient resolution, the actual Earth surface in valleys of mountainous regions may fall below the nominal surface if nodes of the grid happened to be located on a mountain ridge.

3.1.4. Computation of air refractivity

For computation of refractivity with the accuracy we need, $5 \cdot 10^{-4}$, we can consider all atmospheric gases, except water vapor, maintaining a fixed mixing ratio. The atmosphere can can be considered consisting of two components: dry and wet.

Detailed overview of the state-of-the-art of modeling refractivity is given in Rüeger (2002). The working group of the International Union of Geodesy and Geophysics on Fundamental Constants, issued a recommendation in 1991 that stated that "the group refractive index in air for electronic distance meter measurements to better than one part per million with visible and near infrared waves in the atmosphere be computed using the procedure published by Ciddor (1996) and Ciddor & Hill (1999).

The refractivity of moist air is computed according to Ciddor (1996)

$$r(P, P_w, T, k) = \frac{\rho_d(P, P_w, T)}{\rho_{d,r}} r_{d,r}(k) + \frac{\rho_w(P, P_w, T)}{\rho_{w,r}} r_{w,r}(k) \tag{17}$$

where $\rho_d(P, P_w, T)$ is density of dry component of the atmosphere as a function of total pressure P, P_w is partial pressure of water vapor and air temperature, $\rho_w(P, P_w, T)$ is density of wet component, and $\rho_{d,r}$, $\rho_{w,r}$ are densities of dry air and water vapor at certain reference conditions. $r_{d,r}$ and $r_{w,r}$ are refractivity of dry air and water vapor at these standard conditions as a function of wave number which is reciprocal to wavelength $k = 1/\lambda$. Here, we define the refractivity as $\frac{c-v}{v}$, where c is the speed of light in vacuum and v is the speed of light in the medium.

Ciddor (1996) suggests the following equations for group refractivity $n_{d,r}$ and $n_{w,r}$ for reference meteorological conditions:

$$\begin{aligned} r_{d,r}(k) &= d_1 \frac{d_0 + k^{2\,2}}{d_0 - k^2} + d_3 \frac{d_2 + k^{2\,2}}{d_2 - k^2} \\ r_{w,r}(k) &= C\,(w_0 + 3w_1\,k^2 + 5w_2\,k^4 + 7w_3\,k^6) \end{aligned} \tag{18}$$

Reference meteorological conditions are $P = 101325$ Pa, $P_w = 0$, $T = 288.15$K for $r_{d,r}(k)$ and $P = 1333$ Pa, $P_w = 1333$ Pa, $T = 293.15$K for $r_{w,r}(k)$.

Using the formula for the density of moist air 2, the expression for air refractivity can be regrouped to

$$r = \frac{S_t\,P \;+\; S_w\,P_w}{T}\,Z^{-1} \tag{19}$$

where

$$S_t = \frac{M_d}{R}\frac{r_{d,r}}{\rho_{d,r}}; \qquad S_w = \frac{M_w}{R}\frac{r_{w,r}}{\rho_{w,r}} - S_t \tag{20}$$

Thus, according to the rigorous algorithm for computing air refractivity in optical and near infrared range, path delay for ICESat-1 and ICESat-2 is computed using the following expression:

$$n = \left(S_t(\lambda)\,\frac{P}{T} + S_w(\lambda)\frac{P_w}{T} \right) Z^{-1}(P, P_w, T) \tag{21}$$

where P is the total pressure, P_w is the partial pressure of water vapor, T is air temperature, S_t and S_w are parameters that depend only on wavelength and Z is the air compressibility.

Refractivity is computed for the each element of the 4D uniform grid using total atmospheric pressure, partial pressure of water vapor, and air temperature.

3.1.5. Expanding the refractivity field into basic functions

Equations of wave propagation assume light propagating in a continuous medium. To satisfy this formalism, we have to expand the refractivity field into some basic functions. The natural choice is B-spline basic functions (de Boor, 2001), because their property of limited support (i.e. B-spline is zero outside a small bounded area) allows to built an extremely efficient algorithm for expansion into multi-dimensional basis.

For processing an ICESat dataset, typically one day, we extract 3D fields for all epochs within the time range of the ICESat data and two epochs before the dataset start date and two epochs after the dataset stop date. We also augment the dataset with three steps over longitude: $\lambda_{n+1}, \lambda_{n+3}, \lambda_{n+2}$ being equal to $\lambda_1, \lambda_2, \lambda_3$. This is done in order to avoid degradation of quality of interpolation at the edges of the grid: near start/stop time and near the zero meridian.

Then the set of *discrete* refractivity values at the 4D grid are used for computation of the coefficients of expansion the refractivity field into the four-dimensional tensor product of B-splines:

$$r(h, \lambda, \varphi_{gd}, t) = \sum_{i=1-m}^{i=d_1-1} \sum_{j=1-m}^{j=d_2-1} \sum_{k=1-m}^{k=d_3-1} \sum_{l=1-m}^{l=d_4-1} f_{ijkl}\, B_i^m(h)\, B_j^m(\lambda)\, B_k^m(\varphi_{gd})\, B_l^m(t), \tag{22}$$

where $B_s^m(x)$ is the basis spline function of variable x of degree m with the pivot node s, and d_1, d_2, d_3, d_4 are dimensions. Since our grid is uniform, we can proceed with expansion over each axis independently.

A B-spline is defined at a sequence of grid points, traditionally called knots. A B-spline of degree m is defined at a non-decreasing (i.e. $x_{i+1} \geq x_i$) sequence of $m+1$ knots, called support. The first knot of the sequence is called a pivotal knot. A B-spline is zero outside the area of its support, i.e. $\forall x \notin [x_i, x_{i+m+1}] B^m(x) = 0$.

In general, the sequence of knots for the B-spline basis may not coincide with the grid points. If the number of knots of the basis is less than the number of grid points, B-spline expansion will smooth the data. We can force the B-spline expansion to pass through the data at grid points. To do that in a 1D grid $x_1, x_2, \ldots x_n$ we consider the so-called extended sequence of knots: $x_{1-m}, x_{2-m}, \quad \ldots \quad x_1, x_2, \quad \ldots \quad x_{n-1}, x_n, x_{n+1}, \quad \ldots \quad x_{n+m}$, where $x_{1-m} = x_{2-m} = \ldots x_1$ and $x_{n+m} = x_{n+m-1} \ldots = x_n$, or using another language, the first and the last knot has multiplicity m. Since $B^m(x_n) = 0$, we have $n + m - 1$ B-spline at a sequence of n knots. In this work we use B-splines of the 3rd degree. Therefore, in order to force the expansion over the B-spline basis to

pass through each 1D grid point, we need to evaluate coefficients of $n+2$ B-splines. We need to augment n equations

$$F(x_k) = \sum_{k=1-m}^{k=n-1} f_k B^m(x_k) \tag{23}$$

with two more equations. There is more than one way to do it. Here we add equations requiring that the first derivative at the end of the interval be equal to first differences:

$$\begin{aligned} \frac{F(x_2)-F(x_1)}{x_2-x_1} &= \sum_{k=1-m}^{k=n-1} f_k B'^m(x_k) \\ \frac{F(x_k)-F(x_{k-1})}{x_k-x_{k-1}} &= \sum_{k=1-m}^{k=n-1} f_k B'^m(x_k). \end{aligned} \tag{24}$$

Recurrent algorithms for computing $B^m(x_k)$ and $B'^m(x_k)$ are given in the Appendix A.1.

Equations 23–24 form a system of $n+2$ linear algebraic equations with $n+2$ unknowns. Considering B-spline is non-zero only at m+1 knots, the the first and the last equations look like that:

$$\begin{aligned} f_{-2}B'^2_{-2}(x_1) + f_{-1}B'^2_{-1}(x_1) &= F'(x_1) \\ f_{-2}B^2_{-2}(x_1) + f_{-1}B^2_{-1}(x_1) + f_0 B^2_0(x_1) &= F(x_1) \\ f_{-1}B^2_{-1}(x_2) + f_0 B^2_0(x_2) + f_1 B^2_1(x_2) &= F(x_2) \\ \cdots & \\ f_{n-3}\,B^2_{n-3}(x_n) + f_{n-2}\,B^2_{n-2}(x_n) + f_{n-1}\,B^2_{n-1}(x_n) &= F(x_n) \\ f_{n-2}\,B'^2_{n-2}(x_n) + f_{n-1}\,B'^2_{n-1}(x_n) &= F'(x_n) \end{aligned} \tag{25}$$

That means that the $(n+2)\times(n+2)$ matrix in 25 is band-diagonal:

$$\begin{pmatrix} x & x & & & & & & & & \\ x & x & x & & & & & & & \\ & x & x & x & & & & & & \\ & & x & x & x & & & & & \\ & & & x & x & x & & & & \\ & & & & x & x & x & & & \\ & & & & & x & x & x & & \\ & & & & & & x & x & x & \\ & & & & & & & x & x & x \\ & & & & & & & & x & x \end{pmatrix} \tag{26}$$

Linear algebraic systems like 26 are solved very efficiently with algorithms of O(n) complexity.

Expansion over 2D tensor products of B-splines is first done d_2 times over dimension d_1 and then d_1 times over dimension d_2 where d_1 and d_2 are dimensions across the 1st and the 2nd axes. Since our grid is regular, during each run over dimension 1 and 2 we solve k systems of linear algebraic equations with the same left-hand side and different right-hand sides. Left-band side is determined only by the sequence of grid points. BLAS library (Basic Linear Algebra Subprograms)

has efficient and highly optimized routines for solving band-diagonal systems of linear algebraic equations with multiple right-hand sides.

Expansion over 4D tensor products of B-splines is generalized straightforwardly from the 2D case: first the coefficients of B-spline expansion across the first dimension are computed with $d_2 \times d_3 \times d_4$ right-hand sides. Elements of the grid points are replaced with the coefficients of B-splines after each cycle. Then the procedure is repeated across the second dimension with $d_1 \times d_3 \times d_4$ right-hand sides, across the third dimension, and across the fourth dimension.

Thus, the problem of evaluation of the 4D fields of the coefficients of expansion f_{ijkl} is reduced to solving systems of banded linear algebraic equations over 1st, 2nd, 3rd and 4th dimension with multiple right-hand sides.

3.2. Computation of zenith path delay using the refractivity field

Computation of path delay in zenith direction is performed by numerical integration of the refractivity along the vertical direction from the surface height defined by ICESat measurement and the highest layer. Since variables h, λ, φ, and t are independent, integral over variable h is

$$d = \sum_{j=1-m}^{j=d_2-1} \sum_{k=1-m}^{k=d_3-1} \sum_{l=1-m}^{l=d_4-1} B_j^m(\lambda)\, B_k^m(\varphi_{gd})\, B_l^m(t) \sum_{i=1-m}^{i=d_1-1} \int_{h_s}^{h_t} R_{ijkl}\, B_i^m(h)\, dh. \tag{27}$$

Here we introduced function

$$J_k^m(x) = \int_x^{+\infty} B_k^m(x), \tag{28}$$

which is computed similar to $B_k^m(x)$ through a recurrent relationship. Notice that $B_k^m(x)$ is non-zero only at $m+1$ knots in the vicinity of the grid point closest to the x, but not exceeding x, Then we can write integral 20 elegantly:

$$d(h, \lambda, \varphi_{gd}, t) = \sum_{i=i_p-m}^{i=d_1-1} \sum_{j=j_p-m}^{j=j_p} \sum_{k=k_p-m}^{k=k_p} \sum_{l=l_p-m}^{l=l_p} R_{ijkl}\, J_i^m(h)\, B_j^m(\lambda)\, B_k^m(\varphi_{gd})\, B_l^m(t), \tag{29}$$

where i_p is maximum i that $h_i < h$, j_p is maximum j that $\lambda_j < \lambda$, k_p is maximum k that $\varphi_i < \varphi$, and l_p is maximum l that $t_l < t$.

During processing ICESat data, we do not know the surface height precisely. We see from the definition of path delay d that its derivative over height is just air refractivity. It is computed as

$$\frac{\partial d(h, \lambda, \varphi_{gd}, t)}{\partial h} = n = \sum_{i=i_p-m}^{i=i_p} \sum_{j=j_p-m}^{j=j_p} \sum_{k=k_p-m}^{k=k_p} \sum_{l=l_p-m}^{l=l_p} R_{ijkl}\, B_i^m(h)\, B_j^m(\lambda)\, B_k^m(\varphi_{gd})\, B_l^m(t). \tag{30}$$

That derivative is used when we need to correct path delay for update of the surface height with respect to its a priori value. If the height correction is too large, the contribution of the second derivative may appear substantial. In order to evaluate the contribution of the second derivative of delay with height, which is equal to the first derivative of refractivity with height, we compute

it using the ISO standard atmosphere at the sea level: $\frac{\partial^2 d}{\partial h^2} = \text{-}2.7 \cdot 10^{-8}$ m^{-1}. The contribution will reach 1 mm, when the height correction exceeds 270 m.

3.3. Computation of slant path delay using the refractivity field

ICESat shoots at the angle z which is typically less than 1°, and never exceeds 5°. Assuming the atmosphere is flat, the path delay at zenith angle z depends on path delay in zenith direction as

$$d(z) = d_z / \cos(z). \tag{31}$$

Shooting at zenith angle that differs from zero causes another effect: the footprint location is shifted due to bending the light trajectory in the atmosphere. Rigorous computation of slant path delay that accounts for sphericity of the Earth's atmosphere and bending is not a trivial task. It is described in detail in the Appendix A.1. In general case it is reduced to solving a system of non-linear differential equations of the fourth order. Investigation of results of rigorous computation of slant path delay showed

- Errors of approximation of expression 31 does not exceed 1 mm at $|z| < 5°$.
- The footprint offset are less than 30 cm at $|z| < 5°$ at even extreme weather conditions.

Therefore, in order to meet the goal of 1 mm precision of atmosphere path delay computation for ICESat data processing, it is sufficient to scale zenith path delay in expression 29 by a factor $1/\cos(z)$ for computation of slant path delay at zenith angle z. The footprint shift due to bending in the atmosphere is negligible.

4. Algorithm implementation

Convention:

1. Fortran array notation is used throughout this documents: 1) array indexing starts from 1, i.e. [1:n] unless it explicitly stated otherwise; 2) the first dimension in multidimensional arrays loops first. For example, array A[2,3] is unraveled as A[1,1], A[2,1], A[1,2], A[2,2], A[1,3], A[2,3].
2. 2D arrays with spatial data have the following order of dimensions: longitude, latitude
3. 3D arrays with spatial data have the following order of dimensions: height, longitude, latitude
4. 4D arrays with spatial data have the following order of dimensions: height, longitude, latitude, time.
5. longitude grid runs from 0 to $[2\pi - 2\pi/n_\lambda]$, where n_λ is the dimension of longitude axis.
6. longitude extended grid has $n_\lambda + 4$ nodes with indices in a range $[-2, n_\lambda + 1]$. Index 1 corresponds to longitude 0. $\lambda[n_\lambda + 1] = \lambda[0]$.
7. latitude grid runs from $-\pi/2$ to $+\pi/2$.
8. Height runs from down to top.

5. Algorithm description

All computations belong to three categories:

1. Computation of time-invariant quantities. This is done only once. Results of these computations are stored and used later.
2. Asynchronous processing a new dataset. This procedure runs asynchronously with respect to processing ICESat data. As soon as a new output of numerical weather model GEOS-FPIT became available, it is downloaded and transformed to another dataset. That secondary dataset is written down.
3. Computation of path delay. Processing of ICESat data is done by chunks of data. First, the start and stop date of the chunk of data is determined. Then, a set of files with a secondary dataset is read, and an expansion of refractivity over 4D B-spline basis is computed. Using these expansion coefficients, the path delay is computed in a loop for a given instant of time and given geolocation.

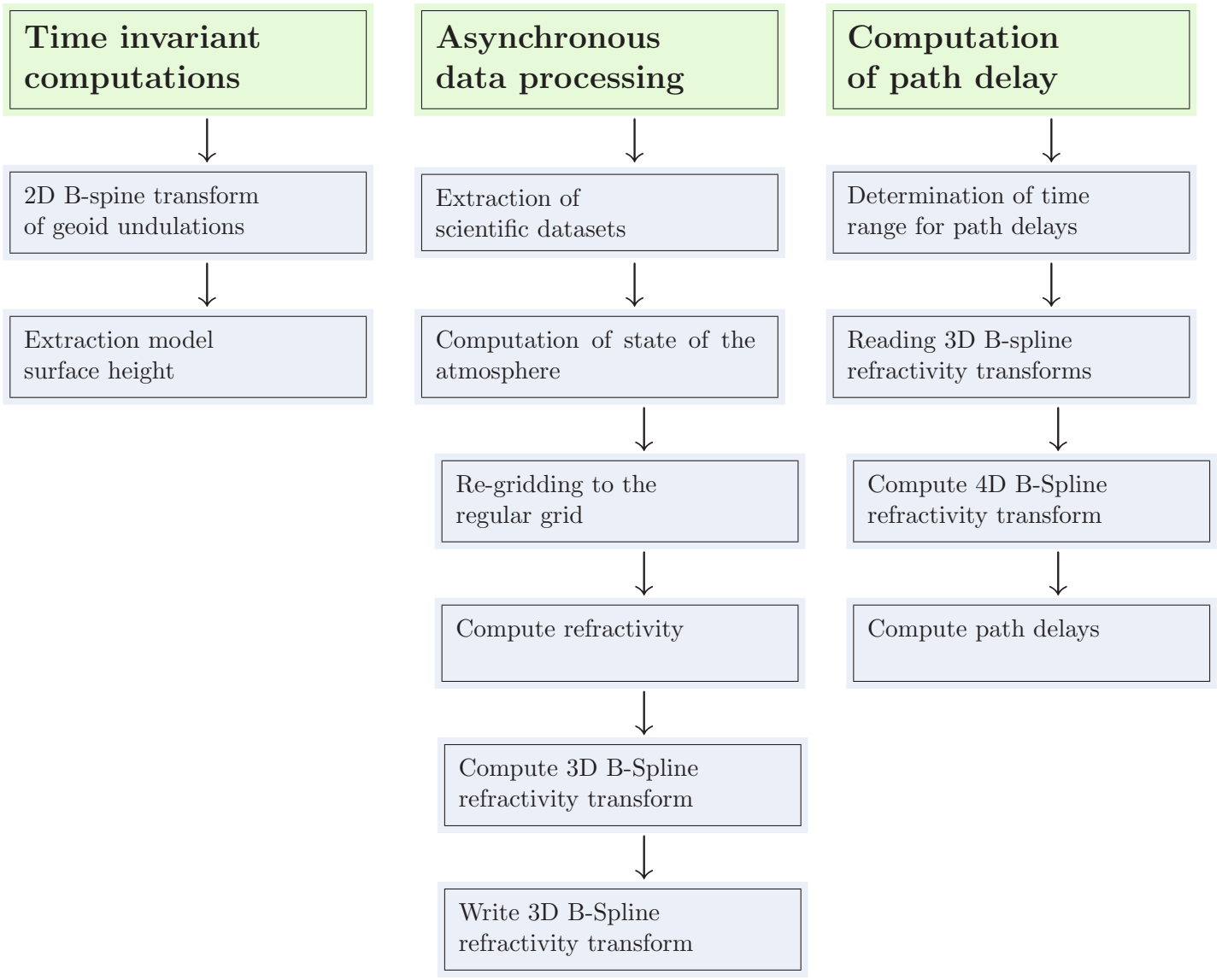

Figure 1. Chart of processing flow

5.1. Computation of time-invariant quantities needed for deriving the refractivity field from the output numerical weather models

Although numerical weather models contain product called "surface pressure" we do not trust it because surface definition used in a numerical weather model is not precise. Its accuracy is sufficient for modeling weather, but not sufficient for modeling path delay with target accuracy 1 mm. Two time-invariant datasets related to the surface are required for computation of path delay:

1. Surface height used in the numerical weather model. Data collection of numerical weather model GEOS-FPIT has product DFPITC0NXASM accessible at

   http://aurapar2u.ecs.nasa.gov/goldsfs1/data/GEOS5/DFPITC0NXASM.5.9.1/

2012/365/.hidden/DAS.fpit.asm.const_2d_asm_Nx.GEOS591.00000000_0000.V01.nc4

This product contains dataset PHIS "surface geopotential". This name is misleading. According to M. Bosilovich, the GEOS-FPIT model surface height is H_g = PHIS/g_n, where $g_n = 9.8m/s^2$ is the nominal gravity acceleration used in GEOS-FPIT. Therefore, PHIS is the increment of geopotential at the nominal surface with respect to the geopotential at the geoid, under assumption that local gravity acceleration is 9.8 m/s^2. In fact, dataset PHIS has been generated from some digital elevation model scaled by a factor 9.8. Dimension of H_g : $n_\lambda = 576$ over longitude and $n_\varphi = 361$ over latitude. Latitude is in range $[-\pi/2, \pi/2]$, longitude of PHIS product is in range $[-\pi, \pi - 2\pi/n_\lambda]$. Therefore, PHIS grid is to be transformed to $[0, 2\pi - 2\pi/n_\lambda]$ by cycling rotation:

for $i = 1, 2, \ldots, n_\lambda$ parallel do
  if $i =< n_\lambda/2$then
    PHIS$[i, j] := PHIS[i + n_\lambda/2, j]$
  else
    PHIS$[i, j] := PHIS[i - n_\lambda/2, j]$

2. Geoid height (or geoid undulation) above the reference ellipsoid in WGS84 system. The geoid height with respect to the geopotential EGM2008 at $1' \times 1'$ grid can be found at

   http://earth-info.nga.mil/GandG/wgs84/gravitymod/egm2008/

   Und_min1x1_egm2008_isw=82_WGS84_TideFree.gz

   This is a binary file in Low Endian with (180*60+1) records. Each record has a 4-byte long prefix, 360*60 elements of data as single precision real numbers, and a 4-byte long trailer. Grid starts at $-\pi/2$ along latitude and it starts at 0 along longitude.

   The data from this file are put in the extended grid [-2:21601][-2:10801] and replaced with their 2D B-spline transform.

These 2D arrays are generated only once and stored.

5.2. Asynchronous data processing

5.2.1. Extraction of datasets

URLs with the output of GEOS-FPIT model at native grid have the following format
http://aurapar2u.ecs.nasa.gov/goldsfs1/data/GEOS5/DFPITT3NVASM.5.9.1/ Y@@@/D@@/
.hidden/DAS.fpit.asm.inst3_3d_asm_Nv.GEOS591.d@@@@@@@@_t@@@.V01.nc4
where

- Y@@@ is a four-digit year;
- D@@ is the three-digit day of year with leading zeroes,
- d@@@@@@@@ is date in yyyymmdd format. The first 4 characters are the year, following two character are the month number with leading zeros, and last three characters are the day of the month with leading zeroes if necessary;
- t@@@ is time in hhmm format. The first two characters are the hour, the last two characters are the minute. Since the output of GEOS-FPIT is generated with a step of 3 hours, the following values are valid: 0000, 0300, 0600, 0900, 1200, 1500, 1800, 2100.

Example: http://aurapar2u.ecs.nasa.gov/goldsfs1/data/GEOS5/DFPITI3NVASM.5.9.1/

2014/076/.hidden/DAS.fpit.asm.inst3_3d_asm_Nv.GEOS591.20140317_1500.V01.nc4
This file contains the model output at the native grid at 2014.03.17T15:00:00 UTC.

The data file is in NetCDF-4 format. Three scientific datasets are to be extracted: DELP — pressure thickness, T — air temperature, and QV — specific humidity. Although the datasets are declared as 4D arrays, the last dimension (time) is one. **NB:** the levels are counted from the top of the atmosphere downwards in all GEOS models, including GEOS-FPIT.

5.2.2. Computation of the state of the atmosphere on the original grid

A general procedure for computation of the state of the atmosphere is an execution of a nested loop that takes for a given latitude and a given longitude 1) the vertical column of pressure thickness $\Delta p[k]$, 2) the vertical column of air temperature $T[k]$, 3) the vertical column of specific humidity $q[k]$, and 4) the surface potential increment PHIS. The output is the column at the uniform grid with a) atmospheric pressure, b) partial water vapor pressure; c) air temperature. **NB:** these three output columns are at the fixed vertical uniform grid that does not depend on longitude and latitude. The vertical grid of the original input grid is not defined explicitly and it depends on latitude and longitude.

for $i = 1, 2, \ldots, n_\varphi - 1, n_\varphi$ do
  $\varphi[i] = -\pi/2 + (i-1) * \pi/(n_\varphi - 1)$
  for $j = 1, 2, \ldots, n_\lambda - 1, n_\lambda, n_\lambda + 1$ do
    if $j < n_\lambda/2$ then
      $\lambda[j] = (j + n_\lambda/2) * 2\pi/n_\lambda$
    else if $j < n_\lambda + 1$ then
      $\lambda[j] = (j - n_\lambda/2) * 2\pi/n_\lambda$
    else if $j = n_\lambda + 1$ then
      $\lambda[j] = 0$
    Process the vertical profile at $(\lambda[j], \varphi[i])$

Below is the detailed description of the procedure "Process the vertical profile" at $(\lambda[j], \varphi[i])$ referred above.

Input data:

1) PHIS[i,j]	the surface geopotential increment used by the numerical weather model in m^2/s^2;
2) Δp[1:n,i,j]	array of pressure thickness in Pa;
3) t[1:n,i,j]	array of absolute air temperature in K;
4) q[1:n,i,j]	array of specific humidity (dimensionless).

Output data:

1) h[0:n,i,j]	array of mid-layers geometric heights above geoid in meters; h[0] is the geometric height at the nominal surface.
2) p[0:n,i,j]	array of atmospheric pressure at mid-layer in Pa; p[0] is the atmospheric pressure at the nominal surface.
3) p_w[1:n,i,j]	array of partial pressure of water vapor at mid-layer in Pa;
4) t[1:n,i,j]	array of air temperature in K (unchanged).

1. Compute the local gravity acceleration at the reference ellipsoid with geodetic latitude $\varphi[i]$:

$$g_{\text{geoid}} = g_{eq} * (1 + k * \sin^2(\varphi))/\text{sqrt}(1 - (2f_e - f_e^2) * \sin^2(\varphi))$$

where g_{eq} is the equatorial gravity acceleration and f_e is Earth's flattening.

2. Compute atmospheric pressure at mid-layer. The pressure at the upper edge of the topmost level is fixed: 1 Pa. Then combining thickness of two adjacent layers ΔP, the compute atmospheric pressure at the middle of a layer is computed as

$$\begin{aligned}
&p[n] = 1.0 + \Delta p[1]/2 \\
&tt = t \\
&qq = q \\
&\textsf{for } k = n-1, n-2, \ldots, 2, 1 \textsf{ do} \\
&\quad p[k] = p[k+1] + (\Delta p[n-k] + \Delta p[n+1-k])/2.0 \\
&\quad q[k] = qq[n+1-k] \\
&\quad t[k] = tt[n+1-k] \\
&p[0] = p[1] + \Delta p[n]/2.0
\end{aligned}$$

This procedure also reverses layer indexing for total atmospheric pressure p, air temperature t, and specific humidity q. From now on, index 1 corresponds to the middle of the lowest layer, index n corresponds to the middle of the topmost layer, index 0 corresponds to the nominal surface. Similar convention is used for arrays that will be introduced below: the array for partial pressure of water vapor p_w, the array of partial pressure for dry air p_d, the array of compressibility of moist air z_m, the array of gravity acceleration g, the array of geometric height h, and the array of right hand-side of the hypsometric differential equation y. Array of layer pressure thickness Δp remains unchanged.

3. Convert specific humidity to partial pressure of water vapor p_w and partial pressure of dry air p_d. Using these quantities, as well as absolute air temperature t, compute a) compressibility of moist air z_m, b) the gravity acceleration, and c) the right hand-side of the hypsometric differential equation $y = \dfrac{dh}{dp}$:

$$\begin{aligned}
&\textsf{for } i = 1, 2, \ldots, n-1, n \textsf{ do} \\
&\quad p_w[i] = q[i]/(M_w/M_d + (1.0 - M_w/M_d) * q[i]) * p[i] \\
&\quad p_d[i] = p[i] - p_w[i] \\
&\quad z_m[i] = 1.0/(\quad 1.0 \\
&\qquad\qquad -p[i]/t[i] * (a_0 + a_1 * (t[i] - t_0) + a_2 * (t[i] - t_0)^2) \\
&\qquad\qquad +p_w[i]/t[i] * (b_0 + b_1 * (t[i] - t_0)) \\
&\qquad\qquad +p_w^2[i]/(p[i] * t[i]) * (c_0 + c_1 * (t[i] - t_0)) \\
&\qquad\qquad +p^2[i]/t^2[i] * e_0 \\
&\qquad\qquad +p_w^2[i]/t^2[i] * f_0 \\
&\qquad\quad) \\
&\quad g[i] = g_{\text{geoid}} * (g_0 + g_1 * \ln(p[i])) \\
&\quad y[i] = R * t[i]/(g[i] * z_m[i] * (M_d * p_d[i] + M_w * p_w[i])) \\
&y[0] = y[1] - \Delta p[n]/2.0 * (y[2] - y[1])/(p[2] - p[1])
\end{aligned}$$

where M_d and M_w are constants: molar mass of dry air and water vapor respectively; t_0 is water freezing temperature in K. $y[0]$ at the nominal surface is computed by extrapolation.

4. Perform 1D B-spline transform of degree m (m=3) of array y:

$$Y(p) = \mathcal{B}^m(y(p))$$

5. Compute the coarse estimate of mid-layer ortho-heights:

$$h_k = PHIS[i,j]/g_n + \int_{p[0]}^{p[k]} Y(p)\,dp$$

Using results in subsection B.1.5, this can be done using the following algorithm:

$h[0] = PHIS[i,j]/g_n$
for $k = 1, 2, \ldots, n-1, n$ do
    $h[k] = h[k-1]$
    for $l = k-m-1, k-m \ldots, k-1$ do
        $h[k] = h[k] + Y[l] * K_\ell^m(p[l])$

where $K_\ell^m(x)$ is a function defined in expression 64: the integral of B-spline of degree m with the pivotal index ℓ in a range [x[l-1], x[l]].

6. Compute an improved approximation to function y(p) using a more precise model of changing the gravity acceleration with atmospheric pressure.

for $i = 1, 2, \ldots, n-1, n$ do
    $g[i] = g_{\text{geoid}} * (1 - 2/R_e * h[i] + 3/R_e^2 * h^2[i])$
    $y[i] = R * t[i]/(g[i] * z_m[i] * (M_d * (p[i] - p_w[i]) + M_w * p_w[i]))$

Here R_e is the Earth's equatorial radius and R is the universal gas constant.

7. Repeat step 5 one time.

5.2.3. Re-gridding the state of the atmosphere to the uniform grid

for $i = 1, 2, \ldots, n_\varphi - 1, n_\varphi$ do
    for $j = 1, 2, \ldots, n_\lambda - 1, n_\lambda, n_\lambda + 1$ do
        Re-grid the columns p[1:n], p_w[1:n], t[1:n] as a function of h[1:n], to
            $\bar{p}$[1:$\bar{n}$], $\bar{p}_w$[1:$\bar{n}$], $\bar{t}$[1:$\bar{n}$], as a function of $\bar{h}$[1:$\bar{n}$]
            $\bar{h}$[1:$\bar{n}$] does not depend on i, j

Regridding the column is explained in detail below.
Input data:

1) n=72 input dimension over layers
2) $h[1{:}n]$ input mid-layer height above geoid
3) $p[1{:}n]$ input mid-layer air pressure
4) $p_w[1{:}n]$ input mid-layer water vapor pressure
5) $t[1{:}n]$ input mid-layer air temperature

Output data:

1) $\bar{n}$=125 output dimension over layers
2) $\bar{h}[1{:}\bar{n}]$ constant array of output grid heights above geoid
3) $\bar{p}[1{:}\bar{n}]$ output array of air pressure
4) $\bar{p}_w[1{:}\bar{n}]$ output array of water vapor pressure
5) $\bar{t}[1{:}\bar{n}]$ output array of air temperature

1. Compute B-spline transform of arrays p, p_w, and t as a function of h:
   $P(h) = \mathcal{B}^m(p(h))$
   $P_w(h) = \mathcal{B}^m(p_w(h))$
   $T(h) = \mathcal{B}^m(T(h))$

2. Compute lapse rate $t_r = \dfrac{\partial t}{\partial h}$ in the troposphere in a certain range of heights: $[h_{k_{\min}}, h_{k_{\max}}]$.
   We select $k_{\max} = \max k : \ h[k] < h_{\max}$, where $h_{\max} = 9000$ m, and $k_{\min} = k_{\max}/2$. Lapse rate in the range $[h_{k_{\min}}, h_{k_{\max}}]$ is computed with least squares by fitting two parameters of the straight line: $t(h) = t_0 + t_r * h$. Here is the algorithm:

   $k_{\max} = 1$
   for $k = 1, 2, \ldots\ n$ do
   if $h[k] < h_{\max}$ then $k_{\max} = k$

   at = 0; aht = 0; a = 0; ah = 0; ahh = 0;
   for $k = k_{\min}, k_{\min} + 1 \ldots\ k_{\max}$ do
   a := a + 1
   ah := ah + $h[k]$
   ahh := ahh + $h[k]^2$
   at := at + $t[k]$
   aht := aht + $h[k] * t[k]$
   Det = at*ahh - ah^2
   t_r = (a*aht - ah*at)/Det

3. Transform the column of data to the uniform grid

$\bar{n} := 125$
$\mu_1 := 20.25319$
$\mu_2 := 1200.0$
$\mu_3 := 107.30782$
for $k = 1, 2, \ldots \bar{n} - 1, \bar{n}$ do
  $\bar{h}[k] = \exp((k - \mu_3)/\mu_1) - \mu_2$
  $\bar{g}[k] = g_{eq} * (1 - 2 * (\bar{h}[k]/R_e) + 3 * (\bar{h}[k]/R_e)^2)$
  if $\bar{h}[k] < h[1]$ then #Extrapolation below the lowest layer
    $\bar{t}[k] = t[1] + T_r * (\bar{h}[k] - h[1])$
    $\bar{p}_w[k] = p_w[1] + (\bar{t}[k]/t[1]) * *(-\bar{g}[k]\, M_w/(R * T_r))$
    $\bar{p}[k] = p[1] + (\bar{t}[k]/t[1]) * *(-\bar{g}[k]\, M_d/(R * T_r))$
  else if $\bar{h}[k] > h[n]$ then #Extrapolation above the highest layer
    $\bar{t}[k] = t[n]$
    $\bar{p}_w[k] = p_w[n] * \exp(-\bar{g}[k]\, M_w * (\bar{h}_k - h[n])/(R * \bar{t}[k]))$
    $\bar{p}[k] = p[n] * \exp(-\bar{g}[k]\, M_d * (\bar{h}_k - h[n])/(R * \bar{t}[k]))$
  else #B-spline interpolation
    Find the pivotal knot l for $h[k], \max l : \ \bar{h}[k] \le h[l]$:
    $l := 1$
    for $r = 2, 3, \ldots n - 1$ do
      if $\bar{h}[k] < h[r]$ then $l := r$
    $\bar{p}[k] = 0$
    $\bar{p}_w[k] = 0$
    $\bar{t}[k] = 0$
    for $r = l - m, l - m + 1 \ldots l$ do
      $\bar{p}[k] = \bar{p}[k] + P[l] * B_r^m(h)$
      $\bar{p}_w[k] = \bar{p}_w[k] + P_w[l] * B_r^m(h)$
      $\bar{t}[k] = \bar{t}[k] + T[l] * B_r^m(h)$

5.2.4. Compute refractivity using the state of the atmosphere

Input data:

1) $\bar{h}$ 1D array of grid point geometric heights above geoid in m
2) $\bar{\varphi}$ 1D array of grid point geodetic latitudes in range $[-\pi/2, +\pi/2]$ in rad
3) $\bar{\lambda}$ 1D array of grid point longitudes in range $[0, 2\pi]$ in rad
4) $\bar{p}$ 3D array of total air pressure in Pa
5) $\bar{p}_w$ 3D array of partial water vapor pressure in Pa
6) $\bar{t}$ 3D array of air temperature in K

Input 3D arrays should be sized as $[1-m : n_h, 1-m : n_\lambda, 1-m : n_\varphi]$, where n_h is the dimension over layers in the regular grid ($n_h = 125$). Their elements with indices < 1 are not defined.

Output data:

1) r 3D array of air refractivity at 532 nm.

Output 3D array r should be sized as $[1 - m : n_h, 1 - m : n_\lambda, 1 - m : n_\varphi]$. Its elements with indices $n_h, n_\lambda, n_\varphi$ are not defined.

for $i = 1, 2, \ldots, n_\varphi - 1, n_\varphi$ do
$\varphi[i] = -\pi/2 + (i-1) * \pi/(n_\varphi - 1)$
for $j = 1, 2, \ldots, n_\lambda - 1, n_\lambda, n_\lambda + 1$ do
for $k = 1, 2, \ldots, n_h - 1, n_h$ do
$z_m = 1.0/($
$-\bar{p}[k,j,i]/\bar{t}[k,j,i] * (a_0 + a_1 * (\bar{t}[k,j,i] - t_0) + a_2 * (\bar{t}[k,j,i] - t_0)^2)$
$+\bar{p}_w[k,j,i]/\bar{t}[k,j,i] * (b_0 + b_1 * (\bar{t}[k,j,i] - t_0))$
$+\bar{p}_w^2[k,j,i]/(\bar{p}[k,j,i] * \bar{t}[k,j,i]) * (c_0 + c_1 * (\bar{t}[k,j,i] - t_0))$
$+\bar{p}^2[k,j,i]/\bar{t}^2[k,j,i] * e_0$
$+\bar{p}_w^2[k,j,i]/\bar{t}^2[k,j,i] * f_0$
$)$
$r[k,j,i] = (S_t * \bar{p}[k,j,i] + S_w * \bar{p}_w[k,j,i])/\bar{t}[k,j,i] * z_m$

S_t and S_w are constants.

5.2.5. Expand refractivity into 3D B-spline basis

Perform 3D B-spline transform following algorithm B.4:

$R(i,j,k) = \mathcal{B}^m_{i,j,k} r(i,j,k)$

Since the input 3D array r is sized as $[1-m : n_h, 1-m : n_\lambda, 1-m : n_\varphi]$, the output 3D array R will fit the same place and replace original array r.

5.2.6. Write down 3D B-spline transform of air refractivity

The array of 3D B-spline transform R is written in a file. File naming convention: refr_d@@@@@@@@_t@@@, where d@@@@@@@@ is date in yyyymmdd format. The first 4 characters are the year, following two characters are the month number with leading zeros, and last three characters are the day of the month with leading zeros if necessary. t@@@ is time in hhmm format. The first two characters are the hour, the last two characters are the minute. Since the output is generated with a step of 3 hours, the following values are valid: 0000, 0300, 0600, 0900, 1200, 1500, 1800, 2100.

5.3. Computation of path delay

5.3.1. Determination of time range for path delay

The time range for path delay computation is determined on a base of a definition of the data chunk to be processed. The beginning of the interpolation epoch is determined as the latest epoch that is less than the first observation minus one time step of numerical weather model (3 hours for GEOS-FPIT). The end of the interpolation epoch is determined as the earliest epoch that is greater than the last epoch of observation plus one one time step of numerical weather model. Here is an exanple: let the time range of observations be [2014.04.13T07:23:00, 2014.04.13T17:45:00]. Then the interpolation time range is [2014.04.13T03:00:00, 2014.04.13T21:00:00]. An extra epoch at the beginning and the end of interpolation range is added for improving the accuracy of interpolation.

5.3.2. Reading 3D B-spline refractivity transforms

The coefficients for 3D B-splines for n_t time epochs are read and form a 4D array. The 1st axis is height, the second axis is longitude, the third axis is latitude, and the fourth axis is time.

5.3.3. Compute 4D B-spline refractivity transform

This is done using steps 10–12 of 4D B-spline expansion algorithm described in section B.5. Basically, the expansion is performed in two steps: 1) the 4D array is extended over time axis by adding three extra nodes with index -2, -1, and 0; 2) in a cycle over axes 1, 2, and 3 a subsection of refractivity $r[i, j, k, -2 : n_l]$ is extracted. A 1D B-spline transform is performed and the result is written back in subsection $r[i, j, k, -2 : n_l]$.

5.3.4. Compute path delays

Path delay in zenith direction is computed by integration along the height axis from the surface to the top of the atmosphere. It should be noted that the height axis has zero at the sea level (or more precisely, at the geoid). If the a priori position of the ground point is referred to the height above the reference ellipsoid, the ellipsoid height should be converted to the ortho-height (i.e. height above the geoid): $h_o(\varphi, \lambda) = h_e(\varphi, \lambda) - u(\varphi, \lambda)$. Since the 2D B-spline transform of geoid undulation has already been computed, we compute $u(\varphi, \lambda)$ using this expansion:

$$u(\varphi, \lambda) = \sum_{-2+j}^{j} \sum_{-2+i}^{i} U_{ij} B_j^m(\varphi)\, B_i^m(\lambda)$$

where U is the B-spine transform of geoid undulations, i and j are so called pivotal indices: $\max i : \lambda_i < \lambda, \max j : \varphi_j < \varphi$.

Finally, zenith path delay is computed as

$$d(h_o, \lambda, \varphi_{gd}, t) = \sum_{l=l_p-m}^{l=l_p} \sum_{k=k_p-m}^{k=k_p} \sum_{j=j_p-m}^{j=j_p} \sum_{i=i_p-m}^{i=d_1-1} R_{ijkl}\, B_l^m(t)\, B_k^m(\varphi_{gd})\, B_j^m(\lambda)\, J_i^m(h_o),$$

where h_o is surface ortho-height, and i, j, k, l are pivotal elements for ortho-height, longitude, latitude, and time respectively.

Its derivative over height is computed as

$$\frac{\partial d(h_o, \lambda, \varphi_{gd}, t)}{\partial h_o} = \sum_{l=l_p-m}^{l=l_p} \sum_{k=k_p-m}^{k=k_p} \sum_{j=j_p-m}^{j=j_p} \sum_{i=i_p-m}^{i=d_1-1} R_{ijkl}\, B_l^m(t)\, B_k^m(\varphi_{gd})\, B_j^m(\lambda)\, B_i^m(h_o).$$

6. Test data and results

6.1. Test of computation of the state of atmosphere from the output of GEOS-FPIT numerical weather model

Date: 2014.02.25-12:00:00
GEOS-FPIT data file: DAS.fpit.asm.inst3_3d_asm_Nv.GEOS591.20140225_1200.V01.nc4
Retrieved by command

```
wget --http-password="" http://aurapar2u.ecs.nasa.gov/goldsfs1/data/GEOS5/
DFPITI3NVASM.5.9.1/2014/056/.hidden/DAS.fpit.asm.inst3_3d_asm_Nv.GEOS591.20140225_1200.V01.nc4
```

Table below contains parameters of the state of the atmosphere for the column of air at geodetic latitude -88.0°and longitude 349.375° at 2014.02.25-12:00:00. Indices in the GEOS-FPIT original grid: (5,560).

Table 1: Raw data: pressure thickness δp_i and air temperature $t[i]$ and intermediate results: preliminary gravity acceleration $g[i] = g_{\text{geoid}} * (g_0 + g_1 * \ln(p[i]))$, air compressibility Zm$[i]$, and and right hand-side for the hypsometric equation $y[i]$. Order of layers: from the ground to top.

Layer	$\Delta p[i]$	t[i]	g[i]	Zm[i]	y[i]
72	1.000000	200.31622	9.60178376	1.00000003	$3.9916380 \cdot 10^{+03}$
71	1.270000	204.54309	9.61335325	1.00000005	$2.3174349 \cdot 10^{+03}$
70	1.488501	212.04710	9.62199758	1.00000007	$1.5755816 \cdot 10^{+03}$
69	1.841500	221.49547	9.62912240	1.00000009	$1.1624261 \cdot 10^{+03}$
68	2.334500	231.56816	9.63555167	1.00000010	$8.8800047 \cdot 10^{+02}$
67	3.035800	241.27260	9.64164867	1.00000011	$6.8709835 \cdot 10^{+02}$
66	3.979201	250.14551	9.64759046	1.00000013	$5.3305223 \cdot 10^{+02}$
65	5.185400	256.04990	9.65341938	1.00000015	$4.1054464 \cdot 10^{+02}$
64	6.717703	258.53607	9.65913553	1.00000019	$3.1362148 \cdot 10^{+02}$
63	8.651503	260.68179	9.66473889	1.00000024	$2.4056639 \cdot 10^{+02}$
62	11.076500	262.65546	9.67022946	1.00000030	$1.8541326 \cdot 10^{+02}$
61	14.097301	264.40204	9.67560727	1.00000038	$1.4356198 \cdot 10^{+02}$
60	17.835503	268.26874	9.68087229	1.00000045	$1.1265636 \cdot 10^{+02}$
59	22.430611	270.98141	9.68602449	1.00000055	$8.8496741 \cdot 10^{+01}$
58	28.107002	270.46353	9.69106970	1.00000071	$6.9050418 \cdot 10^{+01}$
57	35.028000	266.77374	9.69601235	1.00000098	$5.3511164 \cdot 10^{+01}$
56	43.418015	262.37238	9.70085243	1.00000136	$4.1556302 \cdot 10^{+01}$
55	53.524017	256.92145	9.70558999	1.00000191	$3.2293087 \cdot 10^{+01}$
54	65.622009	251.55862	9.71022500	1.00000266	$2.5218099 \cdot 10^{+01}$
53	80.014038	246.77486	9.71475748	1.00000362	$1.9829464 \cdot 10^{+01}$
52	97.023010	243.30896	9.71918740	1.00000479	$1.5749940 \cdot 10^{+01}$
51	117.000000	239.39148	9.72351477	1.00000635	$1.2546239 \cdot 10^{+01}$
50	140.304077	238.16663	9.72773961	1.00000797	$1.0156474 \cdot 10^{+01}$
49	167.309998	234.87671	9.73186189	1.00001032	$8.1909187 \cdot 10^{+00}$
48	198.395996	232.69888	9.73588164	1.00001303	$6.6694581 \cdot 10^{+00}$

Layer	$\Delta p[i]$	t[i]	g[i]	Zm[i]	y[i]
47	236.650024	230.90681	9.73982131	1.00001628	$5.4605139 \cdot 10^{+00}$
46	281.370117	229.99455	9.74369943	1.00001998	$4.5011072 \cdot 10^{+00}$
45	333.450073	230.41312	9.74751600	1.00002388	$3.7429881 \cdot 10^{+00}$
44	393.880127	230.03204	9.75127099	1.00002886	$3.1110982 \cdot 10^{+00}$
43	463.739990	230.14713	9.75496440	1.00003448	$2.5992625 \cdot 10^{+00}$
42	544.220215	229.64398	9.75859628	1.00004151	$2.1723125 \cdot 10^{+00}$
41	636.540283	229.14737	9.76216660	1.00004981	$1.8209958 \cdot 10^{+00}$
40	746.850098	229.53174	9.76568648	1.00005874	$1.5361456 \cdot 10^{+00}$
39	874.399902	229.43329	9.76916533	1.00006970	$1.2957180 \cdot 10^{+00}$
38	1021.550293	229.21729	9.77260315	1.00008270	$1.0945476 \cdot 10^{+00}$
37	1190.890137	229.18913	9.77599991	1.00009763	$9.2722231 \cdot 10^{-01}$
36	1385.340820	228.92416	9.77935562	1.00011548	$7.8623600 \cdot 10^{-01}$
35	1629.729492	228.70306	9.78269243	1.00013636	$6.6742767 \cdot 10^{-01}$
34	1917.401367	228.39130	9.78602928	1.00016127	$5.6634343 \cdot 10^{-01}$
33	2255.599609	228.12888	9.78936613	1.00019057	$4.8067011 \cdot 10^{-01}$
32	2653.701172	228.66145	9.79270297	1.00022217	$4.0937910 \cdot 10^{-01}$
31	2915.396240	228.27863	9.79592942	1.00026165	$3.4914029 \cdot 10^{-01}$
30	3046.896973	226.77856	9.79890031	1.00031013	$3.0002206 \cdot 10^{-01}$
29	3259.590820	226.24045	9.80163545	1.00035753	$2.6189868 \cdot 10^{-01}$
28	3459.401367	220.08844	9.80419711	1.00044868	$2.2481815 \cdot 10^{-01}$
27	2483.254639	217.79785	9.80622378	1.00051416	$2.0151513 \cdot 10^{-01}$
26	2438.766113	218.36220	9.80776304	1.00054907	$1.8741077 \cdot 10^{-01}$
25	2371.893555	220.16035	9.80916360	1.00057070	$1.7646774 \cdot 10^{-01}$
24	2334.121094	222.67993	9.81044706	1.00058268	$1.6764988 \cdot 10^{-01}$
23	2294.205811	225.50577	9.81163562	1.00058890	$1.6021017 \cdot 10^{-01}$
22	2278.551758	227.89062	9.81274601	1.00059706	$1.5336463 \cdot 10^{-01}$
21	2239.467285	229.71201	9.81378712	1.00060890	$1.4693170 \cdot 10^{-01}$
20	2239.384766	231.42410	9.81476959	1.00062021	$1.4109696 \cdot 10^{-01}$
19	2212.899414	233.30623	9.81570177	1.00062817	$1.3591963 \cdot 10^{-01}$
18	1467.517090	234.87262	9.81644161	1.00063362	$1.3198131 \cdot 10^{-01}$
17	1466.509399	236.19720	9.81701290	1.00063648	$1.2907773 \cdot 10^{-01}$
16	1465.506348	237.44476	9.81756834	1.00063963	$1.2629102 \cdot 10^{-01}$
15	1452.071045	238.21530	9.81810653	1.00064765	$1.2341705 \cdot 10^{-01}$
14	1452.081055	239.29211	9.81862859	1.00065165	$1.2085724 \cdot 10^{-01}$
13	1161.472168	240.29382	9.81908733	1.00065445	$1.1867812 \cdot 10^{-01}$
12	868.057556	241.00342	9.81943662	1.00065719	$1.1701757 \cdot 10^{-01}$
11	866.946899	241.74799	9.81973058	1.00065772	$1.1570837 \cdot 10^{-01}$
10	865.525513	241.69543	9.82001997	1.00066769	$1.1405964 \cdot 10^{-01}$
9	865.521362	241.88831	9.82030511	1.00067466	$1.1257238 \cdot 10^{-01}$
8	865.534912	242.34647	9.82058634	1.00067828	$1.1124852 \cdot 10^{-01}$
7	863.664307	242.74982	9.82086348	1.00068244	$1.0993695 \cdot 10^{-01}$
6	861.897339	243.06985	9.82113636	1.00068752	$1.0862593 \cdot 10^{-01}$
5	861.886841	243.03839	9.82140538	1.00069698	$1.0719412 \cdot 10^{-01}$
4	861.905518	242.79001	9.82167092	1.00070925	$1.0570413 \cdot 10^{-01}$

Layer	$\Delta p[i]$	t[i]	g[i]	Zm[i]	y[i]
3	860.014526	242.05850	9.82193279	1.00072800	$1.0404416 \cdot 10^{-01}$
2	859.705566	240.05029	9.82219104	1.00076456	$1.0188073 \cdot 10^{-01}$
1	1052.805664	232.66653	9.82247447	1.00088317	$9.7371944 \cdot 10^{-02}$

Table 2: Intermediate results of computation gravity acceleration $g[i]$, total pressure $p[i]$, partial pressure of water vapor $p_w[i]$, dry pressure $p_d[i]$, and height above the geoid $h[i]$. Order of layers: from top to the ground.

Layer	g[i]	p[i]	p_w[i]	p_d[i]	h[i]
1	9.82400803	69759.05402	$1.17192 \cdot 10^{+01}$	69747.33482	2632.974
2	9.82371440	68802.79840	$2.47031 \cdot 10^{+01}$	68778.09533	2728.322
3	9.82344149	67942.93835	$3.20390 \cdot 10^{+01}$	67910.89939	2816.946
4	9.82316348	67081.97833	$3.39323 \cdot 10^{+01}$	67048.04599	2907.233
5	9.82288099	66220.08215	$3.41871 \cdot 10^{+01}$	66185.89505	2998.976
6	9.82259465	65358.19006	$3.39866 \cdot 10^{+01}$	65324.20350	3091.975
7	9.82230438	64495.40924	$3.23242 \cdot 10^{+01}$	64463.08504	3186.252
8	9.82201005	63630.80963	$3.08669 \cdot 10^{+01}$	63599.94277	3281.856
9	9.82171191	62765.28149	$2.94604 \cdot 10^{+01}$	62735.82105	3378.699
10	9.82141009	61899.75806	$2.92146 \cdot 10^{+01}$	61870.54349	3476.744
11	9.82110375	61033.52185	$2.90664 \cdot 10^{+01}$	61004.45549	3576.258
12	9.82079304	60166.01962	$2.70384 \cdot 10^{+01}$	60138.98125	3677.202
13	9.82042505	59151.25476	$2.52217 \cdot 10^{+01}$	59126.03307	3796.758
14	9.81994341	57844.47815	$2.20922 \cdot 10^{+01}$	57822.38591	3953.247
15	9.81939779	56392.40210	$1.98510 \cdot 10^{+01}$	56372.55115	4130.539
16	9.81883741	54933.61340	$1.75471 \cdot 10^{+01}$	54916.06629	4312.640
17	9.81826149	53467.60553	$1.49058 \cdot 10^{+01}$	53452.69977	4499.810
18	9.81767248	52000.59229	$1.27621 \cdot 10^{+01}$	51987.83016	4691.248
19	9.81691439	50160.38403	$1.02525 \cdot 10^{+01}$	50150.13154	4937.668
20	9.81596636	47934.24194	$7.91348 \cdot 10^{+00}$	47926.32847	5245.870
21	9.81497495	45694.81592	$6.26463 \cdot 10^{+00}$	45688.55129	5568.222
22	9.81393230	43435.80640	$5.08046 \cdot 10^{+00}$	43430.72593	5907.288
23	9.81283046	41149.42761	$4.03982 \cdot 10^{+00}$	41145.38779	6265.660
24	9.81166493	38835.26416	$3.07545 \cdot 10^{+00}$	38832.18871	6644.812
25	9.81042173	36482.25684	$2.32599 \cdot 10^{+00}$	36479.93085	7049.309
26	9.80907858	34076.92700	$1.80019 \cdot 10^{+00}$	34075.12681	7486.409
27	9.80761069	31615.91663	$1.47922 \cdot 10^{+00}$	31614.43741	7964.209
28	9.80567267	28644.58862	$1.11363 \cdot 10^{+00}$	28643.47499	8595.200
29	9.80316826	25285.09253	$6.47272 \cdot 10^{-01}$	25284.44526	9410.878
30	9.80045571	22131.84863	$4.97196 \cdot 10^{-01}$	22131.35144	10294.704
31	9.79749937	19150.70203	$1.73043 \cdot 10^{-01}$	19150.52898	11258.382
32	9.79427309	16366.15332	$1.17497 \cdot 10^{-01}$	16366.03582	12310.555

Layer	g[i]	p[i]	p_w[i]	p_d[i]	h[i]
33	9.79094079	13911.50293	$8.81496 \cdot 10^{-02}$	13911.41478	13397.851
34	9.78761096	11825.00244	$6.49573 \cdot 10^{-02}$	11824.93748	14484.902
35	9.78427662	10051.43701	$4.89417 \cdot 10^{-02}$	10051.38807	15573.986
36	9.78093933	8543.90186	$3.80657 \cdot 10^{-02}$	8543.86379	16664.599
37	9.77757965	7255.78638	$3.22436 \cdot 10^{-02}$	7255.75413	17763.094
38	9.77417756	6149.56616	$2.81247 \cdot 10^{-02}$	6149.53804	18876.044
39	9.77073343	5201.59106	$2.49072 \cdot 10^{-02}$	5201.56616	20003.344
40	9.76724506	4390.96606	$2.18033 \cdot 10^{-02}$	4390.94426	21145.737
41	9.76372001	3699.27087	$1.89621 \cdot 10^{-02}$	3699.25191	22300.776
42	9.76014459	3108.89062	$1.60622 \cdot 10^{-02}$	3108.87456	23472.964
43	9.75649790	2604.91052	$1.35877 \cdot 10^{-02}$	2604.89693	24669.194
44	9.75278933	2176.10046	$1.18299 \cdot 10^{-02}$	2176.08863	25886.422
45	9.74901536	1812.43536	$1.00614 \cdot 10^{-02}$	1812.42530	27125.838
46	9.74518243	1505.02527	$8.77513 \cdot 10^{-03}$	1505.01649	28385.367
47	9.74128609	1246.01520	$7.39101 \cdot 10^{-03}$	1246.00781	29666.510
48	9.73730240	1028.49219	$6.21445 \cdot 10^{-03}$	1028.48597	30977.182
49	9.73320827	845.63919	$5.07837 \cdot 10^{-03}$	845.63411	32325.047
50	9.72895317	691.83215	$4.19790 \cdot 10^{-03}$	691.82796	33726.825
51	9.72455975	563.18011	$3.49714 \cdot 10^{-03}$	563.17662	35175.155
52	9.72000897	456.16861	$2.89681 \cdot 10^{-03}$	456.16571	36676.419
53	9.71528147	367.65009	$2.39406 \cdot 10^{-03}$	367.64769	38237.126
54	9.71036433	294.83206	$1.96353 \cdot 10^{-03}$	294.83010	39861.679
55	9.70523492	235.25905	$1.59660 \cdot 10^{-03}$	235.25745	41557.710
56	9.69987862	186.78803	$1.28601 \cdot 10^{-03}$	186.78675	43330.238
57	9.69430649	147.56503	$1.02653 \cdot 10^{-03}$	147.56400	45175.788
58	9.68852681	115.99752	$8.12118 \cdot 10^{-04}$	115.99671	47091.815
59	9.68258148	90.72872	$6.36884 \cdot 10^{-04}$	90.72808	49064.602
60	9.67653687	70.59566	$4.96402 \cdot 10^{-04}$	70.59516	51072.252
61	9.67044790	54.62926	$3.84153 \cdot 10^{-04}$	54.62887	53096.600
62	9.66429320	42.04236	$2.95725 \cdot 10^{-04}$	42.04206	55144.815
63	9.65805397	32.17836	$2.26315 \cdot 10^{-04}$	32.17813	57223.228
64	9.65174031	24.49375	$1.72122 \cdot 10^{-04}$	24.49358	59328.565
65	9.64535658	18.54220	$1.30169 \cdot 10^{-04}$	18.54207	61459.447
66	9.63894443	13.95990	$9.79419 \cdot 10^{-05}$	13.95980	63602.033
67	9.63262139	10.45240	$7.32751 \cdot 10^{-05}$	10.45233	65717.023
68	9.62635477	7.76725	$5.44116 \cdot 10^{-05}$	7.76720	67815.287
69	9.62008404	5.67925	$3.97432 \cdot 10^{-05}$	5.67921	69917.073
70	9.61332344	4.01425	$2.79614 \cdot 10^{-05}$	4.01422	72185.457
71	9.60572876	2.63500	$1.81498 \cdot 10^{-05}$	2.63498	74736.689
72	9.59519354	1.50000	$9.68205 \cdot 10^{-06}$	1.49999	78280.983

6.2. Test of refractivity computation

Table 3: Intermediate results of computation height above the geoid $h[i]$, total pressure $p[i]$, partial pressure of water vapor $p_w[i]$, air temperature $t[i]$, and air refractivity at 532 nm $r[i]$ at the uniform grid by using interpolation.

Layer	h[i]	p[i]	p_w[i]	t[i]	r[i]
1	-1000.000	117854.89131	16.23614632	242.36836	0.0003983650
2	-989.877	117685.87530	16.22166707	242.34133	0.0003978376
3	-979.242	117508.54730	16.20646728	242.31293	0.0003972842
4	-968.069	117322.51140	16.19051176	242.28309	0.0003967036
5	-956.330	117127.35573	16.17376378	242.25174	0.0003960944
6	-943.996	116922.64780	16.15618469	242.21880	0.0003954553
7	-931.039	116707.93814	16.13773420	242.18420	0.0003947849
8	-917.426	116482.75677	16.11837005	242.14785	0.0003940818
9	-903.123	116246.61283	16.09804797	242.10965	0.0003933442
10	-888.097	115998.99731	16.07672189	242.06953	0.0003925707
11	-872.311	115739.37874	16.05434353	242.02737	0.0003917596
12	-855.725	115467.20203	16.03086229	241.98308	0.0003909091
13	-838.300	115181.89240	16.00622556	241.93654	0.0003900175
14	-819.992	114882.84836	15.98037806	241.88765	0.0003890826
15	-800.759	114569.44783	15.95326234	241.83629	0.0003881028
16	-780.551	114241.04238	15.92481825	241.78233	0.0003870758
17	-759.321	113896.95851	15.89498301	241.72563	0.0003859995
18	-737.017	113536.50014	15.86369135	241.66607	0.0003848718
19	-713.583	113158.94131	15.83087486	241.60349	0.0003836902
20	-688.964	112763.53185	15.79646240	241.53774	0.0003824525
21	-663.098	112349.49635	15.76037995	241.46867	0.0003811561
22	-635.923	111916.02941	15.72255014	241.39610	0.0003797985
23	-607.373	111462.30086	15.68289262	241.31985	0.0003783770
24	-577.377	110987.45243	15.64132369	241.23975	0.0003768888
25	-545.864	110490.59952	15.59775635	241.15559	0.0003753312
26	-512.755	109970.83036	15.55210016	241.06718	0.0003737012
27	-477.971	109427.20683	15.50426119	240.97429	0.0003719957
28	-441.426	108858.76424	15.45414187	240.87669	0.0003702117
29	-403.031	108264.51286	15.40164103	240.77416	0.0003683460
30	-362.693	107643.43882	15.34665381	240.66644	0.0003663952
31	-320.314	106994.50448	15.28907158	240.55327	0.0003643559
32	-275.789	106316.64929	15.22878184	240.43436	0.0003622248
33	-229.011	105608.79431	15.16566842	240.30944	0.0003599983
34	-179.865	104869.84003	15.09961116	240.17820	0.0003576727
35	-128.232	104098.67197	15.03048616	240.04031	0.0003552445
36	-73.985	103294.16167	14.95816567	239.89545	0.0003527098
37	-16.993	102455.17004	14.88251816	239.74325	0.0003500649

Layer	h[i]	p[i]	p_w[i]	t[i]	r[i]
38	42.884	101580.55087	14.80340836	239.58335	0.0003473058
39	105.792	100669.15488	14.72069742	239.41535	0.0003444289
40	171.883	99719.83400	14.63424292	239.23886	0.0003414302
41	241.320	98731.44656	14.54389912	239.05343	0.0003383057
42	314.271	97702.86202	14.44951701	238.85861	0.0003350517
43	390.915	96632.96891	14.35094471	238.65394	0.0003316642
44	471.438	95520.67992	14.24802759	238.43890	0.0003281395
45	556.036	94364.93968	14.14060857	238.21298	0.0003244737
46	644.917	93164.73414	14.02852859	237.97563	0.0003206633
47	738.295	91919.09978	13.91162707	237.72626	0.0003167048
48	836.401	90627.12951	13.78974199	237.46427	0.0003125947
49	939.471	89287.98954	13.66271102	237.18902	0.0003083298
50	1047.759	87900.92645	13.53037174	236.89984	0.0003039071
51	1161.528	86465.28275	13.39256262	236.59602	0.0002993240
52	1281.054	84980.50364	13.24912317	236.27682	0.0002945779
53	1406.631	83446.16114	13.09989579	235.94147	0.0002896668
54	1538.564	81861.95891	12.94472575	235.58915	0.0002845888
55	1677.174	80227.75858	12.78346329	235.21899	0.0002793428
56	1822.800	78543.58376	12.61596353	234.83010	0.0002739277
57	1975.796	76809.64850	12.44208892	234.42152	0.0002683435
58	2136.536	75026.37080	12.26171016	233.99227	0.0002625902
59	2305.413	73194.38490	12.07470718	233.54128	0.0002566690
60	2482.836	71314.57445	11.88097221	233.06748	0.0002505814
61	2669.240	69393.00719	15.84158170	235.68621	0.0002411033
62	2865.078	67482.52052	33.38673499	242.48098	0.0002278616
63	3070.829	65553.19160	34.15226673	243.08998	0.0002207856
64	3286.994	63584.63488	30.78623875	242.32232	0.0002148324
65	3514.099	61573.17230	29.30826223	241.75207	0.0002085242
66	3752.700	59523.58453	25.88128614	240.52713	0.0002026091
67	4003.376	57430.81757	21.32229927	238.95550	0.0001967719
68	4266.741	55298.18959	18.18013518	237.65818	0.0001904984
69	4543.436	53130.44304	14.37425771	235.88426	0.0001844074
70	4834.135	50926.96376	11.25903990	233.95490	0.0001782172
71	5139.548	48692.78546	8.62009258	232.05243	0.0001717954
72	5460.419	46433.64957	6.74296023	230.26466	0.0001650957
73	5797.531	44156.50304	5.42833885	228.51334	0.0001582010
74	6151.706	41865.69166	4.35953457	226.31768	0.0001514477
75	6523.806	39562.39580	3.35876517	223.56702	0.0001448759
76	6914.740	37251.69107	2.54440305	220.91510	0.0001380507
77	7325.461	34945.68546	1.96187275	218.89947	0.0001306953
78	7756.971	32661.62031	1.59902785	217.83393	0.0001227473
79	8210.320	30417.74169	1.34809229	218.23416	0.0001141003
80	8686.616	28242.19659	1.04850555	220.75340	0.0001047251
81	9187.020	26158.56621	0.72971578	224.87381	0.0000952159

Layer	h[i]	p[i]	p_w[i]	t[i]	r[i]
82	9712.750	24157.76328	0.59953550	226.97789	0.0000871144
83	10265.090	22230.60930	0.50715606	226.78026	0.0000802328
84	10845.387	20374.07965	0.27565731	227.47883	0.0000733045
85	11455.054	18595.49820	0.14939277	228.53200	0.0000665950
86	12095.579	16899.72353	0.12158713	228.73535	0.0000604668
87	12768.524	15284.02875	0.10595484	228.40900	0.0000547629
88	13475.530	13750.83313	0.08614009	228.11916	0.0000493311
89	14218.320	12305.47984	0.06980713	228.28439	0.0000441130
90	14998.706	10951.92909	0.05672408	228.56295	0.0000392122
91	15818.591	9691.51772	0.04602637	228.75128	0.0000346702
92	16679.973	8524.36900	0.03795480	228.92804	0.0000304709
93	17584.953	7450.54372	0.03300273	229.16147	0.0000266049
94	18535.738	6468.54541	0.02926336	229.20835	0.0000230933
95	19534.646	5576.38911	0.02619346	229.32012	0.0000198982
96	20584.115	4772.30072	0.02330182	229.55351	0.0000170115
97	21686.699	4052.30951	0.02045890	229.32567	0.0000144592
98	22845.092	3412.15963	0.01760109	229.27902	0.0000121774
99	24062.115	2849.40968	0.01473675	229.97790	0.0000101381
100	25340.736	2358.78817	0.01259136	230.06141	0.0000083894
101	26684.074	1934.46694	0.01064572	230.32777	0.0000068722
102	28095.404	1570.83820	0.00906586	230.05007	0.0000055871
103	29578.168	1262.30354	0.00748027	230.80107	0.0000044751
104	31135.980	1004.94241	0.00607477	232.91458	0.0000035304
105	32772.641	792.91628	0.00476423	235.99429	0.0000027492
106	34492.137	620.42664	0.00380989	238.78455	0.0000021260
107	36298.664	480.87228	0.00303638	242.24055	0.0000016243
108	38196.629	369.69758	0.00240587	246.67807	0.0000012263
109	40190.660	282.09170	0.00188624	252.60073	0.0000009138
110	42285.613	213.86695	0.00146092	259.24907	0.0000006750
111	44486.605	161.06283	0.00111665	265.22876	0.0000004969
112	46798.996	120.32035	0.00084181	270.05310	0.0000003646
113	49228.426	88.89694	0.00062413	270.86510	0.0000002685
114	51780.820	64.56100	0.00045403	266.82788	0.0000001980
115	54462.398	45.88449	0.00032271	263.11669	0.0000001427
116	57279.707	31.94456	0.00022467	260.62179	0.0000001003
117	60239.609	21.75148	0.00015278	257.68662	0.0000000691
118	63349.328	14.44143	0.00010133	251.04059	0.0000000471
119	66616.438	9.21924	0.00006461	237.18174	0.0000000318
120	70048.914	5.56796	0.00003896	220.89142	0.0000000206
121	73655.117	3.15325	0.00002189	207.18337	0.0000000125
122	77443.852	1.71219	0.00001124	201.19417	0.0000000070
123	81424.344	0.90037	0.00000581	200.31622	0.0000000037
124	85606.312	0.45804	0.00000296	200.31622	0.0000000019
125	89999.945	0.22607	0.00000146	200.31622	0.0000000009

6.3. Test of path delay computation

Data: GEOS-FPIT dataset for epochs

2014.02.25T00:00:00
2014.02.25T03:00:00
2014.02.25T06:00:00
2014.02.25T09:00:00
2014.02.25T12:00:00
2014.02.25T15:00:00
2014.02.25T18:00:00
2014.02.25T21:00:00
2014.02.26T00:00:00

Results of computation of path delay (in meter). The first line corresponds to the latitude and longitude of the previous example.

UTC Date	φ_{gd}	λ	h	u(φ, λ)	D
2014.02.25-12:00:00.0	-88.000	-10.625	2612.10	-29.107	1.669249
2014.02.25-12:06:40.0	-85.200	-10.800	2400.00	-13.409	1.718365
2014.02.25-12:13:20.0	-82.175	-11.128	2500.00	-15.958	1.688175

7. Used notation and numerical values of constants

7.1. List of symbols

A Azimuth counted from North to East of the satellite viewed from the footprint in rad.

GM Earth's gravitational constants m^3/s^2.

M_d molar mass of dry air in kg/mol.

M_w molar mass of water vapor in kg/mol.

p total atmospheric pressure in Pa.

p_w partial pressure of water vapor in Pa.

R universal gas constant J/(K·mol)

$R_\oplus$ Earth's equatorial radius.

S_t parameter of expression for refractivity of dry air in K/Pa.

S_w parameter of expression for refractivity of water vapor in K/Pa.

t air temperature in K.

z dimensionless compressibility of moist air. This variable accounts for deviation of state of the atmosphere from the ideal gas law.

d path delay in the atmosphere in m.

$f_\oplus$ the Earth's flattening: the ratio of the differences in lengths of polar and equatorial radius to the equatorial radius.

$e_\oplus$ eccentricity of the Earth's figure: $e_\oplus = \sqrt{2f_\oplus - f_\oplus^2}$.

g gravity acceleration in m/s^2.

g_n nominal gravity acceleration used in the numerical weather model in m/s^2.

g_0 ratio of the gravity acceleration at pressure level 1 Pa to the gravity acceleration at the geoid at the same latitude.

g_1 mean dependence of the Earth's gravity acceleration on logarithm of the total air pressure.

g_{eq} gravity acceleration at the Earth's equator in m/s^2.

g_{geoid} gravity acceleration at the geoid m/s^2.

h geometric height above reference ellipsoid in m.

h_o geometric height above the geoid (ortho-height) m.

h_s geometric height above the reference surface used in the numerical weather model (m).

k dimensionless parameter that describes change of gravity acceleration with latitude.

q specific humidity: ratio of mass of water vapor to the total mass of moist air.

r refractivity coefficient defined as $c - v/c$, where v is the speed of light in air and c is the speed of light in vacuum.

t air temperature in K.

u height of the geoid above the reference ellipsoid in m.

z zenith angle of the satellite viewed from the footprint in rad.

$\Omega_\oplus$ Earth's angular velocity in rad/s.

Φ geopotential in m^2/s^2.

PHIS increment of geopotential at the nominal surface with respect to the geoid in m^2/s^2.

λ longitude.

ρ air density in kg/m^3.

7.2. List of acronyms

7.3. Numerical values of constants

Table 4. Constants used in formula for refractivity

d_0	$2.380185 \cdot 10^{+14}$	m^{-1}
d_1	$5.792105 \cdot 10^{+10}$	—
d_2	$5.7362 \cdot 10^{+13}$	m^{-2}
d_3	$1.67917 \cdot 10^{+09}$	—
w_0	$2.95235 \cdot 10^{-06}$	—
w_1	$2.6422 \cdot 10^{-20}$	m^{-2}
w_2	$-3.2380 \cdot 10^{-34}$	m^{-4}
w_3	$4.028 \cdot 10^{-47}$	m^{-6}
C	1.022	—
R	8.314472	J/(K·mol)
M_d	0.02896546	kg/moll
M_w	0.01801528	kg/moll

Table 5. Constants in the expression for air compressibility

a_0	$1.58123 \cdot 10^{-6}$	$K \cdot Pa^{-1}$
a_1	$-2.933 \cdot 10^{-8}$	Pa^{-1}
a_2	$1.1043 \cdot 10^{-10}$	$K^{-1} \cdot Pa^{-1}$
b_0	$5.707 \cdot 10^{-6}$	$K \cdot Pa^{-1}$
b_1	$-2.051 \cdot 10^{-8}$	Pa^{-1}
c_0	$1.9898 \cdot 10^{-4}$	$K \cdot Pa^{-1}$
c_1	$-2.376 \cdot 10^{-6}$	Pa^{-1}
e_0	$1.83 \cdot 10^{-11}$	$K^2 \cdot Pa^{-2}$
f_0	$-7.65 \cdot 10^{-9}$	$K^2 \cdot Pa^{-2}$

Table 6. Coefficients of refractivity expression for ICESat1 ($\lambda = 1064.0$ nm) and ICESat2 ($\lambda = 532.0$ nm):

	S_t	S_w
ICESat-1	$7.8147358 \cdot 10^{-7}$	$-1.0604128 \cdot 10^{-7}$
ICESat-2	$8.1822296 \cdot 10^{-7}$	$-9.7331360 \cdot 10^{-8}$

Table 7. Numerical values of parameters related to gravity acceleration

g_{eq}	9.7803253359D0	m/s^2
g_0	0.975726	—
g_1	0.0020885	—
k	0.00193185265241	m/s^2
$GM_\oplus$	$3.986004418 \cdot 10^{14}$	m^3/s^2
$f_\oplus$	0.003352810665	—
$\Omega_\oplus$	$7.292115146706387 \cdot 10^{-5}$	rad/s

8. References

International Organization for Standardization, Standard Atmosphere, ISO 2533:1975, 1975.

de Boor C., A Practical Guide to Splines, Springer, 2001

Ciddor P., Refractive index of air: new equations for visible and near infrared, Applied Optics, 35(9), 1566–1573, 1996

Ciddor, P.E. & Hill, R.J, Refractive Index of Air. 2. Group Index, Applied Optics, 38(9), 1663–1667, 1999

Green, R.M. Spherical Astronomy, Cambridge University Press, 1985.

Landau, L.D., E.M. Lifshitz (1987), The Classical Theory of Fields, Volume 2, Butterworth-Heinemann.

Li X., Gotze, H.-J., (2001) Ellipsoid, geoid, gravity, geodesy, and geophysics, Geophysiscs, 66(6), 1660–1668.

Pavlis, N. K., S. A. Holmes, S. C. Kenyon, J. K. Factor (2012), The development and evaluation of the Earth Gravitational Model 2008 (EGM2008), J. Geophys. Res., 117(B4), 2156–2202

Petrov, Yu.P. (1965), Variational methods of the optimal control theory, Energia.

Picard, A., R.S. Davis, M. Gläser, and K. Fujii, Revised formula for the density of moist air (CIPM-2007), Metrologia, 45, 149–155, 2007

Rüeger, J. M., Refractive indices of light, infrared and radio waves in the atmosphere, UNISURV S-68, report from School of Surveying and Spatial Information System, Uni. of New South Wales, Sydney, Australia, 2002

Wahr, J., Geodesy and Gravity: Course Notes, 1996, Samiztat Press

A. Appendix

In the proposed algorithm we stated that slant path delay $D(z)$ at zenith angle z can be computed from the zenith path delay $D(0)$ as $D(z) = D(0)/\cos(z)$ with accuracy better than 1 mm provided $z < 5°$. We also stated that the offset of the footprint due to bending the light in the atmosphere is does not exceed 30 cm if $z < 5°$. This simplifies significantly computation of path delay in the atmosphere. In order to prove these statements, we need to compute slant path delay rigorously. The difficulty is to derive the trajectory of light in the atmosphere. This can be done by solving numerically differential equations of wave propagation. Next section explains how to do it.

A.1. Differential equations of wave propagation

According to the Fermat principle, the wave propagates through a heterogeneous continuous media follows a trajectory that minimizes its travel time. Fermat formulated this principle empirically 1662, but it can be derived theoretically from the electromagnetic wave equations that are solution of Maxwell equations (*Landau and Lifshitz*, 1988). Let us introduce the Cartesian coordinate system ξ, η, ζ and direct the axis ξ along the straight line that connects the emitter S and the expected footprint F (see Figure 2. Let us place axis $\vec{\eta}$ in the plane of the Earth's pole and footprint, and set $\vec{\zeta} = \vec{\xi} \times \vec{\eta}$. This coordinate system and the associated time argument t is thereafter denoted $\xi\eta\zeta t$. Then the Fermat principle defines the trajectory $\eta(\xi), \zeta(\xi)$ that minimizes the following functional:

$$\int_S^F (1 + n(\xi, \eta, \zeta)) \sqrt{1 + \left(\frac{d\eta}{d\xi}\right)^2 + \left(\frac{d\zeta}{d\xi}\right)^2}\, d\xi \longrightarrow \min, \tag{32}$$

where $n(\xi, \eta, \zeta)$ is the refractivity coefficient defined as (c-v)/c. (**NB:** Notation is changed in the Appendix: refractivity was denoted as r in the previous sections).

The additional time delay in the neutral atmosphere along the trajectory $\eta(\xi), \zeta(\xi)$ can be found by integration along the path:

$$\tau_{na} = \frac{1}{c} \int_0^\infty \left((1 + n\,(\xi, \eta, \zeta)) \sqrt{1 + \left(\frac{d\eta}{d\xi}\right)^2 + \left(\frac{d\zeta}{d\xi}\right)^2} - 1 \right) d\xi. \tag{33}$$

A general solution of the classical variation problem $\int_a^b F(x, y, y', z, z')\, dx \longrightarrow \min$, is expressed in the form of a system of Euler equations (e.g. *Petrov* (1965)):

$$\begin{cases} \dfrac{\partial F}{\partial y} - \dfrac{\partial^2 F}{\partial x\, \partial y'} - \dfrac{\partial^2 F}{\partial y\, \partial y'} y' - \dfrac{\partial^2 F}{\partial y'^2} y'' - \dfrac{\partial^2 F}{\partial z\, \partial y'} z' - \dfrac{\partial^2 F}{\partial z'\, \partial y'} z'' &= 0 \\ \dfrac{\partial F}{\partial z} - \dfrac{\partial^2 F}{\partial x\, \partial z'} - \dfrac{\partial^2 F}{\partial z\, \partial z'} z' - \dfrac{\partial^2 F}{\partial z'^2} z'' - \dfrac{\partial^2 F}{\partial y\, \partial z'} y' - \dfrac{\partial^2 F}{\partial y'\, \partial z'} y'' &= 0 \end{cases}. \tag{34}$$

Having substituted the functional from equation 32, we get the following system of non-linear

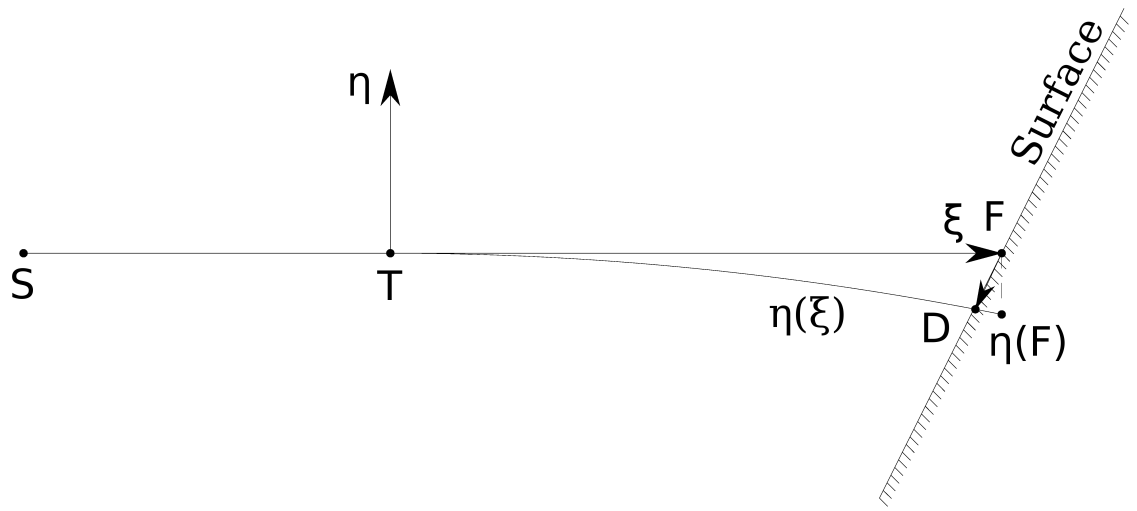

Figure 2. Geometry of photon propagation first from the point S (Satellite) to T(top of the atmosphere) along direction to nominal footprint F and then along the curved line $\xi(\eta)$ till it reaches the surface at point $D = \eta(F)/\cos(z)$.

equations that determines the trajectory of waves thought the inhomogeneous neutral atmosphere:

$$\begin{cases} \eta'' &= \left(\frac{n_\eta}{1+n}\left(1+\zeta'^2\right) - \frac{n_\xi}{1+n}\eta' - \frac{n_\zeta}{1+n}\eta'\zeta'\right)\left(1+\eta'^2+\zeta'^2\right) + \eta'\zeta'\zeta'' \\ \zeta'' &= \left(\frac{n_\zeta}{1+n}\left(1+\eta'^2\right) - \frac{n_\xi}{1+n}\zeta' - \frac{n_\eta}{1+n}\eta'\zeta'\right)\left(1+\eta'^2+\zeta'^2\right) + \eta'\zeta'\eta'' \end{cases}, \tag{35}$$

where notation η' means $\dfrac{d\eta}{d\xi}$ and n_ζ means $\dfrac{\partial n}{\partial \zeta}$.

We can split the linear and non-linear parts:

$$\begin{aligned} \eta'' = &\left(\frac{n_\eta}{1+n} - \frac{n_\xi}{1+n}\eta'\right) - \\ &\left(2\frac{n_\eta}{1+n}\zeta'^2 - \frac{n_\zeta}{1+n}\eta'\zeta' + \frac{n_\eta}{1+n}\eta'^2 - \frac{n_\xi}{1+n}\eta'^3 - \frac{n_\xi}{1+n}\eta'\zeta'^2\right. \\ &\left. -\frac{n_\zeta}{1+n}\eta'\zeta'^3 + \frac{n_\eta}{1+n}\eta'^2\zeta'^2 - \frac{n_\zeta}{1+n}\eta'^3\zeta' + \frac{n_\eta}{1+n}\zeta'^4 + \frac{n_\zeta}{1+n}\eta'\zeta'\zeta''\right) = 0 \end{aligned} \tag{36}$$

By analogy, we can write an equation for ζ'':

$$\begin{aligned} \zeta'' = &\left(\frac{n_\zeta}{1+n} - \frac{n_\xi}{1+n}\zeta'\right) - \\ &\left(2\frac{n_\zeta}{1+n}\eta'^2 - \frac{n_\eta}{1+n}\zeta'\eta' + \frac{n_\zeta}{1+n}\zeta'^2 - \frac{n_\xi}{1+n}\zeta'^3 - \frac{n_\xi}{1+n}\zeta'\eta'^2\right. \\ &\left. -\frac{n_\eta}{1+n}\zeta'\eta'^3 + \frac{n_\zeta}{1+n}\zeta'^2\eta'^2 - \frac{n_\eta}{1+n}\zeta'^3\eta' + \frac{n_\zeta}{1+n}\eta'^4 + \frac{n_\eta}{1+n}\zeta'\eta'\eta''\right) = 0 \end{aligned} \tag{37}$$

After combining equations (36)–(37), we finally get the system of two non-linear equations of the second order:

$$\begin{cases} (1+n)\eta''+n_\xi\eta'-n_\eta = 2n_\eta\zeta'^2-n_\zeta\eta'\zeta'+n_\eta\eta'^2-n_\xi\eta'^3-n_x\eta'\zeta'^2-n_\zeta\eta'\zeta'^3+n_\eta\eta'^2\zeta'^2-n_\zeta\eta'^3\zeta'+n_\eta\zeta'^4+n_\zeta\eta'\zeta'\zeta'' \\ (1+n)\zeta''+n_x\zeta'-n_\zeta = 2n_\zeta\eta'^2-n_\eta\zeta'\eta'+n_\zeta\zeta'^2-n_\xi\zeta'^3-n_\xi\zeta'\eta'^2-n_\eta\zeta'\eta'^3+n_\zeta\zeta'^2\eta'^2-n_\eta\zeta'^3\eta'+n_\zeta\eta'^4+n_\eta\zeta'\eta'\eta'' \end{cases} \tag{38}$$

System of equation (38) is exact. It should be stressed that no simplification was introduced in deriving the system 38. These equations are valid for an arbitrary field of the refractivity index in the atmosphere. Though, it can be shown that for computation of path delay integral through the atmosphere, this system can be simplified by discarding non-linear part in right hand-side and discarding the second equation of system 38.

$$(1+n)\,\eta'' + n_x\eta' - n_\eta = 0 \tag{39}$$

The relative error does not exceed $1 \cdot 10^{-7}$, at elevations of 20° above the horizon, which is more than sufficient for modeling ICESat-1 or ICESat-2.

These equations with respect to unknown function $\eta(\xi)$ should be augmented with the initial conditions. Since we are interested in computing the footprint offset, we assume that the position of the spacecraft and the direction of shooting are known. Selecting the direction of $\vec{\xi}$ axis gives us the fist boundary condition $\eta' = 0$. To simplify solving equations, we shift the origin of the coordinate system to the point at the straight line along the shooting direction where it intersects the nominal top of the atmosphere, 90 km for our computation that approximately corresponds atmospheric pressure 1Pa. In that case, the second boundary condition is $\eta(0) = 0$.

Finally, in order to find the photon trajectory through the heterogeneous atmosphere considered as a continuous media, we need to solve a linear differential equation of the second order with two initial conditions at the origin:

$$\begin{aligned} &\eta'' + \frac{n_\xi}{1+n(\xi,\eta)}\,\eta' - \frac{n_\eta}{1+n(\xi,\eta)} = 0 \\ &\eta'(0) = 0 \\ &\eta(0) = 0 \end{aligned} \tag{40}$$

The footprint displacements along north and east direction, D_N and D_E at the end of the photon trajectory $\eta(f)$ are evaluated as

$$\begin{aligned} D_N &= \eta(f)\frac{\cos A}{\cos z} \\ D_E &= \eta(f)\frac{\sin A}{\cos z} \end{aligned} \tag{41}$$

where A is the azimuth of the satellite as viewed from the footprint from north to east, and z is the zenith angle of the satellite viewed from the footprint.

A.2. Partial derivatives of the refractivity field

In order to integrate equation 40, we need to represent the 4D field of refractivity with a smooth differentiable function. We do it by expanding the refractivity index into the tensor product of B-splines:

$$n(h,\lambda,\varphi,t) = \sum_{l=1-m}^{l=d_t-1} \sum_{k=1-m}^{k=d_\varphi-1} \sum_{j=1-m}^{j=d_\lambda-1} \sum_{i=1-m}^{i=d_h-1} f_{ijkl}\, B_i^m(h)\, B_j^m(\lambda)\, B_k^m(\varphi)\, B_l^m(t), \tag{42}$$

where d_h, d_λ, d_φ, d_t are dimensions of variables height above the geoid, h, longitude λ, geodetic latitude φ, and time, t respectively and $B_k^m(x)$ is the B-spline of degree m with the pivotal knot k evaluated a point x. Algorithm for expanding refractivity index into B-spline basis is given in appendix. The advantage of expansion in form 42 is that partial derivatives of refractivity are easily computed using the same f_{ijkl} expansion coefficients. For instance, refractivity partial derivative with respect to height is expressed as

$$n_h(h,\lambda,\varphi,t) = \sum_{l=1-m}^{l=d_t-1} \sum_{k=1-m}^{k=d_\varphi-1} \sum_{j=1-m}^{j=d_\lambda-1} \sum_{i=1-m}^{i=d_h-1} f_{ijkl}\, B_i^{m\prime}(h)\, B_j^m(\lambda)\, B_k^m(\varphi)\, B_l^m(t), \tag{43}$$

and $B_i^{m\prime}(h)$ is computed using a recurrent relationship similar to the relationship used for computation of $B_i^m(h)$ (see Appendix B).

In order to compute partial derivatives n_η and n_ξ in equation 40m we first form a vector of partial derivatives $n_h, n_\lambda, n_\varphi$. First, than transform it to a vector of partial derivatives with respect to coordinates of the crust-fixed coordinate system n_x, n_y, n_z.

$$\begin{pmatrix} \dfrac{\partial n}{\partial x} \\ \dfrac{\partial n}{\partial y} \\ \dfrac{\partial n}{\partial z} \end{pmatrix} = \begin{pmatrix} \dfrac{\partial x}{\partial h} & \dfrac{\partial y}{\partial h} & \dfrac{\partial z}{\partial h} \\ \dfrac{\partial x}{\partial \lambda} & \dfrac{\partial y}{\partial \lambda} & \dfrac{\partial z}{\partial \lambda} \\ \dfrac{\partial x}{\partial \varphi} & \dfrac{\partial y}{\partial \varphi} & \dfrac{\partial z}{\partial \varphi} \end{pmatrix}^{-1} \begin{pmatrix} \dfrac{\partial n}{\partial h} \\ \dfrac{\partial n}{\partial \lambda} \\ \dfrac{\partial n}{\partial \varphi} \end{pmatrix} \tag{44}$$

Since Cartesian coordinates of a vector are expressed via height above the geoid h (ortho-height), longitude λ and geodetic latitude φ this way:

$$\begin{aligned} x &= \left(\frac{R_\oplus}{\sqrt{1 - e_\oplus^2 \sin^2\varphi}} + h + u(\varphi,\lambda) \right) \cos\lambda\, \cos\varphi \\ y &= \left(\frac{R_\oplus}{\sqrt{1 - e_\oplus^2 \sin^2\varphi}} + h + u(\varphi,\lambda) \right) \sin\lambda\, \cos\varphi \\ z &= \left(\frac{(1 - e_\oplus^2)\, R_\oplus}{\sqrt{1 - e_\oplus^2 \sin^2\varphi}} + h + u(\varphi,\lambda) \right) \sin\varphi \end{aligned} \tag{45}$$

where $u(\varphi, \lambda)$ is the height of the geoid above the reference ellipsoid (geoid undulation), partial derivatives in 44 can be expressed in a closed form as

$$
\begin{aligned}
\frac{\partial x}{\partial h} &= \cos\lambda\ \cos\varphi \\
\frac{\partial y}{\partial h} &= \sin\lambda\ \cos\varphi \\
\frac{\partial z}{\partial h} &= \sin\varphi \\
\frac{\partial x}{\partial \lambda} &= -\left(\frac{R_\oplus}{\sqrt{1-e_\oplus^2\sin^2\varphi}} + h + u\right)\ \sin\lambda\ \cos\varphi \\
\frac{\partial y}{\partial \lambda} &= \left(\frac{R_\oplus}{\sqrt{1-e_\oplus^2\sin^2\varphi}} + h + u\right)\ \cos\lambda\ \cos\varphi \\
\frac{\partial z}{\partial \lambda} &= 0
\end{aligned}
\tag{46}
$$

$$
\begin{aligned}
\frac{\partial x}{\partial \varphi} &= \frac{e_\oplus^2\, R_\oplus\, \sin\varphi\cos\varphi}{\sqrt{1-e_\oplus^2\sin^2\varphi}^{\,3}}\ \cos\lambda\ \cos\varphi \quad - \quad \left(\frac{R_\oplus}{\sqrt{1-e_\oplus^2\sin^2\varphi}} + h + u\right)\ \cos\lambda\ \sin\varphi \\
\frac{\partial y}{\partial \varphi} &= \frac{e_\oplus^2\, R_\oplus\, \sin\varphi\cos\varphi}{\sqrt{1-e_\oplus^2\sin^2\varphi}^{\,3}}\ \sin\lambda\ \cos\varphi \quad - \quad \left(\frac{R_\oplus}{\sqrt{1-e_\oplus^2\sin^2\varphi}} + h + u\right)\ \sin\lambda\ \sin\varphi \\
\frac{\partial z}{\partial \varphi} &= \frac{e_\oplus^2\,(1-e_\oplus^2)\, R_\oplus\, \sin\varphi\cos\varphi}{\sqrt{1-e_\oplus^2\sin^2\varphi}^{\,3}}\ \sin\varphi \quad + \quad \left(\frac{(1-e_\oplus^2)\, R_\oplus}{\sqrt{1-e_\oplus^2\sin^2\varphi}} + h + u\right)\ \cos\varphi
\end{aligned}
$$

A vector in Cartesian coordinate system with axes $\vec{\xi}$, $\vec{\eta}$, $\vec{\zeta}$ is transformed to the crust-fixed Cartesian coordinate system with axes x, y, and z via a sequence of rotations with respect to coordinate axis 1(x) and 3(z): $\widehat{\mathcal{R}}_3(-\lambda)\cdot\widehat{\mathcal{R}}_1(\varphi_{gcn})\cdot\widehat{\mathcal{R}}_1(\pi/2-A)\cdot\widehat{\mathcal{R}}_3(\pi-z)$, where z is the zenith angle, A is the azimuth of the spacecraft as it seen from the footprint, λ and φ_{gcn} are longitude and *geocentric* latitude of the point under consideration. Finally, we can write expression for partial derivatives used in 18 via partial derivatives from the expansion of the refractivity field into B-spline basis:

$$
\begin{pmatrix} \frac{\partial n}{\partial \xi} \\ \frac{\partial n}{\partial \eta} \\ \frac{\partial n}{\partial \zeta} \end{pmatrix} = \left(\widehat{\mathcal{R}}_3(-\lambda)\cdot\widehat{\mathcal{R}}_1(\varphi_{gcn})\cdot\widehat{\mathcal{R}}_1(\pi/2-A)\cdot\widehat{\mathcal{R}}_3(\pi-z)\right)^\top \begin{pmatrix} \frac{\partial x}{\partial h} & \frac{\partial y}{\partial h} & \frac{\partial z}{\partial h} \\ \frac{\partial x}{\partial \lambda} & \frac{\partial y}{\partial \lambda} & \frac{\partial z}{\partial \lambda} \\ \frac{\partial x}{\partial \varphi} & \frac{\partial y}{\partial \varphi} & \frac{\partial z}{\partial \varphi} \end{pmatrix}^{-1} \begin{pmatrix} \frac{\partial n}{\partial h} \\ \frac{\partial n}{\partial \lambda} \\ \frac{\partial n}{\partial \varphi} \end{pmatrix} \tag{47}
$$

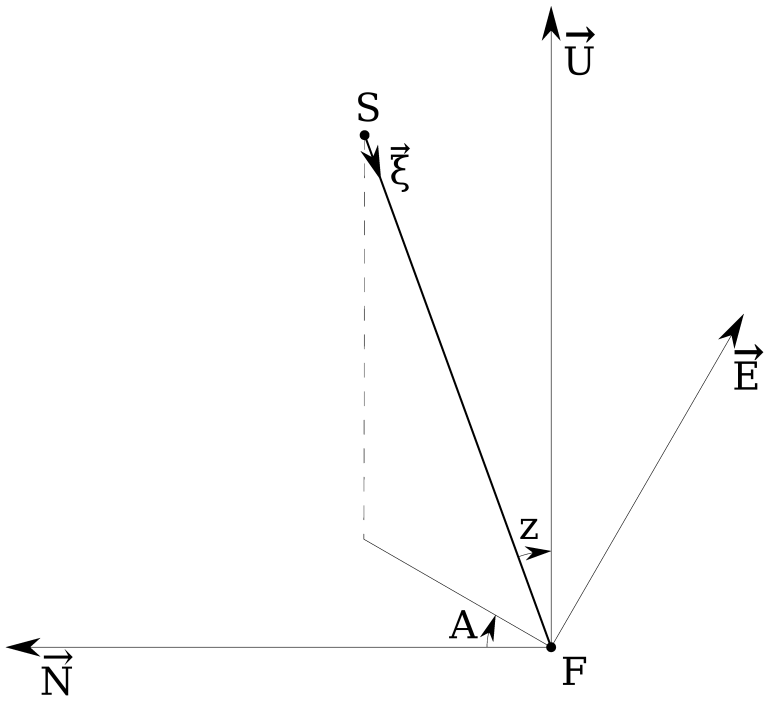

Figure 3. Geometry near the footprint F

A.3. Solving differential equations of wave propagation

Differential equation 42 has to be solved numerically, since its coefficients are empirical functions. There exist many methods for solving this kind of differential equations. However, we should keep in mind that the uncertainty in the results stems mainly due to the uncertainty in the refractivity field. Therefore, we will not gain in accuracy of our solution if we will use finer grid than the grid of the output of the numerical weather model.

First, we get positions of the footprint and the spacecraft. Position of the footprint, height above the geoid, longitude, and geodetic latitude, are provided in ICESat-1 data. We transform it to the Cartesian coordinates according to expression 45. Using zenith angle z and azimuth A, we get the unit vector of the direction to the satellite from the footprint in the local topocentric coordinate system (Up,East,North):

$$\vec{r}_u = \begin{pmatrix} \cos z \\ \sin z \, \sin A \\ \sin z \, \cos A \end{pmatrix}. \tag{48}$$

We rotate this vector to the crust-fixed coordinate system: $\vec{r}_c = \widehat{\mathcal{R}}_3(-\lambda) \cdot \widehat{\mathcal{R}}_1(\varphi_{gcn}) \, \vec{r}_u$. Then we can get coordinates of a point along the vector from the satellite to the footprint that corresponds to a given height H above the geoid ellipsoid. Neglecting Earth's elasticity and the dependence of geoid undulation on coordinates, the distance from the footprint to the level with height H along the direction to the satellite is

$$D(H) = \sqrt{(R_\oplus + H + u)^2 \cos z^2 + 2\,(R_\oplus + H + u)\,H + H^2} \;-\; (R_\oplus + H + u)\cos z \tag{49}$$

We compute the refractivity and its partial derivatives at points where the vector from the footprint intersects layers of the 4D refractivity field starting from the nominal top of the atmosphere $D(H_0)$. Cartesian coordinates of this points are $D(H_i)\,\vec{r}_c$. Therefore, coordinate ξ_i is $D(H_0) - D(H_i)$. For a given sequence of layer height of the numerical weather model grid H_i, we compute the grid ξ_k. The coordinates of this point in the Cartesian system with axes $\vec{\xi}$, $\vec{\eta}$, $\vec{\zeta}$ are $(\xi_k, 0, 0)$. Applying rotation $\widehat{\mathcal{R}}_3(-\lambda)\cdot\widehat{\mathcal{R}}_1(\varphi_{gcn})\cdot\widehat{\mathcal{R}}_1(\pi/2 - A)\cdot\widehat{\mathcal{R}}_3(\pi - z)$, we transform this vector to the crust-fixed coordinate system (x, y, z) and then compute ortho-height, geodetic latitude and longitude h, λ, φ which are arguments of the expansion for refractivity. We compute B-spline and its first derivatives for these arguments and then compute refractivity n and its partial derivatives $\dfrac{\partial n}{\partial h}, \dfrac{\partial n}{\partial \lambda}, \dfrac{\partial n}{\partial \varphi}$. With expression 44 we transform these derivatives to n_ξ, n_η.

We will seek the solution of the differential equation in a form of its expansion into B-spline basis $\eta(\xi) = \sum_{i=-2}^{i=n-1} F_i B_i^m(\xi)$. Since n in equation 42 depends on η, we have to resort to iterations. At the first iteration we assume $\eta = 0$. Differential equation in this basis is transformed to a system of n linear algebraic equations

$$\sum_{i=-2}^{i=n-1} F_i \left(B_i^{m\prime\prime}(\xi_j) + \frac{n_{\xi,j}}{1+n_j} B_i^{m\prime}(\xi_j) \right) = \frac{n_{\eta,j}}{1+n_j} \tag{50}$$

augmented with two boundary conditions

$$\begin{aligned} \sum_{i=-2}^{i=n-1} F_i\, B_i^m(\xi_0) &= 0 \\ \sum_{i=-2}^{i=n-1} F_i\, B_i^{m\prime}(\xi_0) &= 0 \end{aligned} \tag{51}$$

Here $B_i^{m\prime}$ and $B_i^{m\prime\prime}$ are first and second derivative of B-spline that are computed using recurrent relationships.

Close examination of equation 50 reveals that it has only three non-zero terms: F_{j-3}, F_{j-2}, and F_{j-1} due to the property of B-spline function and its derivatives that that is non-zero only at several knots. The first equation in 51 has only one non-zero term: with index -2, and the second equation in 51 has two non-zero terms: with index -2 and -1. Thus, equations 50–51 can be re-written as

$$\begin{array}{llllllll}
a_{-1,-2}F_{-2} & & & & & & & =0 \\
a_{0,-2}\;F_{-2} & a_{0,-1}F_{-1} & & & & & & =0 \\
a_{1,-2}\;F_{-2} & a_{1,-1}F_{-1} & a_{1,-1}F_0 & & & & & =r_1 \\
 & a_{2,-1}F_{-1} & a_{2,0}\;F_0 & a_{2,1}F_1 & & & & =r_2 \\
 & & & & \ddots & & & \\
 & & a_{n-1,n-4}F_{n-4} & a_{n-1,n-3}F_{n-3} & a_{n-1,n-2}F_{n-2} & & & =r_{n-1} \\
 & & & a_{n,n-3}\;F_{n-2} & a_{n,n-2}\;F_{n-2} & a_{n,n-1}F_{n-1} & & =r_n
\end{array} \tag{52}$$

That means that the $(n+2)\times(n+2)$ matrix in 52 is banded diagonal:

$$\begin{pmatrix}
\mathbf{x} & & & & & & & & \\
x & \mathbf{x} & & & & & & & \\
x & x & \mathbf{x} & & & & & & \\
 & x & x & \mathbf{x} & & & & & \\
 & & x & x & \mathbf{x} & & & & \\
 & & & x & x & \mathbf{x} & & & \\
 & & & & x & x & \mathbf{x} & & \\
 & & & & & x & x & \mathbf{x} & \\
 & & & & & & x & x & \mathbf{x}
\end{pmatrix} \tag{53}$$

Bold fonts shows the main diagonal. Solution of this system of equations is trivial:

$$\begin{aligned}
F_{-2} &= 0 \\
F_{-1} &= 0 \\
F_j &= \frac{\frac{n_{\eta,j+1}}{1+n_{j+1}} - F_{j-2}\left(B^m_{j-2}{}''(\xi_{j+1}) + \frac{n_{\xi,j+1}}{1+n_{j+1}} B^m_{j-2}{}''(\xi_{j+1})\right) - F_{j-1}\left(B^m_{j-1}{}''(\xi_{j+1}) + \frac{n_{\xi,j+1}}{1+n_{j+1}} B^m_{j-1}{}'(\xi_{j+1})\right)}{B^{m\,\prime\prime}_j(\xi_{j+1}) + \frac{n_{\eta,j+1}}{1+n_{j+1}} B^{m\,\prime}_j(\xi_{j+1})}
\end{aligned} \tag{54}$$

Using the coefficients F_j, we compute $\eta(\xi_n)$ and then the footprint offset with the use of expression 41.

A.4. Results

Computation of the footprint offsets for ICESat-1 satellite showed that the footprint is shifted at 3–5 cm, which is negligible for processing ICESat observations. Figure shows an example of the bending angle as a function of ξ — the distance from the top of the atmosphere along the trajectory of photons for a hypothetical shot at distance angle 5°. In this example the bending angle at the surface as $4''.705$. According to *Green* (1985), the refraction angle for the standard atmosphere at low zenith angles can be approximated as $60''.29\tan z - 0''.0669\tan z^3$, which gives $5''.27$ – within 10% of the above mentioned approximation. In this example the footprint offset is 0.205 m.

B. Algorithm for the multidimensional B-spline interpolation

Let us consider a N-dimensional function at R^N which is defined on a equi-distant N-dimensional grid and which has $m-1$th derivatives at each point. Dimension of the function is $d_1, d_2, \ldots d_N$. Then at can be expanded over the N-dimension B-spline basis:

$$F(x_{i_1}, \ldots x_{i_N}) = \sum_{i_1=1-m}^{d_1-1} \ldots \sum_{i_N=1-m}^{d_N-1} f_{i_1,\,\ldots i_N} \prod_{\alpha=1}^{\alpha=N} B^m_{i_\alpha}(x_{i_\alpha}) \tag{55}$$

Here α is dimension and N is rank. The coefficients $f_{i_1,\,\ldots i_N}$ of the expansion can be found by the following procedure.

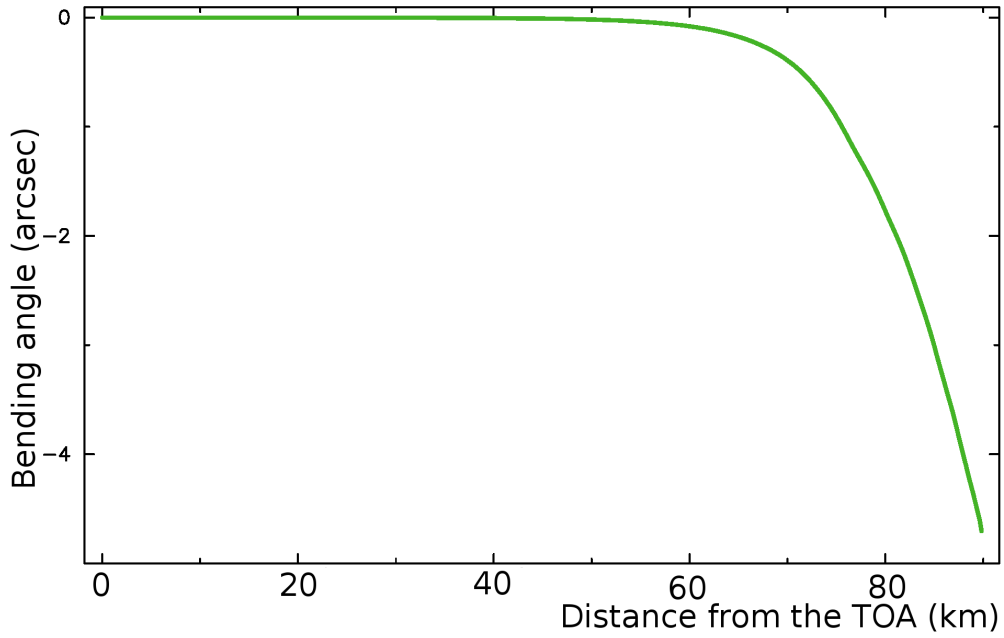

Figure 4. Bending angle as a function of the distance from the top of the atmosphere (TOP) for Greenbelt, MD on 2006.06.23-00:48:43.562. The zenith angle of the satellite is 5°.

First, take dimension β and extend the space of knots over dimension β by adding 1-m knots before the not with index 1. Second, augment the number of knots by 1 for cyclic dimensions, such as longitude, and assign the coordinate of that extra knot to the coordinate of the 1st knot: $x_{n_\alpha+1} = x_1$.

Then extract $k = \prod_{\alpha=1,\alpha\neq\beta}^{\alpha=N} d_\alpha$ one-dimensional sections of function F by fixing all variables at all dimensions, except dimension β. For each such section compute coefficients of B-spline expansion $\bar{f}_i$ (bar denotes a one-dimensional section). For each index of the β dimension $1, 2, \ldots n_1$ we have a linear equations

$$\bar{F}(x_k) = \sum_{i=1-m}^{n_\beta-1} \bar{f}_i B_i^m(x_k) \tag{56}$$

For $m > 1$ we should augment this system of n equations with $n-1+m$ unknowns with $m-1$ additional conditions. For $m = 3$ we usually require B-spline to have first derivatives at the ends of interpolating range that coincide with with first differences:

$$\begin{aligned} \frac{\bar{F}(x_2) - \bar{F}(x_1)}{x_2 - x_1} &= \sum_{i=1-m}^{n_\beta-1} \bar{f}_i B_i^{m'}(x_1) \\ \frac{\bar{F}(x_n) - \bar{F}(x_{n_\beta-1})}{x_{n_\beta} - x_{n_\beta-1}} &= \sum_{i=1-m}^{n_\beta-1} \bar{f}_i B_i^{m'}(x_1). \end{aligned} \tag{57}$$

Since a B-spline is non-zero at $[i-m+1, i]$ knots only, the system of linear algebraic equations 56–57 is banded, with the width of the diagonal band $m+1$. The system is solved by decomposition of the left-hand side matrix. It sufficient to do it only once. The decomposed matrix of the linear system of equations is used for solving interpolation equations with $k = \prod_{\alpha=1,\alpha\neq\beta}^{\alpha=N} d_\alpha$ right-hand sides. After solving each equation with a given right hand-side, the result replaces elements $\bar{f}_i$ of the function F at the extended grid — the former right-hand-side. There exist very efficient parallel algorithms for this procedure.

This operation is repeated for the 2nd, 3rd till the N-th dimension. Operation for each next dimension is done with results of processing the previous dimensions. The computational complexity of this algorithm is proportional to the product of dimensions: $O(\prod_{\alpha=1}^{\alpha=N} d_\alpha)$, i.e. the number of grid elements.

As the input N-dimensional array is put into the extended grid $[1-m : d_1, 1-m : d_2, \ldots 1-m : d_N]$. At the beginning elements with indices 1-m:0 are undefined, elements with indices $1 : d_\alpha$ are filled with the values of the function to be expanded. At the end the N-dimensional array is filled with the coefficients of B-spline expansion in elements with indices $[1-m, d_\alpha - 1]$. Elements with indices d_α keep original values of the input function.

Below is a detailed explanation

B.1. Computation of B-spline and its derivative

Definition 1. Let m be a natural number. *Truncated power function* is:

$$x_+^m = \begin{cases} x^m, & \text{if} \quad x > 0 \\ 0, & \text{if} \quad x \leq 0 \end{cases}$$

Definition 2. Define function $f(t)$ on the set of knots $(t_1, t_2, \ldots, t_k)$ such that $t_1 < t_2 < \ldots t_k$. Then the relationship

$$f[t_j, t_{j+1}] = \frac{f(t_{j+1}) - f(t_j)}{t_{j+1} - t_j}, \quad j \in 1 : k-1$$

is called *divided difference of the first order.*

Definition 3. *Divided difference of the m-th order* is determined with use of the following relationship:

$$f[t_j, t_{j+1}, \ldots, t_{j+m}] = \frac{f[t_{j+1}, t_{j+2}, \ldots, t_{j+m}] - f[t_j, t_{j+1}, \ldots, t_{j+m-1}]}{t_{j+m} - t_j}$$

Definition 4. *Polynomial spline* of degree m on the sequence of knots $(x_1, x_2, \ldots, x_n)$ such that $x_1 \leq x_2 \leq \ldots x_k$ is a function

$$S(x) = \sum_{k=0}^{m} a_k x_k + \sum_{j=1}^{m} b_j (x - x_j)_+^m$$

The knots $(x_1, x_2, \ldots, x_k)$ are called *support of the spline.*

Definition 5. *B-spline of the m-th degree* with the pivot knot i on the knots sequence $(t_k, t_{k+1}, \ldots, t_{k+m+1})$ such that $t_k < t_{k+1} < \ldots < t_{k+m+1}$ is the function

$$B_k^m(x) = \sum_{j=i}^{k+m+1} (t-x)_+^m [t_j, t_{j+1}, \ldots, t_{j+m}]$$

B.1.1. Recurrent relationship for B-spline

B-spline of the 0-th degree with the pivot knot i degree on the on the knots sequence $(t_k, t_{k+1}, \ldots, t_n)$ such that $t_k < t_{k+1} < \ldots < t_k$ is determined the following way:

$$B_k^0(x) = \begin{cases} 1, & \text{if} \quad x \in [x_k, x_{k+1}) \\ 0, & \text{otherwise} \end{cases} \tag{58}$$

B-spline of the m th degree with the pivot knot i on the on the knots sequence $(t_k, t_{k+1}, \ldots, t_n)$ is expressed via B-splines of the $m-1$ th degree this way:

$$B_k^m(x) = \frac{x - x_k}{x_{k+m} - x_k} B_k^{m-1}(x) \quad + \quad \frac{x - x_{k+m+1}}{x_{k+1} - x_{k+m+1}} B_{k+1}^{m-1}(x) \tag{59}$$

B.1.2. Recurrent relationship for the first derivative of a B-spline

The first derivative of the B-spline of the m-th degree with the pivot knot i is expressed via B-splines of the $m-1$ th degree this way:

$$B_k^{m\prime}(x) = \frac{m}{x_{k+m} - x_k} B_k^{m-1}(x) \quad + \quad \frac{m}{x_{k+1} - x_{k+m+1}} B_{k+1}^{m-1}(x) \tag{60}$$

B.1.3. Recurrent relationship for the second derivative of a B-spline

The first derivative of the B-spline of the m-th degree with the pivot knot i is expressed via B-splines of the $m-1$ th degree this way:

$$B_k^{m\prime\prime}(x) = \frac{m}{x_{k+m} - x_k} B_k^{m-1\prime}(x) \quad + \quad \frac{m}{x_{k+1} - x_{k+m+1}} B_{k+1}^{m-1\prime}(x) \tag{61}$$

B.1.4. Expression for integral of the B-spline

Integral of the B-spline of the m-th degree with the pivot knot i is expressed via a sum of the B-splines of the $m+1$ th degree this way:

$$I_k^m(x) = \int_{-\infty}^{x} B_k^m(x)\, dx \quad = \frac{x_{k+m+1} - x_k}{m+1} \sum_{p=k}^{k+m} B_p^{m+1}(x) \tag{62}$$

$$J_k^m(x) = \int_{x}^{+\infty} B_k^m(x)\, dx \quad = \frac{x_{k+m+1} - x_k}{m+1} \left(1 - \sum_{p=k}^{k+m} B_p^{m+1}(x)\right) \tag{63}$$

$$K_k^m(x) = \int_{x_{k-1}}^{x_k} B_k^m(x)\,dx \;= \frac{x_{k+m+1} - x_k}{m+1} \sum_{p=k}^{k+m} \left(B_p^{m+1}(x_k) - B_p^{m+1}(x_{k-1}) \right) \tag{64}$$

B.1.5. Computation of integral from a B-spline transform

Let us have function $x_k(t_k)$ defined on a non-decreasing sequence of knots $t_1, t_2, \dots t_{n-1}, t_n$ such that $\forall k < n-1 \;\; t_k \le t_{k+1}$. Let it have 1D B-spline transform

$$x(t_i) = \sum_{k=1-m}^{k=n-1} X_k\, B_k^m(t_i). \tag{65}$$

Integral from the 1st to the kth knot ($k > 1$) can be easily evaluated

$$\int_{t_1}^{t_k} x(t)\,dt = \sum_{p=k-m-1}^{k-1} X_p K_p^m(x) \tag{66}$$

B.2. 1D B-spline transform

Let us have function $x_k(t_k)$ defined on a non-decreasing sequence of knots $t_1, t_2, \dots t_{n-1}, t_n$ such that $\forall k < n-1 \;\; t_k \le t_{k+1}$. Then we define the so-called B-spline transform of degree m $\mathcal{B}^m(x)$ that transforms $x \to X$ at the extended sequence of knots $t_{1-m}, t_{2-m}, \dots t_1, t_2, t_{n-1}, t_n$ such that $\forall k \le 1 \;\; t_k = t_1$. X_n is undefined. Thus, $X(t)$ has $n + m - 1$ values.

The transform is defined in such a way that

$$x(t_i) = \sum_{k=1-m}^{k=n-1} X_k\, B_k^m(t_i). \tag{67}$$

Since $B_k^m(t_i)$ is zero outside interval (t_{i-m}, t_{i+1}), the sum in 67 is reduced to

$$x_i(t_i) = \sum_{k=i-m}^{k=i} X_k\, B_k^m(t_i). \tag{68}$$

Since the number of unknowns is by $m-1$ greater than the number of equations, we need to add $m-1$ conditions. For $m = 3$ we add two conditions at the ends of the interval constraining the first derivative. We require that the first derivative $\dot{x}(t_1)$ be equal to the first difference

$$\dot{x}(t_1) = \sum_{k=1-m}^{k=1} X_k\, B_k'^m(t_1) = \frac{x(t_2) - x(t_1)}{t_2 - t_1} \tag{69}$$

and analogous condition at the other end of the interval:

$$\dot{x}(t_n) = \sum_{k=n-1-m}^{k=n-1} X_k\, B_k'^m(t_n) = \frac{x(t_n) - x(t_{n-1})}{t_n - t_{n-1}}. \tag{70}$$

Here $B_k'^m(t)$ is the first derivative of B-spline.

Equations 68–70 forms the band-diagonal matrix. In a case of m=3, it has this form:

$$\begin{pmatrix} x & & & & & & & & \\ x & x & & & & & & & \\ & x & x & x & & & & & \\ & & x & x & x & & & & \\ & & & x & x & x & & & \\ & & & & x & x & x & & \\ & & & & & x & x & x & \\ & & & & & & x & x & x \\ & & & & & & & x & x \\ & & & & & & & & x \end{pmatrix} \tag{71}$$

This system of equations is efficiently solved using with BLAS routines SGBTRF (matrix LU decomposition) and SGBTRS (solving linear algebraic equations of the decomposed matrix). These routines require the matrix to be stored in the so-called band-storage format.

The one-dimensional B-spline transform of degree m of array of values x and argument t of dimension n is performed in 4 steps:

Step 1: Fill the matrix of equations in banded format. Reserve (m+1)*(n+m-1) elements of memory for matrix M. Although, M is a matrix, the index expression below unrolls dimensions, i.e. matrix M is treated as a one-dimensional array.

Fill it in two loops:

for $i = 1-m, 1-m+1, \ldots, n-1$ do
  for $k = \max(i+1, 0) \ldots, \min(i+m, n+1)$ do
    if $k == 0$ then $M[i-1+m(j+2)] = B_i'^m(t_1 + \varepsilon)$
    else if $k == n+1$ then $M[i-1+m(j+2)] = B_i'^m(t_n - \varepsilon)$
    else if $k == n$ then $M[i-1+m(j+2)] = B_i^m(t_n - \varepsilon)$
    else then $M[i-1+m(j+2)] = B_i^m(t_i)$

where ε is a small number taken in order to avoid errors in comparison of two real numbers due to rounding.

Step 2: Decompose matrix band M with dimension n+m-1, sub-diagonal dimension 1, and super-diagonal 1 into a L*U product using BLAS routine SGBTRF/DGBTRF:

SGBTRF (n+m-1, n+m-1, 1, 1, M, m+1, I, INFO)

where I is an integer array of dimension n+m-1 undefined at input

Step 3: Build the right hand-side array R of dimension [n+m-1]

R[1] = (x[2] - x[1])/(t[2] - t[1]) # initial condition

for $i = 2, 3, \ldots, n$ do
  $R[i] = x[i-1]$

R[n+m-1] = (x[n] - x[n-1])/(t[n] - t[n-1]) # initial condition

Step 4: Solve for B-spline transform by solving a system of linear equations with decouples banded matrix of dimension $(n+m-1) \times (n+m-1)$ with sub-diagonal dimension 1, and super-diagonal 1 and with right-hand side array R.

SGBTRS ('T', n+m-1, 1, 1, 1, M, m+1, I, R, n+m, INFO)

The result is written into place of R:
X := R

end of algorithm.

B.3. 2D B-spline transform

Let us have a function $x_{ij}(t_{ij})$ defined on a *regular* 2D grid o of dimension $n_1 \times n_2$. A 2D grid t_{ij} is called regular if $\forall j, i_1, i_2, j_1, j_2$ the difference $t_{i_2,j} - t_{i_1,j} = \text{const}$ and $t_{i,j_2} - t_{i,j_1} = \text{const}$, i.e. the interval over one dimension does not depend on another dimension. Let grid interval over each dimension d is a non-decreasing sequence with indices $1, 2, \ldots n_{d_1}$. Then the 2D B-spline transform is defined as

$$x_{ij}(t_{ij}) = \sum_{k=1-m}^{n_1-1} \sum_{l=1-m}^{n_2-1} X_{kl}\, B_k^m(t_i)\, B_l^m(t_j). \tag{72}$$

Basically, the 2D B-spline transform is done by performing n_2 times the 1D B-spine transform over columns and them by performing n_1 times the 1D B-spine transform over rows over the result of the previous procedure. Columns of original function x_{ia} are replaced with their 1D B-spline transform $\mathcal{B}(x_{ia}) = X^{1d}_{ia}$. Then, the result is transformed over rows: $X^{2d}_{bj} = \mathcal{B}(X^{1d}_{bj})$

The 2D array X_{ij} has dimension $[1-m : n_1 - 1, 1-m : n_2 - 1]$. In order to simplify implementation, we size input array x as [1-m:n,1-m:n] with elements with indices less than 1 undefined at input. Results of transform are put in the same place with elements with index equal to n undefined (and unused). The algorithm is the following:

Step 1: Extract 1D array $T_1(a) = t[a, 1]$ for $a = 1, 2, \ldots, n_1$.

Step 2: Perform Steps 1, and 2 of the 1D algorithm over T_1 array. As a result, we get decomposed matrix M_1.

Step 3:

for $i = 1, 2, \ldots, n_2$ parallel do
Step 3a : Create right-hand side:
$R[1] = (x[2, 1] - x[1, 1])/(t[2, 1] - t[1, 1])$
for $j = 2, 3, \ldots, n_1$ do
$R[j] = x[j - 1, i]$
$R[n + m - 1] = (x[n_1, i] - x[n_1 - 1, i])/(t[n_1, 1] - t[n_1 - 1, 1])$
Step 3b : Solve for 1D B-spline transform for the i-th column:

SGBTRS ('T', n_1+m-1, 1, 1, 1, M_1, m+1, I, R, n_1+m, INFO)
Step 3c : Put result in place of original data

for $j = 1 - m, 2 - m, \ldots, n_1$ do
$x[j, i] := R[j + m]$

Step 4: Extract 1D array $T_2(b) = t[1, b]$ for $b = 1, 2, \ldots, n_2$.

Step 5: Perform Steps 1, and 2 of the 1D algorithm over T_2 array. As a result, we get decomposed matrix M_2.

Step 6:

for $i = 1, 2, \ldots, n_1$ parallel do
Step 6a : Create right-hand side:
$R[1] = (x[1,2] - x[1,1])/(t[1,2] - t[1,1])$
for $j = 2, 3, \ldots, n_1$ do
$R[j] = x[i, j-1]$
$R[n+m-1] = (x[i,n_2] - x[i,n_2-1])/(t[1,n_2] - t[1,n_2-1])$
Step 6b : Solve for 1D B-spline transform for the i-th row:

SGBTRS ('T', n_2+m-1, 1, 1, 1, M_2, m+1, I, R, n_2+m, INFO)
Step 6c : Put result in place of original data

for $j = 1-m, 2-m, \ldots, n_2 - 1$ do
$x[i,j] := R[j+m]$

End of algorithm.

Comment: Longitude-latitude $n \times m$ data are usually represented at an equal-angular grid that runs from 0 to $2\pi - \Delta\lambda$ and from $-\pi/2$ to π, where $\Delta\lambda = 2\pi/n$ is the grid step over longitude. The grid defined such a way covers longitude interval [0, $2\pi - \Delta\lambda$]. Therefore, a longitude range $(2\pi - \Delta\lambda, 2\pi]$ falls in a blind zone. In order to overcome this problem, the grid is extended to $n+1 \times m$ by adding a row of longitude 2π such that $\forall m: \; G(n+1, m) = G(1, m)$. When we preform 2D, 3D, or 4D B-spline transform that involves one of dimensions along longitude, we always extend the grid on the fly. Memory for the extended grid should be sized accordingly.

B.4. 3D B-spline transform

Algorithm of 3D B-spline transform is organized in a similar way as the 2D transform. The same procedure is performed three times: over the 1st, 2nd, and 3rd dimension. Each time a 2D section complimentary to a given dimension is extracted, 1D transform is performed and results are put in place of the original data.

Step 1: Extract 1D array $T_1(a) = t[a, 1, 1]$ $\forall a = 1, 2, \ldots, n_1$.

Step 2: Perform Steps 1, and 2 of the 1D algorithm over T_1 array. As a result, we get decomposed matrix M_1.

Step 3:

for $i = 1, 2, \ldots, n_3$ parallel do
  Step 3a : Create right-hand side:
  for $j = 1, 2, \ldots, n_2$ do
    $R[1, j] = (x[2, j, i] - x[1, j, i])/(T_1[2] - T_1[1])$
    for $k = 2, 3, \ldots, n_1$ do
      $R[k, j] = x[k - 1, j, i]$
    $R[n + m - 1, j] = (x[n_1, j, i] - x[n_1 - 1, j, i])/(T_1[n_1] - T_1[n_1 - 1])$
  Step 3b : Solve for 1D B-spline transform for the i-th column
    with multiple right-hand sides:
  SGBTRS ('T', n_1+m-1, 1, 1, n_2, M_1, m+1, I, R, n_1+m, INFO)
  Step 3c : Put result in place of original data

  for $j = 1 - m, 2 - m, \ldots, n_2$ do
    for $k = 1 - m, 2 - m, \ldots, n_1 - 1$ do
      $x[k, j, i] := R[k + m, j]$

Step 4: Extract 1D array $T_2(b) = t[1, b, 1]$ for $b = 1, 2, \ldots, n_2$.

Step 5: Perform Steps 1, and 2 of the 1D algorithm over T_2 array. As a result, we get decomposed matrix M.

Step 6:
  for $i = 1, 2, \ldots, n_3$ parallel do
    Step 6a : Create right-hand side:
    for $j = 1 - m, 2 - m, \ldots, n_1 - 1$ do
      $R[1, j] = (x[j, 2, i] - x[j, 1, i])/(T_2[2] - T_2[1])$
      for $k = 2, 3, \ldots, n_2$ do
        $R[k, j] = x[j, k - 1, i]$
      $R[n + m - 1, j] = (x[j, n_2, i] - x[j, n_2 - 1, i])/(T_2[n_2] - T_2[n_2 - 1])$
    Step 6b : Solve for 1D B-spline transform for the i-th dimension
      with multiple right-hand sides:
    SGBTRS ('T', n_2+m-1, 1, 1, n_1, M_1, m+1, I, R, n_2+m, INFO)
    Step 6c : Put result in place of original data

    for $j = 1 - m, 2 - m, \ldots, n_1 - 1$ do
      for $k = 1 - m, 2 - m, \ldots, n_2 - 1$ do
        $x[j, k, i] := R[k + m, j]$

Step 7: Extract 1D array $T_3(c) = t[1, 1, b]$ for $b = 1, 2, \ldots, n_3$.

Step 8: Perform Steps 1, and 2 of the 1D algorithm over T_3 array. As a result, we get decomposed matrix M.

Step 9:

for $i = 1-m, 2-m, \ldots, n_2$ parallel do
Step 9a : Create right-hand side:
for $j = 1-m, 2-m, \ldots, n_1 - 1$ do
$R[1,k] = (x[k,i,2] - x[k,i,1])/(T_3[2] - T_3[1])$
for $k = 2, 3, \ldots, n_3$ do
$R[k,j] = x[j,i,k-1]$
$R[n+m-1,j] = (x[j,i,n_2] - x[j,i,n_2-1)/(T_3[n_3] - T_3[n_3-1])$
Step 9b : Solve for 1D B-spline transform for the i-th dimension
with multiple right-hand sides:
SGBTRS ('T', n_3+m-1, 1, 1, n_1, M_1, m+1, I, R, n_3+m, INFO)
Step 9c : Put result in place of original data

for $j = 1-m, 2-m, \ldots, n_1 - 1$ do
for $k = 1-m, 2-m, \ldots, n_3 - 1$ do
$x[j,i,k] := R[k+m,j]$

B.5. 4D B-spline transform

Algorithm of a 4D B-spline transform is organized similar to the 3D transform algorithm. The same procedure is performed four times: over the 1st, 2nd, 3rd, and 4th dimension. Each time a 3D section complimentary to a given dimension is extracted, 1D transform is performed and results are put in place of the original data.

Step 1: Extract 1D array $T_1(a) = t[a,1,1,1]$ for $a = 1, 2, \ldots, n_1$.

Step 2: Perform Steps 1, and 2 of the 1D algorithm over T_1 array. As a result, we get decomposed matrix M_1.

Step 3:

for $i = 1, 2, \ldots, n_4$ parallel do
Step 3a : Create right-hand side:
for $j = 1, 2, \ldots, n_3$ do
for $k = 1, 2, \ldots, n_2$ do
$R[1,k,j] = (x[2,k,j,i] - x[1,k,j,i])/(T_1[2] - T_1[1])$
for $l = 2, 3, \ldots, n_1$ do
$R[l,k,j] = x[l-1,k,j,i]$
$R[n+m-1,k,j] = (x[n_1,k,j,i] - x[n_1-1,k,j,i])/$
$(T_1[n_1] - T_1[n_1-1])$
Step 3b : Solve for 1D B-spline transform for the i-th column
with multiple right-hand sides:
SGBTRS ('T', n_1+m-1, 1, 1, $n_2 \cdot n_3$, M_1, m+1, I, R, n_1+m, INFO)
Step 3c : Put result in place of original data

for $j = 1, 2, \ldots, n_3$ do
for $k = 1, 2, \ldots, n_2 - 1$ do
for $l = 1-m, 2-m, \ldots, n_1 - 1$ do
$x[l,k,j,i] := R[l+m,k,j]$

Step 4: Extract 1D array $T_2(b) = t[1,b,1,1]$ for $b = 1, 2, \ldots, n_2$.

Step 5: Perform Steps 1, and 2 of the 1D algorithm over T_2 array. As a result, we get decomposed matrix M_1.

Step 6:
for $i = 1, 2, \ldots, n_4$ parallel do
Step 6a : Create right-hand side:
for $j = 1, 2, \ldots, n_3$ do
for $k = 1 - m, 2 - m, \ldots, n_1$ do
$R[1, k, j] = (x[k, 2, j, i] - x[k, 1, j, i])/(T_2[2] - T_2[1])$
for $l = 2, 3, \ldots, n_2$ do
$R[l, k, j] = x[k, l - 1, j, i]$
$R[n + m - 1, k, j] = (x[k, n_2, j, i] - x[k, n_2 - 1, j, i])/$
$(T_2[n_2] - T_2[n_2 - 1])$
Step 6b : Solve for 1D B-spline transform for the i-th dimension
with multiple right-hand sides:
SGBTRS ('T', n_2+m-1, 1, 1, $(n_1 + m) \cdot n_3$, M_2, m+1, I, R, n_2+m, INFO)
Step 6c : Put result in place of original data

for $j = 1, 2, \ldots, n_3$ do
for $k = 1 - m, 2 - m, \ldots, n_1 - 1$ do
for $l = 1 - m, 2 - m, \ldots, n_2 - 1$ do
$x[k, l, j, i] := R[l + m, k, j]$

Step 7: Extract 1D array $T_3(c) = t[1, 1, c, 1]$ for $c = 1, 2, \ldots, n_3$.

Step 8: Perform Steps 1, and 2 of the 1D algorithm over T_3 array. As a result, we get decomposed matrix M_2.

Step 9:
for $i = 1, 2, \ldots, n_4$ parallel do
Step 9a : Create right-hand side:
for $j = 1 - m, 2 - n, \ldots, n_2$ do
for $k = 1 - m, 2 - m, \ldots, n_1$ do
$R[1, k, j] = (x[k, j, 2, i] - x[k, j, 1, i])/(T_3[2] - T_3[1])$
for $l = 2, 3, \ldots, n_3$ do
$R[l, k, j] = x[k, j, l - 1, i]$
$R[n + m - 1, k, j] = (x[k, j, n_3, i] - x[k, j, n_3 - 1, i])/$
$(T_3[n_3] - T_3[n_3 - 1])$
Step 9b : Solve for 1D B-spline transform for the i-th dimension
with multiple right-hand sides:
SGBTRS ('T', n_3+m-1, 1, 1, $(n_1 + m) \cdot (n_3 + m)$, M_3, m+1, I, R, n_3+m, INFO)
Step 9c : Put result in place of original data

for $j = 1 - m, 2 - m, \ldots, n_2$ do
for $k = 1 - m, 2 - m, \ldots, n_1 - 1$ do
for $l = 1 - m, 2 - m, \ldots, n_3 - 1$ do
$x[k, j, l, i] := R[l + m, k, j]$

Step 10: Extract 1D array $T_4(d) = t[1, 1, 1, c]$ for $d = 1, 2, \ldots, n_4$.

Step 11: Perform Steps 1, and 2 of the 1D algorithm over T_4 array. As a result, we get decomposed matrix M_3.

Step 12:
for $i = 1-m, 2-m, \ldots, n_3$ parallel do
Step 12a : Create right-hand side:
for $j = 1-m, 2-n, \ldots, n_2$ do
for $k = 1-m, 2-m, \ldots, n_1$ do
$R[1,k,j] = (x[k,j,i,2] - x[k,j,i,1])/(T_4[2] - T_4[1])$
for $l = 2, 3, \ldots, n_4$ do
$R[l,k,j] = x[k,j,i,l-1]$
$R[n+m-1,k,j] = (x[k,j,i,n_4] - x[k,j,i,n_4-1])/$
$(T_4[n_4] - T_4[n_4-1])$
Step 12b : Solve for 1D B-spline transform for the i-th dimension
with multiple right-hand sides:
SGBTRS ('T', n_4+m-1, 1, 1, $(n_1+m)\cdot(n_2+m)$, M_4, m+1, I, R, n_4+m, INFO)
Step 12c : Put result in place of original data

for $j = 1-m, 2-m, \ldots, n_2$ do
for $k = 1-m, 2-m, \ldots, n_1-1$ do
for $l = 1-m, 2-m, \ldots, n_4-1$ do
$x[k,j,i,l] := R[l+m,k,j]$

B.6. Computation of a function and its derivative using its B-spline transform.

Let us consider a general case of a 4D function. Computation of function is done in two steps. First step is to find so-called pivot indices for every dimension for a given value of argument A pivot index i_p for argument t_a is such that $t[i_p] = max(t_i \le t_a)$, i.e. the maximum grid element index among those not exceeding t_a. Then

$$x(t_{ijkl}) = \sum_{i=i_p-m}^{i=i_p} \sum_{j=j_p-m}^{j=j_p} \sum_{j=k_p-m}^{j=k_p} \sum_{j=l_p-m}^{j=l_p} X_{ijkl}\, B_i^m(t_i)\, B_j^m(t_j)\, B_k^m(t_k)\, B_l^m(t_l) \tag{73}$$