atmosphere

MDPI

Article

Spherical Grid Creation and Modeling Using the Galerkin Compiler GC_Sphere

Jürgen Steppeler

Climate Service Center Germany (GERICS), Helmholtz-Zentrum Hereon, Fischertwiete 1, 20095 Hamburg, Germany; jsteppeler@t-online.de

Abstract: The construction of spherical grids is, to a large extent, a question of organized programming. Such grids come in the form of rhomboidal/triangular grids and hexagonal grids. We are here mainly interested in Local-Galerkin high-order schemes and consider the classical fourth-order o4 method for comparison. High-order Local-Galerkin schemes imply sparse grids in a natural way, with an expected saving of computer runtime. Sparse grids on the sphere are described for rhomboidal and hexagonal cells. They are obtained by not using some of the full grid points. Technical problems and grid organization will be discussed with the purpose of reaching fully realistic applications. We present the description of a programming concept allowing people, using different programming styles at different locations, to work together. The concept of geometric files is introduced. Such geometric files can be offered for downloading and are supposed to allow Local-Galerkin methods to be introduced into an existing model with little effort. When the geometric files are known, the solution on a spherical grid is equivalent to the limited-area Galerkin solutions on the (irregular) plane grids on the patches. The proposed grids can be used with spectral elements (SE) and the Local-Galerkin methods o2o3 and o3o3. The latter offer an increased numerical efficiency which, in a toy model test, resulted in a ten-times-reduced computer run time.

Keywords: spherical grids; Galerkin; software engineering

Citation: Steppeler, J. Spherical Grid Creation and Modeling Using the Galerkin Compiler GC_Sphere. *Atmosphere* **2023**, *14*, 966. https://doi.org/10.3390/atmos14060966

Academic Editors: Miodrag Rancic and Ivana Tosic

Received: 18 April 2023
Revised: 20 May 2023
Accepted: 25 May 2023
Published: 31 May 2023

1. Introduction

Spherical grids are based on Platonic or semi-Platonic solids, and are generated by dividing the patches on the surface of the solid into cells. In a second step, these grids are projected onto the sphere (see [1] hereafter referred to as St-Li) for a review of spherical grids). Early attempts to use grids based on the icosahedron ([2–4], failed due to two reasons which are described in St-Li. [5,6]) argued that such grids could be used when coordinate systems are not the same for all patches of the solid. The scheme [5] was introduced into the DWD (Deutscher Wetterdienst) global model during several visits around 2000. A recent description of this model is given by [5,7] used the cloud-grid method. This method constructs a finite difference stencil using the nearest neighbors of the target point. The method is described in St-Li and, under efficiency considerations, is limited to second order approximations. It had homogeneous approximation order 2 and did not need grid smoothing. Later versions of the ICON model were not of homogeneous approximation order 2 and needed grid smoothing. Grid smoothing is a procedure which makes the grid cells as homogeneous as possible. A homogeneous third-order version of the icosahedral rhomboidal grid was given by [8–14]. Let us call a realistic model, into which a spherical grid discretization is introduced, the target model. The grid construction and programming suggestions made in this paper have the purpose of easily allowing the transfer of a grid system tested in a toy model to different realistic modeling systems.

Spherical grids come in a number of variations which are different in efficiency. All spherical grids divide the sphere into patches (St-Li). For example, the cubed sphere has 6 patches, the icosahedron has 10 and the T36 solid has 18. T36 is a semi-Platonic solid

having 12 equal rhomboids around the poles and a further 6 rhomboids at the equator (see St-Li). It is a triangulated form of the hexagonal prism. Each patch is subdivided into grid cells. In this paper, we consider conforming grids which, in two dimensions, are indexed by *ii,jj*. When the patches contain more than one grid cell, the grid is necessarily irregular, meaning that the grid lengths are not equal. A larger number of patches leads to smaller patches which will result in more uniform grids.

The different grids based on the cubed sphere, icosahedron etc. were evaluated by their developers based on the optical impression of the plots of the grids. Unfortunately, no systematic evaluation of grid properties, beyond the optical evaluation, exist in the literature. The grid statistics of interest would be maximum/minimum grid lengths, angle of coordinate lines and grid isotropy. The latter is the change in effective grid-length seen in different directions, which for a quadratic plane grid has the relation of $1 : \sqrt{2}$. Using the optical impression of the cubed sphere, icosahedron and T36, from plots given in St-Li, it appears that the patches become smaller and the grids more homogeneous. For the angle of co-ordinate lines, most grids have a range of 60 to 120 degrees. For grid isotropy, it seems that hexagonal grids have an advantage over rhomboidal grids. A paper making this conclusion more quantitative would certainly be of great value. Using the plotted grids for cubed sphere, icosahedron and the T36 solid, as given in St-Li, it can be seen that the grid for T36 is the most homogeneous, which provides a justification for the use of semi-Platonic compared to Platonic solids.

This means that the difference of the smallest and the largest cell dimensions is small compared to the grid length. When hanging nodes are avoided, all grids based on solids mentioned above have angles between grid lines ranging from 60 to 120 degrees. When discussing grids, another efficiency consideration depends on the lines of grid points being on great circles. When this property is obtained, some of the numerical methods described in St-Li are more efficient than with grid points not lining up on a great circle.

Finite differences and Local-Galerkin (L-Galerkin) on hexagonal cells have a potential to be more efficient than with rhomboidal cells. For an example of a model based on hexagons see 59. Hexagonal grids may also be sparse (see [12] for a simple test with a sparse hexagonal grid on a plane grid). The potential for computer time saving due to sparseness is larger with hexagons than with rhomboids. In this paper, the sparse hexagonal grid is extended to the sphere and an efficient compact storage is defined which, in [8] was requested. Spherical grids have been used based on finite volume/finite difference schemes. In this paper, we are interested in Galerkin and L-Galerkin schemes. The existing fully realistic models use the spectral element (SE) method. SE schemes are an exception, as they do not support the sparse grids which are used by the classical Galerkin and the high-order L-Galerkin methods described by St-Li. This paper changes the plane grids for L-Galerkin methods to spherical grids and introduces the other concepts necessary for realistic atmospheric models, such as the definition of compact storage of fields for sparse grids. Simple toy models were given by St-Li for the plane. They transfer to the spherical case when replacing linear distances by great circle distances. This change reflects the change in weights which can be accounted for by geometric files. This concept will be introduced in the next section.

For spherical geometry, we refer to [13] which mainly treats the triangulations needed for navigation on the sphere. For the Galerkin compiler, vector algebra was mainly used, which is discussed in the same reference. The basic mathematical tools and definitions for spherical grids are described in Section 7 of St-Li. The existing models use just a small fraction of the possibilities described there. With the exception of the SE method, most L-Galerkin schemes, in practical use, employ low-order basis functions. This sometimes, but not always, leads to low-order methods. Grid point methods can often be derived as L-Galerkin methods with low-order basis functions. An example of a third-order grid point scheme is MPAS (Model for Prediction Across Scales) [9].

This paper aims for the use of schemes of uniform second- to fourth-order. High-order basis functions are proposed with the L-Galerkin versions. Achieving a high approximation

order with a high-order basis function representation can lead to sparse grids with a potential for saving computer runtime. The high-order basis function method, SEs, cannot use sparse grids, as the use of collocation points to compute Galerkin integrals prevents the use of this source of increased computer efficiency. Both the classical Galerkin, and some of the L-Galerkin methods investigated here, use sparse grids. The collocation grid for SE is shown in Figure 1, which is the same as that used with the classical fourth-order finite difference o4 method. SE, however requires an irregular grid. Such results transfer immediately to the spherical grids, with the only difference being that, for spherical grids, the regular grid case does not exist.

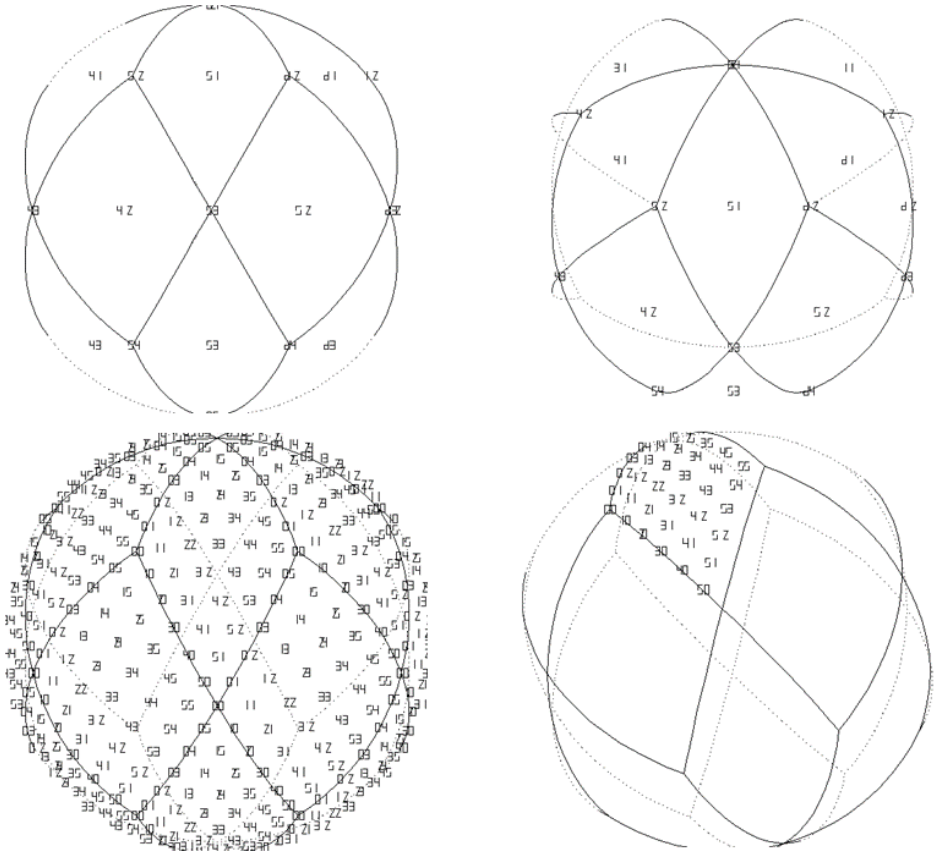

Figure 1. Top left: the patches of the semi-Platonic solid T36 with indices *i,j* of the patch corners: $i = 1,2,3,4,5,6$ and $j = 1,2,3,4,5$ and patch indices: $i = 1,2,3,4,5,6$, $j = 1,2,3$. **Top left** is the view from the *x*-direction. **Top right** is the seme with the north pole visible. **Bottom left** shows the indices *ii,jj* for the cell corner points within the patches for the resolution of 5 * 5 ($i_e = 5$). If classical o4 methods are to be used, the cell indices can also be seen as grid point indices. **Bottom right** shows the indices for just one patch. With this diagram it can be seen why this definition of cell-corner amplitudes results in compact storage. When amplitudes are defined just for the cell corners, as illustrated in this figure, the grid is suitable for most classical schemes, including the classical o4 scheme. For this case, the total number of amplitudes for i_e points on a great circle is $i_e * i_e$ dynamic amplitudes for each of the 6 * 3 = 18 patches. In addition, two amplitudes must be assumed at the two poles. The number of grid points is therefore $3 * 6 * i_e * i_e + 2$. The coarsest grid possible in this frame is obtained with $i_e = 1$. For this case we have 18 + 2 = 20 grid points, which is the number of patch corner points.

St-Li gives simple tests for the L-Galerkin methods on the plane for regular and irregular grids. For toy model demonstrations of L-Galerkin methods, we refer to this, and this paper is concerned with optimizing grid creation methods, transferring such methods to spherical grids and introducing such schemes to real-life models. To create realistic models on such grids, complex software is necessary. In particular, going from a plane to a

spherical grid requires an advanced software construction. The need for good geometrical and software organization was highlighted by the sparse hexagonal grid construction given by [10] For simplicity, these hexagonal cells were considered for the plane. Nonetheless, the program for the simple tests became so complex that, after obtaining some results, the authors were not able to develop the program further and suggested that a new approach for programming and data representation is necessary. This paper defines such a method. In order that a program system can be used, a minimum amount of software engineering is necessary, but this is not sufficient to create a model on spherical grids.

While the grids described here can be used with most methods mentioned in St-Li, we aim for the application of the L-Galerkin methods o2o3 and o3o3 described in St-Li. Together with SE, these methods are further developments of the classic Galerkin methods. While the classic Galerkin method was successfully used on older computers, its use on multiprocessor computers is difficult as they require the global communication of data at each time-step. SE and o2o3 need only local data exchange and this property makes these methods useful with modern multi-processor computers. Furthermore, finite volumes use only local data exchange; finite volumes are, however, mostly used in a version which is second-order on regular grids only. An exception is the MPAS numerical method, which uses finite volumes and is fourth-order accurate because of the use of the Voronoi grid. Other finite volume methods need grid smoothing to have an approximately regular grid to have a reasonable level of accuracy. SE and o2o3 have in common that they use a polynomial basis function representation and, for a third-order polynomial basis, reach fourth-order accuracy on irregular grids. Due to this, they can handle, for example, a 1:2 grid refinement without problems. This property is very useful as regular grids on the sphere do not exist. The main difference between o2o3, SE3 and the SE method with third-order basis functions is that SE3 uses the full grid. o2o3, similarly to the classic Galerkin finite elements, allows the use of sparse grids which, for the classic Galerkin method, are called serendipity grids. For a toy model in 3D, the saving of CPU time due to sparseness was found to be factor 10 (St-Li). A systematic description of the methods mentioned is given by St-Li.

A good geometrical organization of the amplitudes and grid points is necessary, and this, in addition to the software considerations, is the subject of this paper. Here, the IGEL system of grid representation (see St-Li) is used. For grid points i on the sphere, the IGEL grid point r_i consists of three-dimensional (3D) unit vectors where $r_i \in R^3$ has its origin in the earth center $(0,0,0) \in R^3$. These IGEL points can be plotted by projecting each of the vectors r_i to the visualization plane. However, such plots are often confusing if not considering extremely few points. If, for example, the rhomboidal cell corner grid points are plotted in the IGEL system, the surface looks black and like a hedgehog ('Igel' is the German word for Hedgehog). To plot grid points on the surface, it is better to represent each vector by its end point on the sphere. However, the IGEL vectors r_i are very useful to compute quantities such as the great circle distances of two points, the intersection point of two great circles, the spherical angle of two great circle lines etc. Often, the fields necessary to import are given in geographic coordinates. Auxiliary fields to plot such fields, or to perform great circle differentiation, are easily obtained from the IGEL representation. Such auxiliary fields are called geometric fields, as opposed to dynamic fields, such as density or velocities. Using the IGEL representation, extending a grid to neighboring cells to create a halo can easily be carried out. Spherical grids are mainly considered for the icosahedron and the cubed sphere. Here, we take the T36 solid (St-Li) as an example. Geometric considerations can also be used to construct a compact data storage (St-Li) for the sphere. This will be described in Section 2.

The mentioned operations are mostly trivial to program using the IGEL grid, but a large number of such operations is necessary. This set of routines is called the Galerkin Compiler GC_sphere. GC_sphere allows the creation of software for operations on the sphere. The theory is based on St-Li. It allows the construction of grids on all semi-Platonic solids such as the T64 solid (St-Li) and even higher semi-Platonic solids. GC_sphere has

different layers of complexity. The deep layer contains all the simple routines and nearly all the complex operations can be programmed. The higher surface assumes that the IGEL file of the target grid is already created and, then, all relevant geometric fields can be constructed, allowing one to compute all the geometric fields necessary for discretization. Therefore, the deep layer is necessary when new solids, with their grids, are to be constructed. The higher level is sufficient if grids for a particular solid are used and some IGEL files are already pre-computed.

A Galerkin compiler is under construction, named GC_sphere, and is intended to grow until realistic models on different solids and for different grid systems are possible. The purpose is to enable every owner of a model to transfer it to spherical grids. For such intentions, even the higher GC_sphere level may be unnecessarily complicated. To easily allow the creation of a spherical grid version of a given realistic model, the file method is intended for immediate success. The file method consists of creating pre-computed versions of geometric files for a given resolution. The grid for a field consists of four indices for each dynamic or geometric field: for the field $h(r)$ this is $h_{i,j,ii,jj}$. ii and jj are the grid indices and i,j is the index of a patch. Following St-Li, for the T36 solid, the index is written as a double index i,j with i = 1,2,3,4,5,6 and j = 1,2,3.

With this notation, the field index for h is $h_{i,j,ii,jj}$ for the T36 solid discretization, i = 1,2,3, j= 1,2,3,4,5,6, $ii,jj = 0, \ldots, i_e - 1$ where i_e is the number of points in a row of grid points. For the full grid, as indicated in Figure 1, this will result in compact storage for the field. Some other grid systems will be discussed in Section 4. The storage of fields by indices can, for sparse grids, lead to non-compact storage and a system of pointers will be suggested instead. The file method for programming a realistic model will be described in Section 3. This will only be described for a one-layer model. In addition, 3D models can also be treated. For the vertical existing procedures, such as centered difference, vertical schemes, as used in the target model, can be used. When the geometric files are obtained as pre-computed, only the procedure to form the divergence and to create a halo must be programmed, according to St-Li, in order to have a full working model.

St-Li describe the principle of patch generation and give a computational example of the o4 scheme. Simple tests of L-Galerkin schemes were limited to the plane. When the spherical grid patches are given, the results of St-Li can immediately be applied, as the spherical grid case appears as a set of 2D plane grid discretizations with irregular meshes, the irregularity being defined by the spherical geometry. This paper wants provide a step towards the realistic use of the L-Galerkin methods. The software considerations presented aim at achieving this with smaller teams than, for example, those creating the SE models.

The maintenance of complex programs is a point of concern for all meteorological model developments. For example, the model described in [14] had a few hundred contributors and users and the same is true for the MPAS projec1 [9] Rules for the form of software contributions were used and the effect was that most users found it easy to understand the detail of the code and the model code could be used and modified by a large number of people.

For the development of GC_sphere, a different concept is envisaged. Contributors and users are considered to be only occasional workers on this problem. Most of the developers are creating application programs. The GC_sphere lives on the server ECS (Edgar's Communication Server, see huckert.com and search for "meteorology project"). ECS was designed at the time of the COVID pandemic and allows users to cooperate remotely. In particular, all subroutines can be easily shared with other users as all are using the same computer. The software shared by different users is contributed under the assumption of scientific cooperation. Nobody is guaranteed a specific result. Everybody may use his/her own programming style; just the interfaces must be well defined by any contributor. It is intended that research versions of geometric files will be put on ECS. Therefore, everybody is invited to participate in this research. The simplest geometric files are those to be used for data import and plotting. They are already available on ECS and an example will be given in Section 4.

The mentioned theoretical work is at a point where, now, the usefulness of such methods for realistic modelling can be investigated. Realistic models are models using realistic initial data and that include all physical processes, such as radiation etc. Such models are indispensable tools for weather forecasting and climate modeling (for a list of such models, see St-Li). While older models use the spectral methods and latitude–longitude grids, all developments from the last 10 years use spherical grids using the methods mentioned above and described in detail by St-Li. While a few theoretical questions concerning spherical grids and L-Galerkin methods remain open, the new methods need to be tested with realistic modeling. If the development of the global spectral method and spectral elements is an indication, this step into the real meteorological world is a huge project (for a detailed account of the literature, see St-Li). The present suggestion has the purpose of making this step possible by using as much of the existing modeling infrastructure as possible, for example, of the MPAS model. Methods such as the MPAS dynamics, SE or o2o3 increase the order of approximation of model dynamics. It is a remarkable fact that, such increase in numerical accuracy so far has not lead to an increase in forecast scores, which all other research areas, such as the improvement in radiation, have provided. This could point to another serious error, such as an error of the lower boundary condition. The latter is potentially improved by cut cells, the development of which is connected to basis function methods (for an extensive discussion of the related literature, see St-Li).

2. Grids on the Sphere

A description of grid cells based on Platonic and semi-Platonic grids, including graphical illustrations, is given in St-Li. Here, we give additional constructions to be used in working towards L-Galerkin sparse grids and practical modeling. For sparse grids, each cell contains more than one grid point. Methods known for the plane can be transferred to the sphere. It was argued in St-Li that, the grids based on Platonic solids, such as the cubed sphere or the icosahedron, are only approximately regular and, therefore, it makes sense to start with a semi-Platonic solid, rather than a Platonic solid. For sparse grids, the pointer technique, to be explained later, will be used. We follow the results of [15] who, from considerations of practical modeling, concluded that going beyond order 3 for the basis functions does not appear reasonable. This result for SE applies, also, to the L-Galerkin methods, for which the spherical grids are discussed here. Since, so far, sparse grids were constructed only for field representations of order smaller or equal to 3, this makes all L-Galerkin methods under consideration rather similar. We have chosen the T36 solid with 18 basic rhomboids as, optically, it appears to have a more homogeneous cell structure than T20 (icosahedron) or T12 (cubed sphere). Quantitative information on spherical grid homogeneity is not yet available in the literature. The choice between different grids was, so far, mostly established from a visual impression of the plotted grids and no systematic evaluation of grid properties is available.

The grid is defined by the condition that all edges of its 18 patches (see Figure 1) are of equal great circle length. The great circle angle distance of these edges is computed numerically and, for T36, it is angle $ang_{T36} = 0.8571 = \arccos\sqrt{\frac{3}{7}}$ (radian).

The simplest high-order numerical scheme is classical o4, which was used by [9] for a spherical grid based on the icosahedron. SE uses the same grid, but needs to use irregular grid intervals. The o4 scheme is non-conserving and, together with the SE schemes (St-Li), this is the only uniform order 4 L-Galerkin scheme that has been tested on the sphere until now. The grid employed can be used for most classical finite difference schemes and also for the SEs. For the latter, an irregular grid must be chosen. Further developments were proposed by St-Li which require different grids. Such generalizations are the sparse conserving and non-conserving grids, and hexagonal conserving fourth-order sparse grids. With the exception of the classical fourth-order scheme, so far, these schemes have been tested on the plane only. In this paper, the grids will be constructed on the sphere. The file method will be illustrated by the simple example of the geometric files used for plotting fields. The use of geometric files allows a modeling specialist to

implement such methods without becoming a specialist on spherical grids. At least, the programs for differentiation on the sphere using geometric files can be carried out by implementing small and simple routines on the target computer. The simplicity is achieved by using pre-computed geometric files. The simple program for differentiation on spherical grids will be given as an example later. This differentiation routine is universal and can be used with different models. The Arakawa A-grid, shown in Figure 1, is not only the grid for classical o4 and SE. For the sparse grids to be shown in Section 4, the grid of Figure 1 is called the full grid and the sparse grids are obtained by using only some of the points of the full grid.

It is intended that, from the example of the o4 grid, this paper will move towards more advanced schemes involving grid sparseness. The software and grid aspects will be treated in Section 3. In this section, we consider questions of grid organization and halo construction. The patches and their indices are shown in Figure 1. The patches have equal sides and consist of great circle lines as edges. The numbering of the patch corner points is defined in St-Li and is indicated in Figure 1. The corners have double indices I = 1,2,3,4,5,6 and j = 1,2,3,4,5. For j = 1 and j = 5, we have the north and south poles. For the poles, the different i indicate the same point. This means that the number of different patch corners is 3 * 6 + 2=20. The number of patches is 18. The patches are indicated by the double index i,j, with i = 1,2,3,4,5,6 and j = 1,2,3. In Figure 1, the patch numbers are indicated in the middle of a patch. The great circle angles of patch edges and cell edge lines are important to compute field derivatives. The patch corners at the poles are shared by six patches, those at the equator are shared by four patches and those between the equator and poles are shared by three patches. From this, and the fact that the edge lines of patches are equal, it follows that the patch angles at the poles are 60° and that two of the angles at the equator are also 60°. It also follows that the other two angles at the equator are 120°. In this paper, we want to add grid cells for sparse grids to the patches.

Each of the rhomboidal patches has a grid indexed by ii,jj = 0,1,2, ..., $i_e - 1$. Figure 1 gives a plot of the indices ii,jj for i_e = 5. It can be seen that, with the indices ii,jj = 0,1,2, . . . , i_e, some lines of grid points belong to more than one patch. This representation is redundant and is not a compact representation of the grid (see St-Li). Consulting Figure 1, it can be seen that, having run ii and jj only to $i_e - 1$ implies no redundancy and nearly all grid points are represented. Using the 18 grids for the patches with end point $i_e - 1$ and, in addition, the two pole amplitudes, results in a compact representation. This representation includes all grid points and has no double storage for any points.

It is important to have a compact representation for efficient model development. The compact storage described above for the grid uses data parcels of equal size. There is another option whereby the grids for patches on the equator (see Figure 1) are slightly larger than those around the poles. With this option, the grids for equator patches are indexed ii,jj = 0,1,2, ..., i_e. For the north-pole and south-pole patches, the grid indices are ii = 0,1,2, ..., $i_e - 1$, jj = 1, . . . , $i_e - 1$.

For the classical o4 scheme, the derivative of a field $h_{ii,jj}$ in the direction of the ii index uses two points to the right and to the left of the target point:

$$h_{x,ii,jj} = w^{-2}_{ii,jj} h_{ii-2,jj} + w^{-1}_{ii,jj} h_{ii-1,jj} + w^{0}_{ii,jj} h_{ii,jj} + w^{1}_{ii,jj} h_{ii+1,jj} + w^{2}_{ii,jj} h_{ii+2,jj} \quad (1)$$

In Equation (1), the indices i,j were suppressed. For further explanations in connection with this, see St-Li. Equation (1) assumes that the grid points are lined up on a great circle, which is the assumption we make in this paper. Rhomboidal full and irregular grids, with points not lining up on great circles, are rather expensive and complicated to implement. Baumgardner's cloud method (see St-Li) is an example of second-order approximation on an irregular grid. Its generalization to third order is too expensive to be feasible (Baumgardner (2004), private communication [16]). Hexagonal sparse grids are also theoretically rather efficient for irregular grids. This was tested for a plane grid (see St-Li). Using the hexagonal grid geometry explained later, this can be transferred to the

sphere. The weights w in Equation (1) depend on the great circle differences between the grid points involved. The case for the plane is the same.

Grid point differences on the plane must be replaced as great circle differences for the spherical cases. The treatment of patch boundaries is different for different methods and for sparse grids. For the example of the classical o4 method, Equation (1) shows that grid points outside the patch are necessary, which are called the halo.

The Galerkin compiler proposed in Section 3 will allow one to compute the weights in Equation (1). To compute the flux divergence, the great circle angle between the coordinate great circle must be known. This is, again, carried out using the routines of the Galerkin compiler. There is another problem to be solved in connection with Equation (1): the index *ii,jj* must be transformed into a compact grid representation. If a point is near the patch boundary, up to two halo lines of data are needed outside the compact grid. As indicated above, for some lines such information can be copied from other patches without interpolation. This grid is called the redundant grid. When consulting Figure 1, it is seen that some data have to be taken in reverse order. For $i_e = 5$, the data in the target grid are indexed $ii = 0,1,2,3,4,5$ and they are copied from a grid line indexed $jj = 5,4,3,2,1,0$. For an equatorial grid patch two lines are required outside the patch. Therefore, the halo grid is: $ii = 0,1,2, \ldots, i_e, jj = -2,-1,0, \ldots, i_e, i_e + 1, i_e + 2$ and analog for *jj*. The points with indices smaller than 0 or larger than i_e must be obtained by interpolation. This was performed in Steppeler et al. (2008) for the icosahedron and the shallow water equations. Here, we intend to design this in such a way that people from outside, with, at most, a week of introduction, can use this method to create their own model. When using the Galerkin Compiler as described in Section 3, much longer introductory times will be needed. From experience with comparable projects, it is assumed that six months to three years of work is necessary to transfer a numerical procedure running in a toy model to a realistic model.

Therefore, an even simpler method is suggested to enable the owner of a model to transfer it to a spherical grid. This method consists in the GC_sphere programmers providing geometric files to the effect that the model programmer has to create only very simple programs. Geometric files are all index fields $g_{ii,jj}$ which are not dynamic. For simplicity, we again suppress the indices *i,j*. As an example, the simple *ii,jj* grid to be used with the classical o4 method is used. It is possible to extend this to more complicated numerical methods and grids, such as sparse grids or hexagonal/triangular grids. We envisage, however, that this increase in complexity is more on the side of the GC_sphere programmer and that the job of the model developer is not increasing too much by tackling the more complex grids. This method is described for shallow water models on the sphere. Three dimensional modelling is possible, as the vertical classical methods can be applied. For example, the vertical discretization from the target model can be taken over. To obtain large sparseness numbers, L-Galerkin methods may be used also in the vertical and, in this case, this can be carried out analog to the horizontal. Full efficiency is obtained when using the L-Galerkin method also in the vertical, for which a 1D discretization must be added to the horizontal scheme. Such 1D programs are given in Appendix 2 of St-Li.

For a shallow water model, the model programmer has to define the storage for the fields *u,v,h*. Our description is only for the treatment of the field *h*. For *h*, the modeler must provide compact storage $h_{ii,jj}$, $ii.jj = 0,1,2, \ldots, i_e - 1$. The modeler also needs to provide halo fields with $ii,jj = -2,-1,0,1, \ldots, i_e, i_e + 1$. The halo field contains two additional lines in *ii* and *jj* directions.

The geometric grids are provided for a specific resolution. Therefore, it is necessary that these fields are obtained for different values of i_e, dependent on the resolutions for which the model is to be run.

It is necessary [14] that the *h* field is given as a function of the geographic coordinates (λ, φ). The geometric fields necessary are $\lambda_{ii,jj}$ and $\varphi_{ii,jj}$.

Using this field, the initialization of the model is carried out by:

$$h_{ii,jj} = h\left(\lambda_{ii,jj}, \varphi_{ii,jj}\right) \tag{2}$$

Equation (2) already provides the shallow water model to be constructed with initial data for h. Such data can also be plotted in the $\lambda - \varphi$ space. We need to use a plot routine using a list of x, y, $\lambda_{ii,jj}$, $\varphi_{ii,jj}$ and $h_{ii,jj}$ values as input. Furthermore, the projection into the x-y plane can be obtained by using the geometric field $x_{i,j,ii,jj}$ and $y_{i,j,ii,jj}$. These geometric fields need to be provided by the GC_sphere programmer and the modeler need not know how this is performed. Now, we assume it to be the case that the dynamic equations are a set of conservation laws. For classical o4 differencing, we need to differentiate the right-hand side $RHS_{ii,jj}$ of the dynamic equation, where, as an example, we consider differentiation in the direction of ii, assumed to represent the coordinate x. We need to compute the right-hand side on the halo grid. Thus, we call the routine "grid extension" which creates the halo grid by interpolation. On the server ECS, this routine will be available. On other computers, it is necessary to create such a routine. This is easy, as the interpolation geometric files can be made available from GC_sphere once for all. The right-hand side is then available in grid point form $RHS_{ii,jj}$. Then, differentiation according to Equation (1) must be carried out. Again, on ECS, a differentiation routine will be available, otherwise it can be created using Equation (1) and the weights w provided by geometric files.

Time differentiation can be performed by any method available. On ECS, examples for time differentiation exist, or can be provided by other users. ECS is a cooperative platform.

This provision of modules, here mainly data and field modules, is common for the development of numerical weather prediction (NWP). In the early days, the author created plots without any complicated software, only commands steering the pencil. While for plotting and spectral transforms, for example, the use of software packages has become common place, there is no reason that spherical-discretization-designated software should not be used. When the resources to create such a software package are missing, the use of the platform ECS, and the provision of geometric files, is an option available with little effort.

The grids are presented in 2D form. As for the vertical, we envisage a simple column-wise discretization; no special description for the 3D form is necessary. As sparseness in 3D is much larger than in 2D, the sparseness factors in Table 1 will also be given for 3D.

Table 1. Number of grid points for the different grids.

Schemes	o3	o3o3	hex_o3	hex_o3_QUASAR
i_e = 9 (2D)	1458	810	432	324
i_e = 9 (3D per three layers)	4374	1134	540	432
dx	555 km	555 km	555 km	555 km

3. The Galerkin Compiler GC_Sphere

The word compiler implies that we have a well-organized piece of code designed to be used, not only by the developers, but rather by a large community. Here, we think of the routines for the construction of grids on the sphere. A simple method is the grid construction on bilinear polygons, which are then projected onto the sphere. This method is illustrated in Figure 2 and described in detail in St-Li. With a limited programming effort, it can be realized. Further routines are needed to analyze grid lengths, and other diagnostic quantities, to compute the halo grid, the weights for differentiation, etc. Even though most such routines are simple, the number of small routines needed is such that it will be a major piece of software. In order to be useful, this needs to be shared by a larger group. The result would be that it would be a major task to create an L-Galerkin model. The file method described in Section 2 allows one to achieve this without specialized knowledge on spherical grids. This reduces the number of people necessary with knowledge of the compiler.

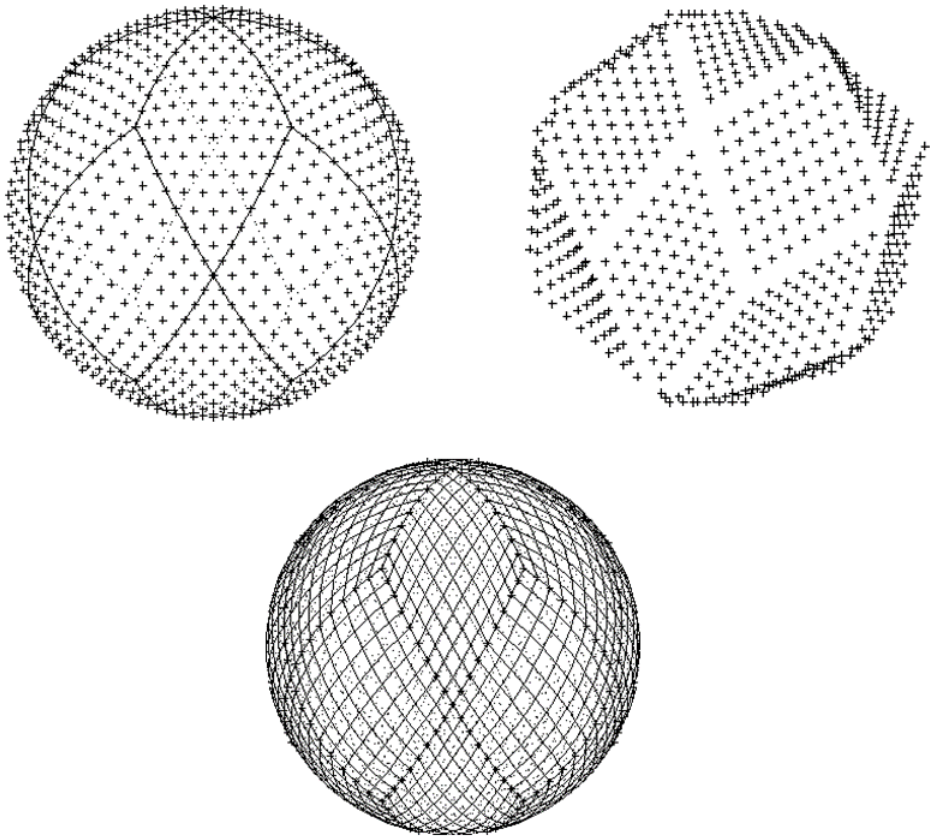

Figure 2. **Upper left**: grid points in spherical projection with patch edges as solid lines. **Upper right**: same for projection to bilinear surfaces, where the edge grid points are omitted. The bilinear surfaces are defined by corners and edges of the T36 solid. Its construction is described in St-Li and is based on Euclid's lemma. **Lower**: representation of grid lines. Points on patch edges are indicated as *. It can be seen that, at patch corners, at all places where 4 or 6 edges meet, the edges of different patches continue as straight lines. This means that, for these corners, we have straight line stencils. For the full grid there is one cell per point.

Early NWP models were specific for a computer, and the way to transfer them to other sites was to reprogram them using model descriptions. Even today, a number of research and toy models are constructed with no provisions to make them available to other researchers. This means that cooperation for such models is possible only by joining the development group and taking over the complete model. Certainly, the trend in NWP within the last 50 years is towards portability of models and codes. Another trend that can be seen is towards separate program packages, such as plotting, which are delivered by specialized groups. This paper suggests the use of spherical grid software in the same way as with plot routines.

Today's NWP models are often portable. For example, the model LM [10] used strict coding rules and a code management system and was developed by a dedicated group.

For the development of GC_sphere, no such dedicated group is visible. Therefore, a more chaotic way of programming must be tolerated. The aim is to allow cooperation on spherical grids without having a central management, and to collect contributions as voluntary offers. Some work on spherical grids with uniform high order, such as [7,8], was carried out on a shoestring budget and the authors came together in person or by e-mail contact to work on the problem. While the need for personal meetings and exchange of guest scientists will remain, it is desirable to accompany this with efficient ways of making spherical grid expertise easily available to interested researchers. For realistic meteorological modelling on the spherical grids discussed here, resources for a well-

organized programming effort are not visible to the author. Therefore, cooperation, leaving the initiative to each of the partners involved, is considered.

A first step is the use of the file method, which allows for the use of spherical discretization with few preparations. This method, as proposed in Section 2, requires the use of some pre-defined resolutions. The difference method is also pre-defined and, for example, can be the classical fourth-order method. The modeling equations, for example those for shallow water, 3D non-hydrostatic or passive advection, are approximated by the user of the file method.

If modifications of the derivative approximation, other resolutions or alternative grids, such as adaptive are desired, new geometric files must be created. The same is true for modifications of the grid, for example, for different choices of the grid-dividing relations on patch edges. For this, the use of the program GC_sphere is necessary. As this program is not sufficiently organized in its present form, it cannot be easily shared between different users. The solution is that cooperating users employ the same computer where GC_sphere resides, which is ECS. For further information, see huckert.com and search for "meteorology project". This site is organized in such a way that each user contributes programs for sharing or not for sharing, as he or she likes. The programs offered to be shared are put into the public folder. GC_sphere is on this site. As no good program organization or manual is available, this is a difficulty for its use. The solution to this problem is that, for a certain project, such as investigating the grid lengths, only a few of the commands are necessary, the use of which will be no problem when entering cooperations. In this way, both shortcomings of GC_sphere can be overcome. As an example, let us consider that, currently, ECS has only rudimentary plotting facilities. These plots are used for debugging the grid creation, and figures in this paper were created using the elementary plot procedures of GC_sphere. Attractive plots of results would require better plotting routines, which can be provided by any volunteering user of ECS and made available using the public folder. On the other hand, a user of the file system for modeling may want to plot the grids used and change details of the grid. To achieve this, just a few routines from GC_sphere are necessary and, with the current state of the program, the only way of using such routines is through cooperation. It is planned that, for the adventurous user, the geometric files on huckert.com will be offered and able to be copied without the need to become a client of ECS.

GC_sphere contains only basic plotting routines. Figure 1 shows the full grid for $i_e = 5$ with the indices. The resolution $i_e = 9$ would make index plotting difficult.

4. Examples of Sparse Grids

GC_sphere is in a state of development. Together with ECS, it will allow the construction of grids of different types on the sphere. These grids will not be limited to the cubed sphere and the icosahedron, which presently are the most popular versions of spherical grids. The examples presented here are based on the T36 solid (St-Li). T36 divides the sphere into 18 rhomboids. Here, we choose the version with all rhomboids having sides of equal great circle length. When i_e is the number of points on edges, $i_e = 9$ for the full grid is the shortest resolution, allowing the third-order polynomial representation on the sphere (St-Li) with full, sparse rhomboidal and hexagonal cells. The grid points for $i_e = 9$ are shown in Figure 3. The edge lines of the patches are plotted as solid lines. It can be seen that, for patch corners at poles and on the equator, grid lines are not angled and continue as straight lines. This means that 1D difference stencils can be used. For the 12 patch edges between pole and equator, one of the great circle lines meets the grid points, when continued into the neighboring patch. This means that, of such points, differentiation in one of the directions can be carried out in 1D. For differentiation near patch-edge points, we have 3- and 4-legged stencils, which can be treated by one-sided third-order differentiation (See St-Li). The sparse grid for o3 representation is shown in Figure 2. The numerical treatment is analog to the plane case, except that, near patch edges, angled stencils or interpolation must be used (St-Li).

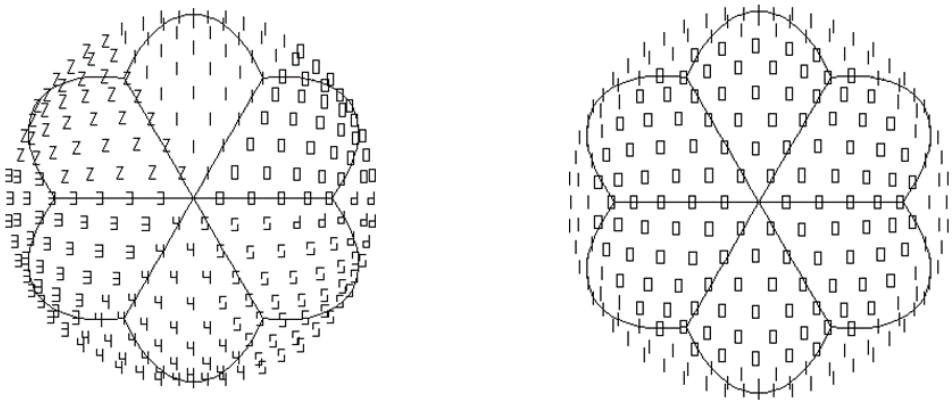

Figure 3. Lot of values of λ (**left**) and φ (**right**) in projection to the *x*-*y* plane. This is an example of plots to be used for debugging purposes. Gridded data of λ and φ are simple examples of geometric fields and can be used to create initial fields and for plotting.

The sparse grid may be obtained by defining an i_e * i_e storage and have some points unused. This, however, leads to an inefficient program; we need a compact storage, where all storage points are used. The purpose of sparse grids is not only to save storage but also to have an efficient message passing, which is possible with compact storage. We can see from Figure 3 that the cells have 4 * 4 = 16 points (inner points of a rhomboidal cell). The corners and edges belong to more than one cell. From Figure 3, it can be seen that corner points belong to either 3, 4 or 6 cells. The edge points belong to two cells. The full grids have 16 amplitudes per cell. All amplitudes on corners and edges belong to more than one cell. For sparse grids, there are 12 points per cell. For the full grids and patches, being full surfaces of the original solid, the indices are illustrated in Figure 1. Figure 1 also shows a method of indexing whereby each dynamic point has an unique index, even though an edge or corner point may belong to more than one patch. The extension of a grid of a patch by neighboring points is called a halo creation and this operation may require message-passing on multi-processor computers. As cell corner points belong to four cells, and edge points belong to two cells, we have five independent dynamic amplitudes per cell. Let us identify a rhomboidal cell by the index of its left corner; the cell grid is then within the cell *ii*, *jj* = 0,1, . . . , $i_e/3 - 1$. In our example of i_e = 9, the indices within a cell take the values 0,1,2. When the patch indices are *i* = 1,2,3,4,5,6 and *j* = 1,2,3, the total cell index for the cell c is:

$$c = c_{i,j,ii,jj}(i = 1,2,3,4,5,6, j = 1,2,3, ii, jj = 0,1,...,i_e - 1) \quad (3)$$

This index is also the index of the leftmost corner point of the rhomboid. A cell with index *i,j,ii,jj* (here *ii,jj* = 0,1,2) has five independent grid points with index *k* = 1,2,3,4,5. A field *h* on the spherical grid is indexed as:

$$h = h_{i,j,ii,jj,k}(k = 1,2,3,4,5) \quad (4)$$

The naming convention is such that, for *k* = 3, we have the leftmost corner point of the cell. The relation of the patch and cell indices *i,j,ii,jj* and pointer *k* to the full grid index *i,j,ii′,jj′*, where *ii′,jj′* = 0,1,2,..., i_e is explained below.

A simple example of a geometric field is λ and φ depending on the spherical grid indices. For the classical o4 grid shown in Figure 3 this is $\lambda_{i,j,ii,jj}$ and $\varphi_{i,j,ii,jj}$. These geometric files can be used to transfer the field data available in $\lambda - \varphi$ coordinates to the grid. The same geometric field can also be used to plot a gridded field in the $\lambda - \varphi$ plane; for an example, huckert.com transfers to the ECS server and give examples of geometric files for people who are not users of ECS.

When gridded fields are given, the divergence of a flux vector is a weighted sum of some neighboring grid points. When the indices of the neighbors and the weights are given as geometric files, the computation of time derivatives and the creation of a realistic model is not too difficult. Therefore, if modelling is limited to cases where geometric files are available, the transfer of a realistic model to spherical grids is not too difficult and will not require a large knowledge of spherical geometry.

The pointer variable k is introduced to create a compact storage for sparse fields. This is explained for the rhomboidal cell as shown in Figure 4. We introduce $cell_{\mathrm{dim}} = \frac{1}{3} i_e$, where, for i_e, only values divisible by three are admitted. While we have ii,jj = 0,1,2,3,4,5,6,7,8,9 for the case i_e = 9, we replace them by ii_c and jj_c, ii_c,jj_c = 0,1,2 for the rhomboidal case shown in Figure 4. For the rhomboidal sparse grid case in Figure 4, we have five dynamic amplitudes belonging to the cell $ii,jj = 3ii_c,3jj_c$. The points belonging to cell ii,jj are:

$$\begin{cases} ii+2, jj \\ ii+1, jj \\ \quad ii, jj \\ ii, jj+1 \\ ii, jj+2 \end{cases} \tag{5}$$

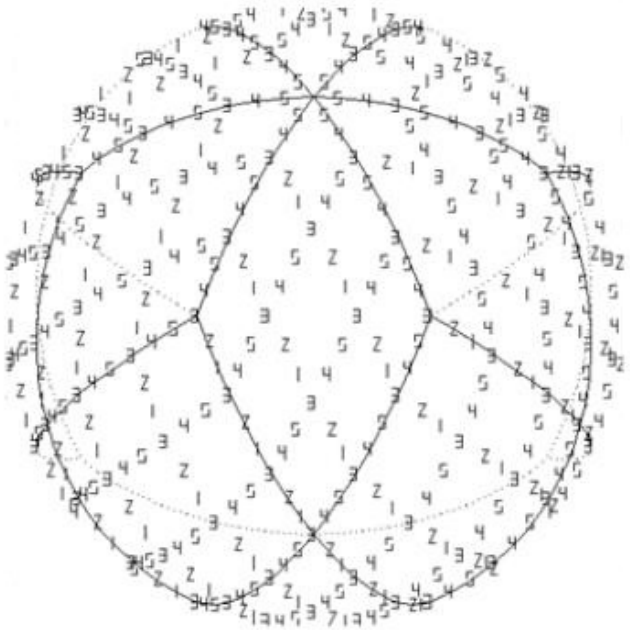

Figure 4. The sparse rhomboidal grid on the sphere The grid points are indicated by Numbers from 1 to 9 which indicate the pointer variable. This pointer variable numbers the amplitudes for each cell. It is seen thet the amplitudes from each cell combine to give a complete sparse grid within each cell.

For the pointer index k = 1,2,3,4,5, the points given in Equation (5) are defined in the sequence given in Equation (5). We have already seen in Section 1 that these points form a compact set. For the case of a 2D sparse rhomboidal grid, we have for the pointer k the range k = 1,2,3,4,5, and we obtain the first index k = 1 shown in Equation (5):

$$h_{ii_c,jj_c,k} = h_{3ii_c+2,jj_c} \tag{6}$$

For values of k = 2,3,4,5, analog equations are obtained from Equation (5). In Equation (6), ii_c, jj_c are the compact indices and ii,jj are the indices of the full grid. Note that, for the sparse rhomboidal case treated here, all amplitudes are on the cell edges.

The other indices shown in Equation (5) are defined analogously. Figure 4, at the bottom right, shows the position of the pointer indices k on the sphere. While with i_e = 9 the range of ii,jj is 0,1,2,3,4,5,6,7,8, the range of ii_c,jj_c is 0,1,2. The indices ii_c,jj_c, k cover 5 * 9

values for i_e = 9 and are a compact representation. The indices *ii,jj* cover 81 values and are not a compact set.

Hexagonal grid cells are created as sub-grids of the full grid shown in Figure 2 They are available when the cell grid numbers i_e^{cell} is a multiple of 3: $i_e^{cell} = 3cell_{\mathrm{dim}}$. We have, then, $i_e = 9cell_{\mathrm{dim}}$. The smallest hexagonal grid on the sphere is obtained for $cell_{\mathrm{dim}} = 3$ and i_e = 9. The sparse hexagonal grid was used in a plane [7] Here, we created the grid for a sparse hexagonal o3 method on the sphere where all amplitudes are on the hexagonal edges. As a previous attempt suffered from inconvenient data storage, here, the pointer method is used for compact data storage. A hexagonal cell is numbered by the index of the cell center. A hexagonal cell has 6 corner point amplitudes and 12 edge amplitudes. A compact subset can be obtained by the pointer method. This means that, for the definition of compact storage, we have 8 amplitudes to be defined by the pointer variable of the full grid and an additional number of inner 18 + 1 = 19 points. This means that, for the sparseness factor in two dimensons, we have 8: (19 + 8) which is much larger than obtained with rhomboids for the plane. The sparseness becomes much larger with three dimensions. The QUASAR method (St-Li) potentially brings the sparseness to 6:27.

The sparse hexagonal grid for $cell_{\mathrm{dim}} = 1$ has i_e = 9 and 3 hexagons per patch. Some of the hexagons degenerate to pentagons. St-Li used the o2 representation while, here, the grid for o3 is created. Figure 5 shows the grid:

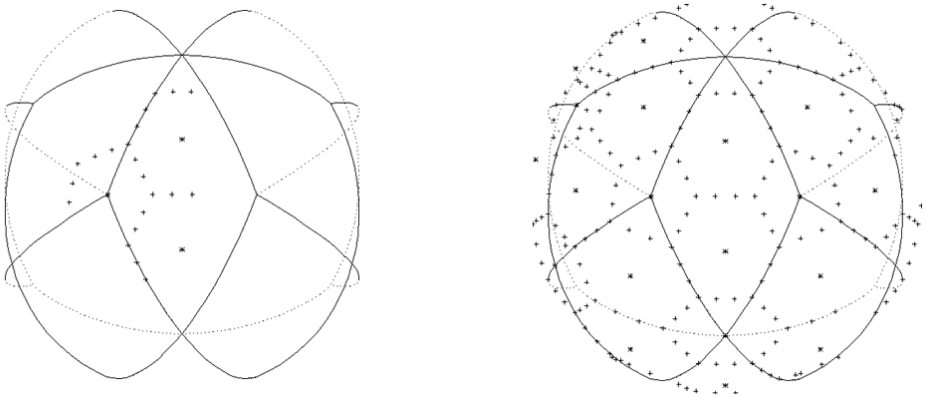

Figure 5. Left: position of amplitudes for the sparse hexagonal grid for patch j = 1,i = 5. The hexagonal centers belonging to this patch are indicated by *. Note that the points indicated by * carry no amplitudes, but the amplitudes at some of the surrounding edge and corner points are stored there. Since for i_e = 9 this is the lowest resolution with hexagonal cells on the sphere, we have three hexagons/pentagons belonging to this patch. Two are inside the patch and one on a patch corner. The dynamic amplitudes belonging to these hexagons are indicated by +. Note that some of the cell grid points belonging to this patch belong to the neighboring patches j = 1, i = 4. In combination of all patches we have a complete sparse hexagonal grid on the sphere. Note that the cell center does not carry an amplitude. To the center of a hexagonal cell we have eight compact amplitudes and for pentagonal cells, five. Note that the poles are centers of hexagonal cells, but the corresponding amplitudes are carried by neighboring cells. The indicated amplitude positions form a compact set. This means that, the combination of the amplitudes for all cells provides all grid points. **Right**: Plot of the amplitude points for the combination of all patches. With * those hexagon centers are noted which carry data. The north pole is the center of a hexagon, but this does not carry amplitudes. The south-pole hexagon center carries data and it stores 18 grid points, 6 corner grid points and 12 edge points.

5. Examples

Examples of geometric fields are $\lambda_{i,j,ii,jj}$ and $\varphi_{i,j,ii,jj}$. These fields can be used to import initial value into a spherical grid model and to plot such grids. Figure 3 gives an example of a plot based on these geometric files and also for the projection to the x-y plane, which

requires the geometric files $x_{i,j,ii,jj}$ and $y_{i,j,ii,jj}$. This demonstration is for the full grid or the o4 grid. The procedures for forming derivatives will depend on further geometric files, which are differentiation weights, angles between coordinates, etc. When such procedures have to be created inside a target model, this is potentially time consuming. When geometric files containing differentiation weights, coordinate line angle, etc. are pre-computed, the creation of such differentiation routines is expected to require exchange visits of no longer than one week.

The gridded values of $\lambda_{i,j,ii,jj}$ and $\varphi_{i,j,ii,jj}$ allow the importing of the initial fields into spherical grids.

As this example is rather simple, we give another example for differentiation using the classical o4 method on the full grid. For a patch, the points are indexed by *ii*,*jj*. The formula given below are easily transferred into a computer program: let a field *fl*, such as a flux component belonging to a dynamic field, be given by $fl_{ii,jj}$ where, again, the patch index *i*,*j* is suppressed. The coordinate directions *ii* and *jj* are on great circles. In the differentiation in *ii* directions given by Equation (1) the weights $w^{-1}_{ii,jj}$, etc. must be given as geometric files where, from Equation (1), we can see that there are five w gridded fields which must be given as geometric files. For differentiation in *jj* directions, we need weights $w^{-1}_{ii,jj}$, which are another five files to be provided as geometric files. In order to have the derivative in any other direction, such as x and y, we must know the angle $\alpha_{ii,jj}$ and can, for the classical o4 method, compute divergences and time derivatives of fields and use any time marching method to integrate the model.

The idea of geometric files is that the cooperation between the spherical modeler and the specialist of a meteorological model, such as WRF, MPAS or ICON, need not be close. The implementation of Equation (1) as a program is simple enough that it may be sufficient to pick up a geometric file from the internet site ECS (go through huckert.com) to create a model in the precomputed resolution. Note that, for the SE method, the same procedures can be used as for o4. For SE, the grid is more irregular than for o4, but SE also uses the full grid.

Other L-Galerkin methods can be used in a similar way, such as o3o3. In particular, with sparse grids, the geometric files require adaptation to the grid.

Note that we create a procedure for differentiation. Once this differentiation routine is available, the form of the equation to be used will still be the choice of the meteorological modeler, not the spherical grid provider. For example, the discretization of $\frac{1}{2}u_x^2$ or $u \cdot u_x$ is established by the same differentiation procedure; only the preparation of the field to be differentiated on the grid is different. The options for the form of equations, and how to approximate the product of the fields, are the choice of the model developer.

6. Conclusions

The paper described a system of spherical grids on the T36 solid, including rhomboidal/triangular cells, for the classical o4 differencing, for o3o3 sparse grid schemes and for sparse hexagonal cells on the sphere. A software concept designed for the cooperation of a number of independent partners is described and a simple example given. This concept imposes few rules for the participants and, consequently, no definite expectations on other partners are possible. The provision of a set of geometric files allows the transfer of a realistic model to spherical grids without deep knowledge of spherical grids. The use of the method of geometric files and rather few very simple subroutines, in combination with geometric files, can be used to transfer L-Galerkin toy models existing on a plane to the sphere and, in a second step, change them to realistic models by using a target model such as WRF or ICON. It is suggested that GC_sphere be used for visualizing the sphere in a similar way as a system of plot routines are used to visualize model fields.

7. Future Research Directions and Possibilities with Exa-Flop Computers and L-Galerkin Methods for Weather Prediction and Climate Research

This paper mainly concerns L-Galerkin methods on sparse grids with the aim of reducing computer run times. It can be expected that global models for forecasting purposes will reach a resolution of 1 km globally for weather forecasts and less for climate research. Within a grid box, LES simulations using 1000 * 1000 * 1000 points seem realistic. Local basis functions were used. For such applications, it will also make sense to use basis functions with the support of the whole modeling area where, often, very few amplitudes are used. Such applications may be called low dimensional approximations as, in comparison to the solutions with local basis functions, the solution is a low dimensional surface in phase space. For climate applications, such solutions can define the attractor. Note that the definition of climate, with only a few measuring sites, requires the assumption of a low dimensional attractor. A famous example of a low dimensional climate attractor is the Chaotic Loernz attractor [15], which is obtained by a low dimensional approximation for the convection between plates. This means that people living in a cubic cave would have the Lorenz chaotic solution [2] as the climate attractor. Living in caves is a consideration for settlement on Mars. Galerkin methods can help to find attractors for the earth system including its dynamic equation. For the very high resolutions of global weather prediction models which are now envisaged, such low dimensional attractor solutions could be considered for the smallest model scales. Consider, for example, a global model with grid boxes of 1 * 1 * 1 km. Using a high resolution within such a box, the simulation of turbulence is possible. It seems likely that there is a 1D attractor whose amplitude is determined by the boundary amplitudes of the box. The attractor may be computed by offline high resolution simulations of a single grid box, and L-Galerkin methods allow one to compute the effect of the local solution on the global solution. L-Galerkin methods work in the same way for polynomial or other solutions. Only local basis functions can be shown to be polynomials. The different box solutions may fit together to form a discontinuous field and are complimented by the continuous function spaces used for larger scales. As currently discontinuous and continuous Galerkin methods are a subject of research, the formalism to carry out such a project is already known. No concrete steps have, however, been taken to realize such models which, by low dimensional modeling within a grid box of 1 * 1 km, would extend the resolution from 1 km to a few cm.

Funding: This research received no external funding. The dropping of page fees by the editor is gratefully acknowledged.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The MOW_GC Galerkin Compiler, as described in the paper, is in a state of development and results are geometric files which, for research resolutions, will be made available on ECS. Furthermore, it is planned that the divergence-forming routine using geometric files will be made available there as a source code. MOW_GC was used for all figures shown in this paper. As no manual is planned, its use is possible by entering co-operations with the author.

Acknowledgments: The author thanks Jinxi Li (from Institute of Atmospheric Physics, Chinese Academy of Sciences) for his technical help with creating the manuscript and for his cooperation with the elaboration of basic spherical tools, as described in St-Li.

Conflicts of Interest: The author is the owner of the company MOW&more which does project management for the commercial and non-commercial applications of global models, including spherical grids and cut cells. Thus far, all existing co-operations were performed without MOW&more, but MOW&more sponsored office space for the co-operations and contributed to the MOW meetings.

References

1. Steppeler, J.; Li, J. *Mathematics of the Weather*; Springer Atmospheric Sciences; Springer: Cham, Switzerland, 2022. [CrossRef]
2. Steppeler, J.; Li, J.; Fang, F.; Zhu, J.; Ullrich, P.A. o3o3: A variant of spectral elements with a regular collocation grid. *Mon. Weather Rev.* **2019**, *147*, 2067–2082. [CrossRef]
3. Sadourny, R. Conservative finite-difference approximations of the primitive equations on quasi-uniform spherical grids. *Mon. Weather Rev.* **1972**, *100*, 136–144. [CrossRef]
4. Williamson, D.L. Integrations of the barotropic vorticity equation on a spherical geodesic grid. *Tellus* **1968**, *20*, 643–653.
5. Baumgardner, J.R.; Frederickson, P.O. Icosahedral discretization of the two-sphere. *SIAM J. Numer. Anal.* **1985**, *22*, 1107–1115. [CrossRef]
6. Steppeler, J.; Prohl, P. Application of finite volume methods to atmospheric models. *Beiträge Phys. Atmosphäre* **1996**, *69*, 297–306.
7. Zängl, G.; Reinert, D.; Rípodas, P.; Baldauf, M. The ICON (ICOsahedral Non-hydrostatic) modelling framework of DWD and MPI-M: Description of the non-hydrostatic dynamical core. *Q. J. R. Meteorol. Soc.* **2015**, *141*, 563–579. [CrossRef]
8. Steppeler, J.; Rípodas, P.; Jonkheid, B.; Thomas, S. Third-order finite-difference schemes on icosahedral-type grids on the sphere. *Mon. Weather Rev.* **2008**, *136*, 2683–2698. [CrossRef]
9. Skamarock, W.C.; Klemp, J.B.; Duda, M.G.; Fowler, L.D.; Park, S.H. A multiscale nonhydrostatic atmospheric m.odel using centroidal Voronoi tesselations and C-Grid staggering. *Mon. Weather Rev.* **2012**, *140*, 3090–3105. [CrossRef]
10. Sröcker, H. *Taschenbuch Mathematischer Fomeln*; Verlag Harry Deutsch: Frankfurt am Main, Germany, 1993.
11. Williamson, D.L.; Drake, J.B.; Hack, J.J.; Jakob, R.; Swarztrauber, P.N. A standard test set for numerical approximations to the shallow water equations in spherical geometry. *J. Comput. Phys.* **1992**, *102*, 211–224. [CrossRef]
12. Steppeler, J.; Li, J.; Fang, F.; Navon, I.M. Third-order sparse grid generalized spectral elements on hexagonal cells for uniform speed advection in a plane. *Meteorol. Atmos. Phys.* **2020**, *132*, 703–719. [CrossRef]
13. Lorenz, E.N. Deterministic nonperiodic Flow. *J. Atmos. Sci.* **1963**, *20*, 130–141. [CrossRef]
14. Herrington, A.R.; Lauritzen, P.H.; Taylor, M.A.; Goldhaber, S.; Eaton, B.E.; Bacmeister, J.T.; Reed, K.A.; Ullrich, P.A. Physics–dynamics coupling with element-based high-order Galerkin methods: Quasi-equal-area physics grid. *Mon. Wea. Rev.* **2019**, *147*, 69–84. [CrossRef]
15. Steppeler, J.; Doms, G.; Schättler, U.; Bitzer, H.W.; Gassmann, A.; Damrath, U.; Gregoric, G. Meso-gamma scale forecasts using the nonhydrostatic model LM. *Meteorol. Atmos. Phys.* **2003**, *82*, 75–96. [CrossRef]
16. Baumgardner, D.G. (Droplet Measurement Technologies, Inc., Longmont, CO, USA). Personal communication, 2004.