University of Tartu

Faculty of Science and Technology

Institute of Ecology and Earth Sciences

Department of Geography

Master's thesis in Geoinformatics for Urbanised Society (30 ECTS)

# Integrating environmental datasets into a Data Cube using Discrete Global Grid Systems

**Ivan Vasilyev**

Supervisors:

Alexander Kmoch, PhD

Holger Virro, MSc

Allowed to defence:

Supervisor:

Head of Department:

Tartu 2021

**Master Thesis in Geoinformatics for Urbanized Society: Integrating environmental datasets into a Data Cube using Discrete Global Grid Systems**


## Abstract

The aim of this thesis was to evaluate applicability of currently available open-source DGGS implementations as a basis for a data cube construction for global-scale environmental datasets management and spatial statistical analysis. Following open-source DGGS implementations were explored in this work: Uber H3, Google S2, RiskAware OpenEAGGR, rHEALPix by Landcare Research New Zealand and DGGRID by Southern Oregon University. Geometric properties of these DGGS implementations were compared based on two metrics of a cell shape: normalized area and Zonal Standardized Compactness (ZSC). Additionally, influence of a DGGS partition aperture level on a cell area change dynamic between DGGS resolutions was examined.

For the construction of a DGGS-based data cube-like structure 6 openly available datasets for Estonian territory were loaded into H3 DGGS. This demonstrated the potential DGGS approach has when a particular implementation is widely adopted and standardized. Unique cell indexes which unambiguously encode the same location throughout different platforms and technologies opens a perspective of 'geometry-free' spatial data manipulations. It has also been shown that with developed DGGS library as a basis for data cube it is possible to easily combine data from different sources, with different initial resolution and data formats on the global scale.



Key words: DGGS, Data Cube, Coordinate Reference Systems, Spatial Indexing, Spatial Data Models

CERCS code: P510 Cartography


**Linnastunud ühiskonna geoinformaatika magistritöö: keskkonnaandmestike andmekuupi integreerimine jagatud globaalsete võrkude süsteemide (DGGS) abil**


## Abstrakt

Uurimustöö eesmärk oli hinnata praegu kättesaadavate avatud lähtekoodiga DGGS-rakenduste kasutatavust globaalsel skaalal keskkonnaandmete haldamiseks ning statistilise ruumianalüüsi teostamiseks andmekuubi koostamise alusena. Töös uuriti järgmisi avatud lähtekoodiga DGGS-rakendusi: Uber H3, Google S2, RiskAware OpenEAGGR, Landcare Research New Zealandi rHEALPix ja Lõuna-Oregoni Ülikooli DGGRID. Nende DGGS-rakenduste

geomeetriliste omaduste võrdlemiseks kasutati kahte võrguruudu kuju näitajat: normaliseeritud pindala ja tsonaalne standarditud kompaktsus (ZSC). Uuriti ka DGGSi jaotusastme mõju DGGS-resolutsioonide vahelise võrguruudu ala muutumise dünaamikale.

DGGS-põhiste andmete kuubilaadse struktuuri ehitamiseks laeti kuus Eesti territooriumil avalikult kättesaadavat andmestikku H3 DGGS-i. See tõestas DGGS-meetodi potentsiaali juhul, kui konkreetne rakendus on laialdaselt kasutusel ja standarditud. Erinevate platvormide ning tehnoloogia kaudu üheselt sama asukohta kodeerivad unikaalsed võrguruuduindeksid pakuvad andmete „geomeetriavabade“ ruumiliste manipulatsioonide võimaluse. Tõestati ka, et kui andmekuubi alusena kasutatakse H3 DGGS-teeki, saab globaalsel skaalal kergesti ühendada erinevatest allikatest pärinevaid, erineva esialgse resolutsiooni ning andmeformaadiga andmeid.

Võtmesõnad: DGGS, andmekuup, koordinaatide referentssüsteemid, ruumiline indekseerimine, ruumiandmete mudelid

CERCS kood: P510 Kartograafia

# Table of Contents

## Introduction

In the era of Big Data and ongoing globalization, scientific community, public and private companies who deal with spatial data face new challenges related to processing enormous amounts of data. Several petabytes of geospatial data occupy servers around the world, and more continues to be generated every day (Mahdavi-Amiri et al., 2015). In such conditions, traditional models for storing, managing and analyzing spatial information are becoming inadequate due to global extent and increasing demand for distributed computation capacity, as it stated by many authors (Sahr et al.,2003, 2015; Sahr, 2008; Mahdavi-Amiri, 2015; Purss et al., 2019). One of possible solutions to tackle this issue is the use of Discrete Global Grid Systems (DGGSs). In a DGGS, the surface of the Earth is discretized into a set of regular cells. These cells are then addressed using a data structure or indexing mechanism that is further used to assign and retrieve data. The DGGS theory has been discussed for decades in the scientific community, but the interest among wider public has gained momentum in recent years including emergence of numerous DGGS-based software products in various domains (Adams, 2017; Uber Technologies Inc, 2018; Global Grid Systems, 2019; Bush, 2017; Gibb, 2016; Robertson et al., 2020) and the recent introduction of the DGGS abstract specification by Open Geospatial Consortium (OGC) (Peterson, 2019).

Another technological concept in the spotlight of the geospatial community for managing Big Spatial Data is the so-called data cube. Data cubes are n-dimensional arrays that are used to store query-ready spatial-temporal data ordered according to various attribute/coordinate axes, which can be spatial or non-spatial in nature (Alderson, 2020). The widest recognition and adoption of data cubes have been gained in recent years in Earth Observation (EO) domain. There is a growing number of implementations currently referred to as EO data cubes (Swiss Data Cube, The Australian Geoscience Data Cube, Open Data Cube to name just a few) with the goal of optimizing the way to store, manage, provide access to and analyse Big EO Data in a more convenient manner (Augustin et al., 2019).

Most of the raw global data cannot easily be analyzed because typically produced in World Geodetic System 84 (WGS 84) geographic coordinate reference system. This spatial data is further projected to planar reference systems to index and align data in order to be able to query, aggregate, summarize and otherwise spatially analyze across raw observation data and thematic layers. This conventional approach of managing spatial data in map-projected data sets has led to a restricted set of data cube implementations that are each tightly coupled to the spatial

constraints of the data and how they are stored resulting in barriers to interoperability and analysis on global scales (Purss et al., 2019). The equal area global properties of DGGS and its implicit global scope would allow for building a globally valid interoperable data cube.

The aim of this thesis is to evaluate the applicability of DGGSs as a basis for a data cube construction for global-scale environmental datasets management and statistical spatial analysis. To achieve the aim the following research questions are stated:

1) Is DGGS suitable for a data cube indexing?
2) What type DGGS is suitable for global-scale data cube construction by its geometric properties among currently available open-source implementations?
3) Which currently available open-source DGGS implementations are suitable for integration with larger software ecosystem to build practical real-world application?

To answer these questions, first, it is necessary to establish theoretical foundation of DGGS. Second, explore experimentally geometric properties of several professional publicly accessible open-source implementations of DGGS. Third, discuss DGGS implementation-specific advantages and disadvantages in particular with focus on supporting a data cube implementation. Finally, demonstrate an example of operational data cube-like structure for the test region.

# 1. Theoretical overview

## 1.1 Traditional spatial data models and coordinate systems

In the very foundation of the geospatial domain, there are two dominating data models for digital representation of real-world entities (Bolstad, 2008). In a vector data model point, line or polygon features represented as sets of coordinates and associated non-spatial attributes of any nature. Raster data model represents the world as a regular grid with non-spatial attributes linked to the grid cells centroids. Commonly, these cells have a square shape and evenly distributed along Cartesian coordinate axes.

Coordinates in the data models are usually initially defined in geographic reference system on spherical or ellipsoidal surface as angular values (latitudes and longitudes). Further, however, it is more common to transform these angular coordinates into 2-dimensional Cartesian coordinates on a flat map projection surface since these projected coordinates are easier to work with for any kind of spatial analysis and geocomputations. A map projection is a function for transformation of coordinates from the surface of an Earth model into coordinates on a flat surface (Bolstad, 2008). A map projection can preserve angles, areas, distances, or directions, but never a combination of all the above. No map projection can preserve both area and angles. There are hundreds of map projections are in use in the geospatial domain and it is often a challenge to choose appropriate one according to the particular application and location of an area of interest.

In the recent years, rapid increase in Earth Observation data accumulation and the rising trend towards Open Data have made a vast variety of high quality, high resolution, globally covering data available to researchers and spatial data analysts (Alderson et al., 2020; De Sousa et al., 2019). To efficiently handle and store this massive volumes of data and provide seamless interoperability without necessity to perform computationally expensive transformations from one projection to another global map projections are increasingly important (Seong et al., 2002; De Sousa et al., 2018; De Sousa & Poggio, 2019). De Sousa (2019) examined 5 most common equal-area global projections available in GIS software: Sinusoidal, Eckert, Homolosine, Mollweide and Hammer. It appeared, that these projections although provide significantly better performance in terms of storage and computation efficiency comparing to other global projection methods (like Mercator-based projections) still have area and shape distortions distributed unevenly throughout the projection surface. Moreover, these global equal-area

projections are still poorly implemented in open-source geoprocessing libraries (De Sousa, 2018).

Attempts were made to introduce multi-projection system with individual projection parameters for continents - the Equi7 Grid (Bauer-Marschallinger et al., 2014). Projections which are used in The Equi7 Grid are based on Equidistant Azimutal and, although do not preserve areas, the oversampling factor is the lowest among discussed which is desired for data storage purposes. However, De Sousa (2018) pointed out that such approaches create its own problems, such as computational overhead in managing different projections simultaneously and overlaps between various Cartesian spaces.

An alternative approach of Discrete Global Grid Systems as a reference system and spatial data model is considered in many studies as a better option to handle Big Spatial Data (Purss et al., 2019; Sahr et al., 2004).

## 1.2 Data Cubes

A data cube represents a multidimensional data array together with metadata describing the semantics of axes, coordinates, cells, and values (Purss et al., 2019). The widest recognition and adoption of data cubes have been gained in recent years in Earth Observation domain together with unprecedent grow in remote sensing data acquisition. There is a growing number of implementations currently referred to as EO data cubes (Swiss Data Cube, The Australian Geoscience Data Cube, Open Data Cube to name just a few) with the goal of optimizing the way to store, manage, provide access to and analyze Big EO Data in a more convenient manner (Augustin et al., 2019). Earth observation data cubes have emerged as a promising solution to efficiently handle Big Earth Observation Data generated by satellites and made freely and openly available from different data repositories (Giuliani et al., 2019). According to The Datacube Manifesto (Baumann, 2017) a data cube “is a massive multi-dimensional array, also called “raster data” or “gridded data”; “massive” entails that we talk about sizes significantly beyond the main memory resources of the server hardware. Data values, all of the same data type, sit at grid points as defined by the d axes of the d-dimensional data cube. Coordinates along these axes allow addressing data values unambiguously”.

In a typical (or “conventional”) data cube implementation, data values that are all the same data type sit at grid points that are defined on the orthogonal spatial axes of the data cube. Such quadrilateral grids may be based on coordinate systems, hierarchical categorizations, spatial

features, or any other system that allows unambiguous referencing of a cell. This tightly couples the structure of such data cubes to the usually two-dimensional spatial reference frame of the data stored within. (Purss et al., 2019).

Most of the raw global data (satellite derived, modelled climate data, land use, GPS-based location data) typically produced in World Geodetic System 84 (WGS 84) geographic coordinate reference system. This spatial data is further projected to planar reference systems (such as UTM zones or country-specific coordinate systems) to index and align data in order to be able to query, aggregate, summarize and otherwise analyze across raw observation data and thematic layers. This conventional approach of managing spatial data in map-projected datasets have led to a restricted set of data cube implementations that are each tightly coupled to the spatial constraints of the data and how they are stored resulting in barriers to interoperability and analysis on global scales (Purss et al., 2019). The equal area global properties of DGGS and its implicit global scope would allow for building a globally valid interoperable data cube.

## 1.3 Discrete Global Grid Systems

### 1.3.1 Brief history and main definitions

A Discrete Global Grid (DGG) is a set of cells which covers the whole surface of the globe. The most common and well-known example of discrete grids are latitude-longitude graticules (Purss, Gibb, Samavati, Peterson, & Ben, 2016). A discrete global grid system (DGGS) is series of such discrete global grids (Barnes, 2019). Global grid systems are not something new as such systems has been under development and implementation with a different extent of success for almost half a century. Over the years many different discrete global grids have been developed, with very different properties, advantages and drawbacks. The terminology for such systems has been changing before it got its current definition: hierarchical spatial data structures (Goodchild & Shiren, 1992), recursive partitions of the globe (D. White, 2000; Denis White, Kimerling, Sahr, & Song, 1998) hierarchical tessellations (G. Dutton, 1989). Now, with a new Open Geospatial Consortium (OGC) standard recently been introduced, the term for such grid systems has been more clearly established. According to the definition from this OGC standard, “a Discrete Global Grid System is a spatial reference system that uses a hierarchical tessellation of cells to partition and address the globe. DGGS are characterized by the properties of their cell structure, geo-encoding, quantization strategy and associated mathematical functions” (OGC Abstract Specification, 2017).

Formal development of DGGS as it known now began in late 1970s with rapid development of use of geographic information systems and emergence of the first global datasets from remote sensing and positioning systems. The first discrete global grid system was implemented by Geoffrey Dutton in 1984 (Purss et al., 2016). This global grid was designed for assembling and managing global terrain data on a triangular global grid, which was the first prototype for what is known now as Quaternary Triangular Mesh (QTM) (G. Dutton, 1989). It is stated that besides traditional graticule QTM is potentially the most established global data structure in terms of algorithms developed for its use (Gregory et al., 2008). Dutton pointed out that it is difficult to estimate to what extent his work influences the development of DGGS concepts. However, he didn't get any recognition of this input rather than from scientific community although large corporations patented some of his key ideas (Dutton, n.d.).

The next wave of scientific and practical interest in global grid systems happened in the early 90, driven by interdisciplinary group from Oregon State University - Dennis White, Scott Overton, Jon Kimerling and Kevin Sahr (Sahr et al., 2004). Their work was focused on equal area tessellations of polyhedrons with further projection it on the sphere. In the same time the most well-known polyhedral map projections such as Icosahedral Snyder Equal Area and Fuller-Gray were developed and formalized which had a great deal of impact on development of this kind of global grids (Gray, 1995; Snyder, 1992).

The most recent surge in interest in DGGS in the past 10-15 years is related to the overall development of information technologies, all-pervading integration of location-based services, earth observation and location data accumulation at a scale and volume which haven't been seen before (Guo, 2017; Mahdavi-Amiri, Alderson, & Samavati, 2015; Purss et al., 2019; Sahr, 2011; Sahr et al., 2004). Geospatial computing and analysis are more widely embedded in business and science than ever. These trends together with the availability of more and more powerful and cheap cloud-computing resources supporting the interest in novel, more efficient approaches to manage Big Spatial Data.

Not every tessellation of the Earth does necessarily produce a DGGS. Single resolution global grids are not sufficient to create a DGGS. Spatial data structures used to organize map tiles or optimize rapid spatial search cannot be considered to qualify as a DGGS in and of themselves; although DGGS often utilize hierarchical indices to identify a cell, the primary feature of the DGGS is the cell geometry not the optimization of a spatial query (Purss et al., 2016).

Particular DGGS implementation steps, which also define its main properties, formulated by Kevin Sahr (2004) as so called ‘design choices’ (Figure 1):

1. A base regular polyhedron.
2. A fixed orientation of the base regular polyhedron relative to the Earth.
3. A hierarchical spatial partitioning method defined symmetrically on a face (or set of faces) of the base regular polyhedron.
4. A method for transforming that planar partition to the corresponding spherical or ellipsoidal surface.
5. A method of indexing grid cells.

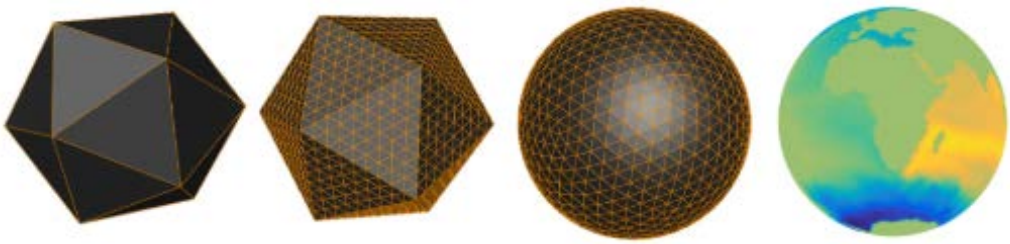

*Figure 1. DGGS construction steps (left to right): the choice base solid and orientation, tessellation of the base solid, projection to the sphere, data ingestion (Source: Jubair et al., 2016).*

These 5 steps are widely accepted in scientific community – the most of analyzed further publications use same classification. In the following sections these characteristics are described in more detail.

### 1.3.2 Construction of a DGGS

**Base solid**

Many different initial solids have been proposed for DGGSs construction. Among the most commonly used to construct DGGSs are the platonic solids (Figure 2): the tetrahedron, cube, octahedron, dodecahedron, and icosahedron (Mahdavi-Amiri, Alderson, et al., 2015).

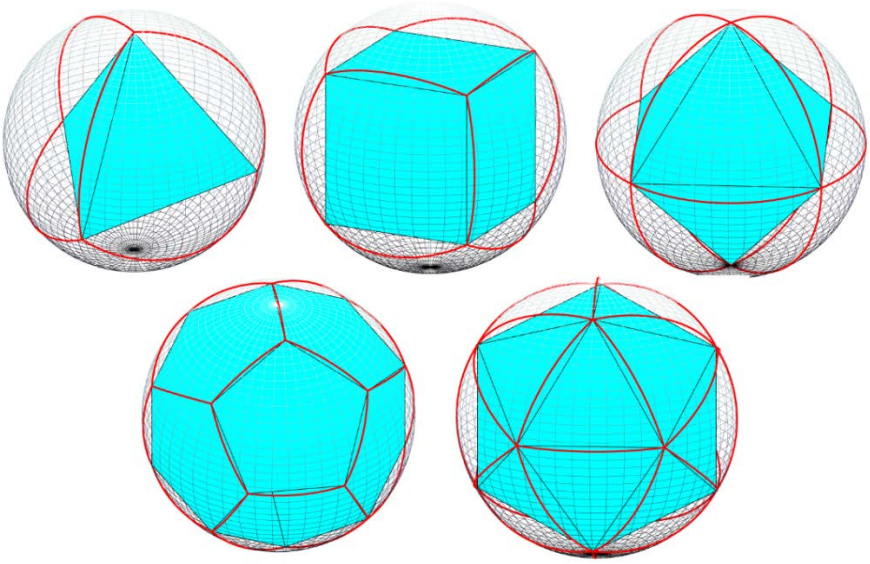

*Figure 2. Platonic solids and their spherical representations (left to right, top to bottom): tetrahedron, cube, octahedron, dodecahedron, icosahedron (Source:* Lei et al., 2020).

Dutton's QTM uses octahedron (Dutton, 1996), the same solid was also implemented by Goodchild & Shiren (1992) and White (2000). The octahedron has the advantage that it can be oriented with vertices at the north and south poles, at the intersection of the prime meridian and the equator, and thus, aligning its eight faces with the spherical octants formed by the equator and prime meridian. This makes conversion from geographic coordinates to DGGS relatively straightforward problem because it is easy to define on which face latitude/longitude point lays. But, because the octahedron has comparably larger faces than some other solids (Figure 3),

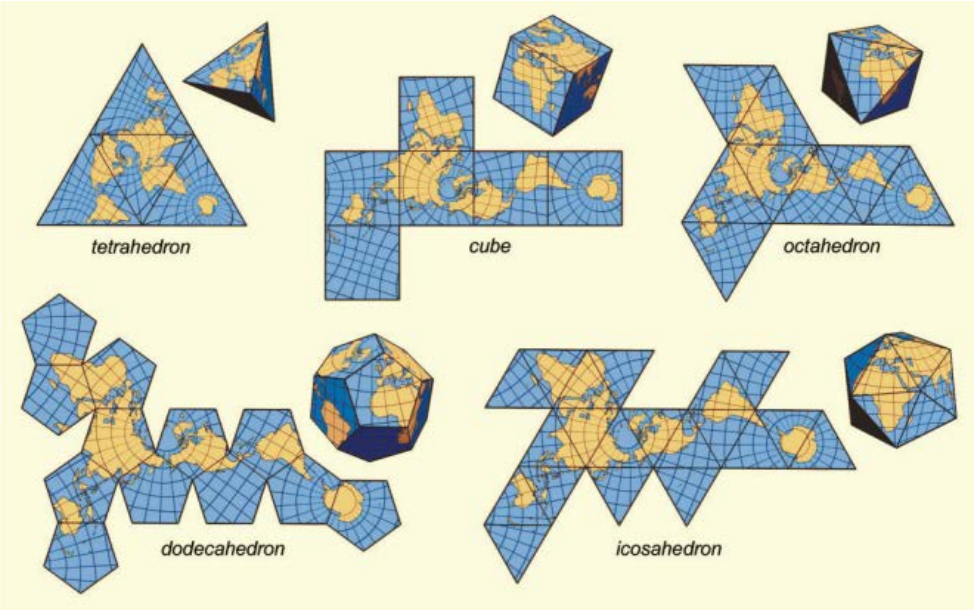


*Figure 3. Platonic solids and corresponding map projections* (Source: Wijk et al., 2008)

partitions defined on the faces of the octahedron tend to have higher distortions when projected back to sphere (Kevin Sahr et al., 2004). Generally, it is well-known in cartography that the larger map projection surfaces the harder it is to minimize distortions when projecting spherical surface onto it. Although the cube has the largest faces and therefore highest distortions when projected it is also widely used as a base for DGGSs (Amiri et al., 2013; Gibb, 2016). The most significant advantage of the cube over other solids is the possibility to tessellate it with rectangular-shaped cells, which is the basis for the most common spatial indexing algorithms. Quadrilateral cells also are more compatible with hardware and display devices and they are also compatible with familiar Cartesian coordinates (Amiri et al., 2013; Lei et al., 2020; Sahr et al., 2004).

The icosahedron tends to be considered as the most suitable option as a base shape. Long before first mentions of DGGS, in 1954, this solid was used by Buckminster Fuller for his Dymaxion world map (Figure 4) and Dymaxion projection later (Gray, 1995).

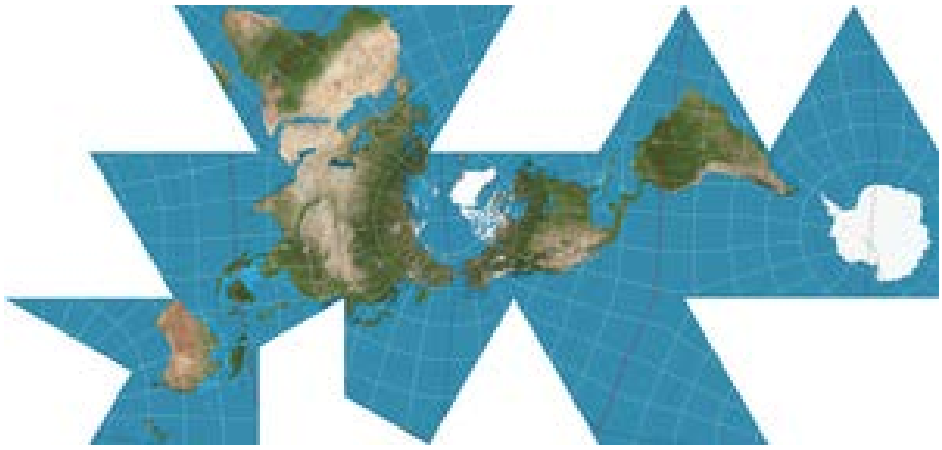

*Figure 4. Buckminster Fuller's Dymaxion world map (Source: https://en.wikipedia.org/wiki/Dymaxion_map)*

This platonic solid has the smallest face size among others and therefore less distortions when projected back to the sphere (Sahr, 2004, Carr et al., 1997). The icosahedron thus became the most frequently chosen base regular polyhedron (Bush, 2017a; Lin et al., 2018; Robertson et al., 2020; Sahr et al., 2011; Uber Technologies Inc, 2018).

However, direct tessellation of the sphere without rely on projection surfaces is also in use, latitude/longitude grids as the simplest example (Kimerling et al., 1999; White, 2000; Zhou et al, 2016). Such grid systems are outside of the scope of the current work.

**Base solid orientation**

Several orientations can be chosen for the base polyhedron, depending on where its vertices intersect the sphere. Generally, distortions are not equal across the whole surface of the polyhedron when projected to sphere (Bush, 2017a). So, the best orientation will be dictated by the nature of application of the DGGS being constructed. If the areas of interest is mostly located on the landmasses, then it is feasible to locate the polyhedron in a way that areas with maximal distortions are located in the oceans (Uber Technologies Inc, 2018). Ideally then the base polyhedron should be orientated with its vertices away from the areas of most interest. (Bush, 2017a).

For the icosahedron three of the most common orientations are polar, when a vertex placed at each of the poles; symmetrical to the equator, when only one vertex placed on land; or the Dymaxion orientation, when all vertices located in oceans (Sahr et al., 2004). The orientation of the octahedron is usually chosen so that its vertices are located at the poles and edges align with the equator and prime meridian, which makes further cells indexing relatively straightforward (Geoffrey Dutton, 1996; Goodchild & Shiren, 1992). Common orientation for the cube is placing two opposite faces centers at the north and south poles (A. M. Amiri et al., 2013; R. Gibb, Raichev, & Speth, 2016; Google groups, n.d.).

Barnes (2019) proposed a quantitative approach for the detection of the most optimal polyhedral orientation based on the Poles of Inaccessibility concept. The proposed algorithm is trying to find positions of the polyhedron that maximize the distance of vertices from a coastline of any landmass. In summary, finding suitable orientations of polyhedrons for DGGS is an optimization problem of finding the optimal distance of coastlines to the polyhedrons' vertices and edges (Barnes, 2019).

**Partitioning shape**

Choosing the shape of the partitioning cell is arguably the most crucial choice during DGGS creation because it defines substantial properties for spatial analysis and statistics. Generally, there are only 4 shapes suitable for a continuous tessellation of the sphere: squares, triangles, rhombus and hexagons (Sahr et al., 2004). Initially, the partition cell choice is dictated by the polyhedron chosen during previous step if it is not a sphere. Triangular-faced solids – the icosahedron or octahedron cannot be tessellated by square-shaped cells (although rhombus is suitable), while the cube can only be tessellated by squares (A. Amiri, Samavati, & Peterson,

2015; Kevin Sahr et al., 2004). Further considerations on the tessellation shape selection will be affected by the shapes properties and particular DGGS application. There is an abundance of different discussions in the scientific literature about advantages and drawbacks of different shapes for the global, as well as local grids.

*Squares*

Squares (Figure 5) undoubtedly are the most popular and traditional shapes for planar discrete grids (Birch, Oom, & Beecham, 2007). A great deal of analysis and indexing algorithms have been developed for square shapes. Most of sensors and displays operate using square shaped pixels so all Earth Observation data comes in form of rasters with rectangular pixels. Geographical Information System rasters and remote sensing thus make a large contribution to the current dominance of rectangular grids (Sahr et al., 2004). Moreover, the square-shaped cells are naturally suitable for Cartesian coordinate systems and commonly used data structures such as quad-trees. Single and multi-resolution indexing algorithms based on space-filling curves are generally developed for square-shaped grids (Amiri et al., 2015).

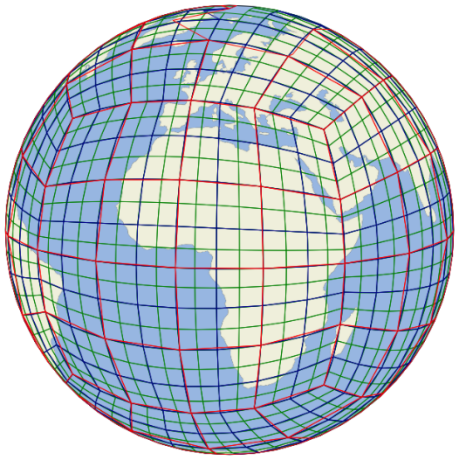

*Figure 5. 3 subsequent resolutions of an aperture 4 square-shaped refinement*

Another property which makes squares and other rectangular shapes so often a desirable choice is congruency – a square can be seamlessly subdivided into as many child squares as needed which naturally supports multiresolution structures. This subdivision is also called refinement - converting a set of coarse cells to finer cells through splitting edges or shrinking cells (Amiri et al., 2015). If a cell is subdivided into i parts it calls on-to-i refinement with an aperture of I (Amiri et al., 2015). Density between recursive subdivisions is important criteria for aggregation/disaggregation issues and smooth transitions between DGGS resolution levels. Therefore, a refinement with a lower factor of subdivision (aperture) is usually desired, as it

can generate a greater number of resolutions under a fixed number of maximum cells (Birch et al., 2007; Mahdavi-Amiri et al., 2014; Sahr et al., 2004). For the quadrilateral shapes the refinement with an aperture 4 is the most widely used which means that parent square is subdivided into 4 child squares, although an aperture 9 (Gibb et al., 2016) and 2 has also been proposed (Figure 6) (Amiri et al., 2013).

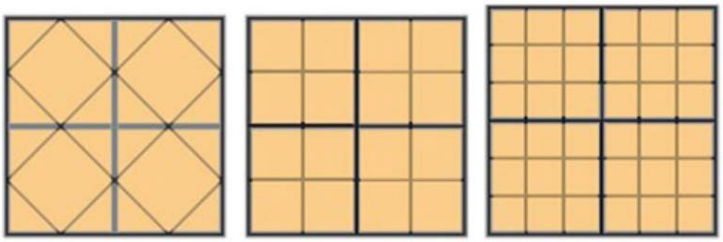

*Figure 6. Square-shaped refinements (left to right): aperture 2, aperture 4, aperture 9 (Source:* Alderson et al., 2020).

*Triangles*

However, as mentioned before, since the commonly used polyhedrons in DGGS initially have triangular faces, triangular cells are also widely used in DGGSs (Figure 7). These cells are also congruent, consequently, quad tree-based data structures are applicable. Triangular shapes can be rendered

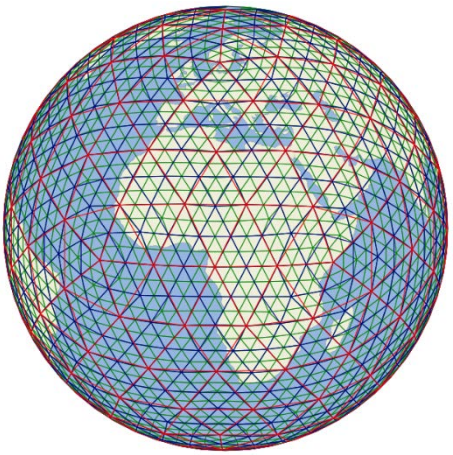

*Figure 7. 3 subsequent resolutions of an aperture 4 triangular-shaped refinement*

very efficiently, as they are supported by many built-in functions in rendering pipelines, such as OpenGL (Amiri et al., 2015). Numerous DGGSs implementations are based on aperture 4 triangular tessellation, or, in other words, a grid of triangular cells where each triangle is subdivided into 4 triangles for the next level of higher resolution (Bush, 2017a; Dutton, 1989; Dutton, n.d.; Goodchild & Shiren, 1992). But, while triangles are suitable for icosahedron tessellation and relatively convenient to work with, Sahr et al. (2004) pointed out that they have

a number of disadvantages as the basis for a DGGS - like square grids, they do not exhibit uniform adjacency - each cell having three edge and nine vertex neighbors (Figure 8).

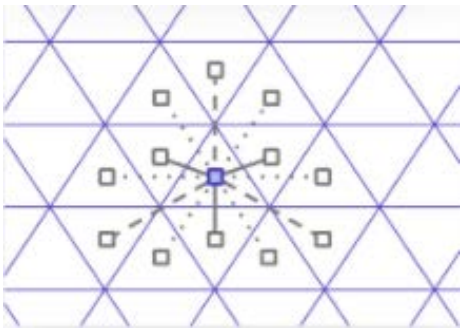

*Figure 8. Adjacency in triangular refinement (Source: https://h3geo.org)*

Also, unlike squares, the cells of triangle-based discrete grids do not have a uniform orientation. At each successive resolution, the orientation of the inner triangle flips. This complicates the implementation of algorithms such as adjacency analysis, spatial queries, and data update on DGGSs built with this shape of cells them, so the algorithms have to take this changing orientation into account (Sirdeshmukh, 2018).

*Diamonds/rhombuses*

Rhombus (diamond) shape has all advantages of squares and can also be used for a complete tessellation of an icosahedron (Figure 9). Because diamond-based grids have a topology identical to square-based grids they can take direct advantage of quadtree-based algorithms, which are inherently used for all square-based analysis (Sahr et al., 2004). White (2000) shows several significant advantages of diamond-shaped cells over other forms. Diamonds, like squares, maintain the same size, shape, and orientation at each level in the DGGS hierarchy. The diamonds hierarchy is nested, each parent diamond completely contains all its children

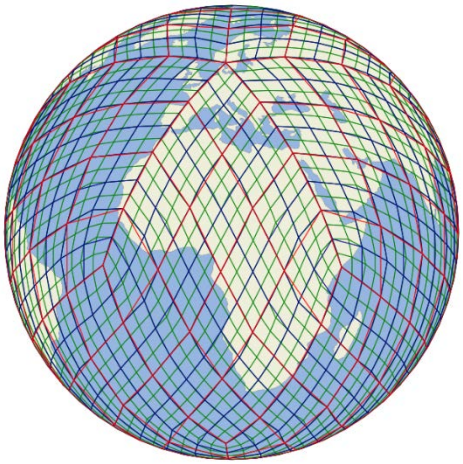

*Figure 9. 3 subsequent resolutions of an aperture 4 diamond-shaped refinement*

diamonds. Also, one of the most beneficial property of diamonds is that space-filling curve algorithms, can be used to index the cells of a rhombus-based DGGS. However, like square grids, diamond-shaped cells do not have uniform adjacency – each cell shares 4 edges and 4 vertices with neighboring cells (Figure 10) (Sahr et al., 2004).

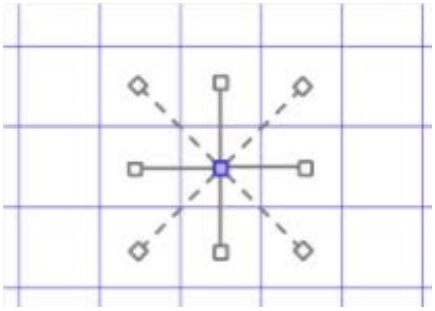

*Figure 10. Adjacency in rectangular refinement (Source: https://h3geo.org)*

*Hexagons*

Some researchers claim that honeycombs and other hexagonal structures are the most common regular pattern in Nature, both for flora and fauna (De Sousa et al., 2018). Hexagonal structures and patterns are regularly reappearing in studies related to image processing, geospatial location coding and geospatial modeling already for many years (Sahr, 2011). In recent years interest towards hexagonal structures has been increasing in areas such as ecological modeling (Birch et al., 2007), hydrological modeling (Liao et al., 2020), image processing (De Sousa et al., 2018), AI and deep learning (Luo et al., 2019).

The superiority of hexagons is stemming from their unique geometric properties: amongst 3 regular polygons that tessellate the plane (triangles, squares, and hexagons), hexagons have the highest symmetry and are the most circular, or compact shape (Sahr, 2011). The higher

compactness of a hexagon is a result of a shorter perimeter in relation to its area compared to a square or a triangle of the same area. Plus, some parts of a square or a triangle are farther from its center than any part of a hexagon of the same area (Birch et al., 2007). In addition, unlike square and triangle grids, hexagon grids display uniform adjacency (Figure 11): each cell in a grid of hexagons has six neighbors, all of which share an edge with it, and all of which have centers exactly the same distance away from its center (Sahr et al., 2015). Each hexagon cell has no neighbors with which it shares only a vertex, as do square or triangular cells (Figure 11).

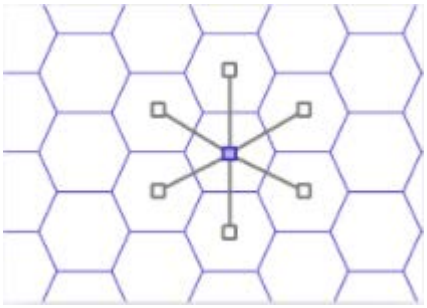

*Figure 11. Adjacency in hexagonal refinement (Source: https://h3geo.org)*

Looking at the frequency of appearing in particular implementations and scientific publications related to geospatial grids hexagons seem to have the most attention as a shape for partitioning in DGGSs construction (Alderson et al., 2020; Amiri et al., 2019; Purss et al., 2016; Robertson et al., 2020; Sahr, 2011; Sahr et al., 2011; Uber Technologies Inc, 2018).

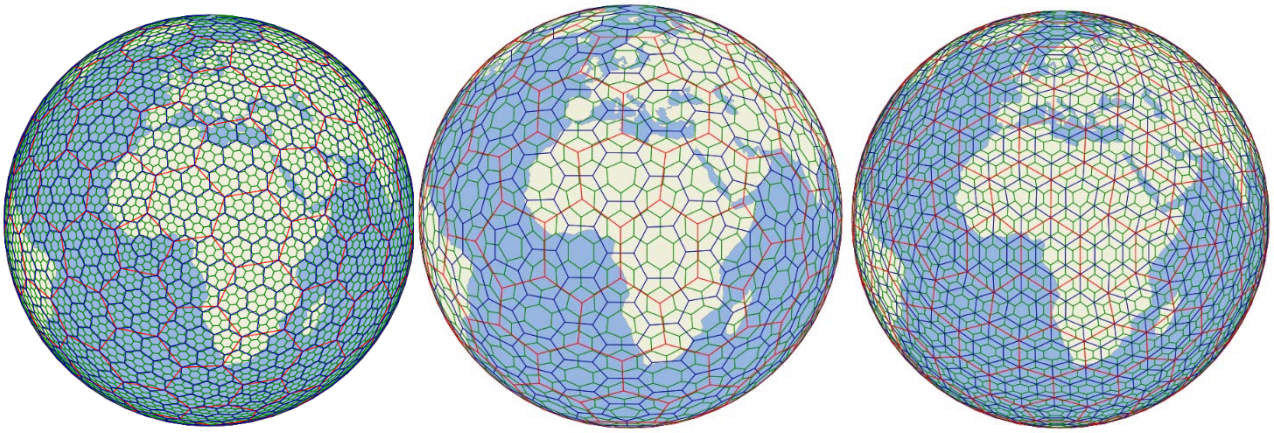

*Figure 12. 3 subsequent resolutions of hexagonal refinements (left to right): aperture 7, aperture 3, aperture 4*

However, there are several drawbacks that have been hindering wide implementation of hexagonal patterns. One of them is the fact that hexagons in contrast to the other shapes are incongruent. It is impossible to exactly subdivide a hexagon into smaller child-hexagons or to aggregate smaller hexagons into a larger one (Sahr, 2003). Therefore, well-known quad-tree based hierarchical indexing methods are not applicable for hexagons-based grids. In

consequence, significantly more complicated indexing algorithms need to be implemented. The non-rectangular shape of hexagons also removes the possibility to use common Cartesian coordinates in the grid structure, so special algorithms for this type of indexing are required as well (A. Amiri et al., 2015; Sahr, 2008; Vince, 2006b).

Another drawback of hexagons as a shape for tessellation particularly in the context of DGGS is the impossibility to completely subdivide a spherical icosahedron into hexagons - at each of the vertices of an icosahedron the remaining area results in forming pentagons (Sahr et al., 2004). These pentagons obviously introduce shape distortions and have a different area from the hexagonal cells on the same resolution level. But, there is always 12 pentagonal cells on every resolution level and the difference in area between this cells and normal hexagonal cells is always equal to 1/6. Such constants simplify the handling of this issue by introducing additional conditions in the algorithms. However, this still leads to additional complexity of the algorithms that need to take both pentagonal and hexagonal shapes into account.

*Multi-shape tessellations*

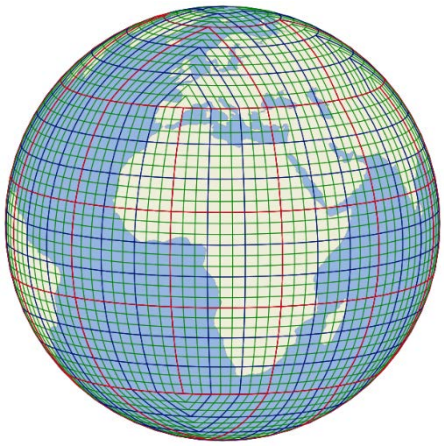

*Figure 13. 3 subsequent resolutions of an aperture 4 multi-shape refinement*

Some approaches to DGGS constructions are going beyond single tessellation shape, which gives unique advantages in indexing and modeling. Gibb et al., (2016) in their rHEALPix DGGS uses 5 different shapes – squares for average latitudes where distortions yet insignificant, skewed squares and triangles in higher latitudes and circles (or caps in terms of spherical geometry) over the polar areas (Figure 13). This allows to achieve equal area aspect for the cells and use all benefits from the algorithms for quadrilateral shapes. Approximately same scheme is proposed by (Zhou et al., 2016) - a pole-oriented DGGS, referred to as the Quaternary Quadrangle Mesh (QQM). The QQM uses semi-hexagon grids for the polar regions and

rectangular grids elsewhere. More sophisticated combination of diamonds-rhombuses, triangles and hexagons were implemented by Jubair et al., (2016) on the base of triangular aperture 4 tessellation of the icosahedron. The possibility of triangular shapes to be aggregated into diamonds or hexagons allows to sample the data not only at cell centroids but also at vertices and edges which gives significant advantage in flows modeling.

**Projection method**

The next step in the construction of a DGGS is to map the chosen partition onto the surface of the sphere or spheroid. Kimerling et al., (1999) defined two basic approaches depending on the selected solid: direct subdivision of the sphere/spheroid or an inverse projection of the subdivided planar faces of the polyhedron onto the sphere/spheroid. The first approach is considered to be more straightforward with constructing initial spherical triangles from intersecting great circle arcs and further triangular aperture 4 subdivision (Gregory et al., 2008; White et al., 1998). For the second approach, a projection method has to be chosen which would guarantee exact mapping of the planar face to the spherical shape so that the sphere is completely tiled with no gaps or overlaps. So called polyhedral projections are used for that. Such projections work by dividing the sphere into regions, each corresponding to a face of the polyhedron, and then employing an existing spherical projection to map points between the spherical region and the planar polyhedron face (Amiri et al., 2015). As with any other single-surface projection, when the faces of a polyhedron are projected onto the sphere, two types of distortions may be created: angular distortion or areal distortion. There is always a trade-off between these two distortions, reducing one type leads to greater distortion of the other (Amiri et al., 2015).

Gregory et al., (2008) describe 4 possible projection methods which satisfy described requirements for common polyhedrons: Gnomonic, Grey-Fuller Dymaxion-based, Snyder equal-area polyhedral projection and Dutton's QTM-based octahedron projection. The choice of the projection is dictated by the desired properties for a particular application – the Snyder projection provides equal areas with shape distortions, Gnomonic projection preserves shapes but gives quite significant variance in areas and Fuller-Gray slightly distorts both aspects. Because areal equality of the cells is the most desired property for statistical or modelling applications Snyder Equal Area projection is widely used in DGGSs (Mahdavi-Amiri et al., 2015; Mahdavi-Amiri et al., 2015a; Sahr et al., 2004).

**Cells indexing method**

Finally, to assign data to grid cells and perform any kind of spatial or statistical analysis it is necessary to have a way of uniquely identifying cells. Not only each cell at each resolution level needs to be uniquely identifiable, but also its location on the actual surface of the Earth and the resolution level ideally should be defined through the index. In order to usefully aggregate or otherwise analyze many cells DGGS indexing methods should support computationally cheap hierarchical neighboring traversal (Amiri et al., 2015). The 3 most common indexing approaches are based on classifications by Amiri et al. (2015) and Bush, (2017a): hierarchy-based indexing, axes-based indexing and space-filling curves or path indexing.

Hierarchy-based indexing in DGGS benefits from its inherited hierarchical structure - the first resolution cells can be initially indexed and then this index can be used as a prefix for the index at the following resolutions recursively (Figure 14). This base index then can be used to define algebraic operations on indices, such as conversion to and from the Cartesian coordinate system or neighborhood finding (Amiri et al., 2015).

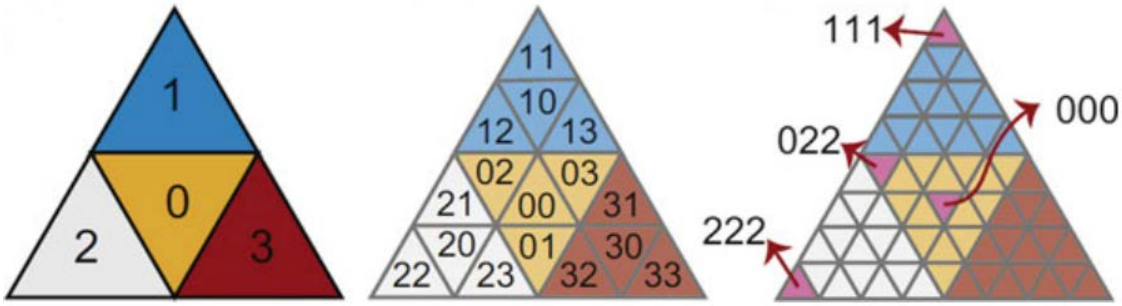


*Figure 14. Hierarchical indexing (Source:* Alderson et al., 2020).

The most common and well-developed hierarchical indexing method is quadtree, but it is only suitable if congruent shapes – such as square or rhombus - are used for partitioning. However, for incongruent shapes such as hexagons there is no straightforward hierarchical relations. It leads to significant challenges for this type of indexing in hexagonal partitions. To tackle the issue, a variety of different sophisticated hierarchy-based indexing methods for hexagonal DGGSs have been developed or are currently under development (Mahdavi-Amiri et al., 2015; Sahr, 2008; Vince, 2006a).

Space-filling curves (SFCs) have been used in many applications, such as compression, rendering and database management. SFC is a path that recursively traverses a space in a pre-defined pattern, addresses locations on its path, and eventually covers the whole space (Amiri

et al., 2015). Some of the common SFCs are Hilbert, Peano, Sierpinski and Morton. Conceptually SFC provides a mapping from n-dimensional space to a 1 dimensional line with ordering indexed elements along the line (Figure 15). Consequently, utilizing a SFC can allow for efficient clustering of spatial data (Sirdeshmukh, 2018).

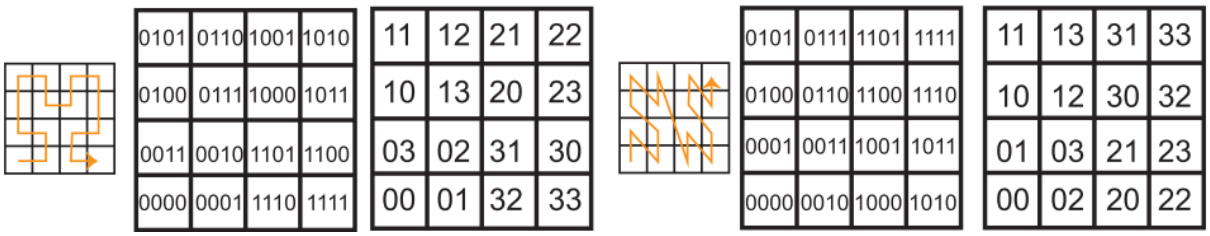


*Figure 15. Indexing by space-filling curves: left – Hilbert, right – Morton (Z) (Source: Amiri et al., 2015).*

The most common example of an axes-based indexing method is the Cartesian coordinate system. A natural way for indexing is to define a set of axes which span the entire space on which the cells lie. Similar to hierarchical indexing, when a refinement is applied to the cells, a prefix is appended to the index indicating the resolution (Amiri et al., 2015). To apply a 2D indexing method on a polyhedron for DGGS, the polyhedron can be unfolded to a 2D domain and the axes defined for the entire 2D domain or each face can be given its own coordinate system (Figure 16) (Mahdavi-Amiri et al., 2015). Again, this type of indexing naturally suits better for quadrilateral shapes of the faces of the base polyhedron and refinement cells. For triangular faces of the icosahedron or octahedron and hexagonal partitions several non-trivial approaches can be applied.

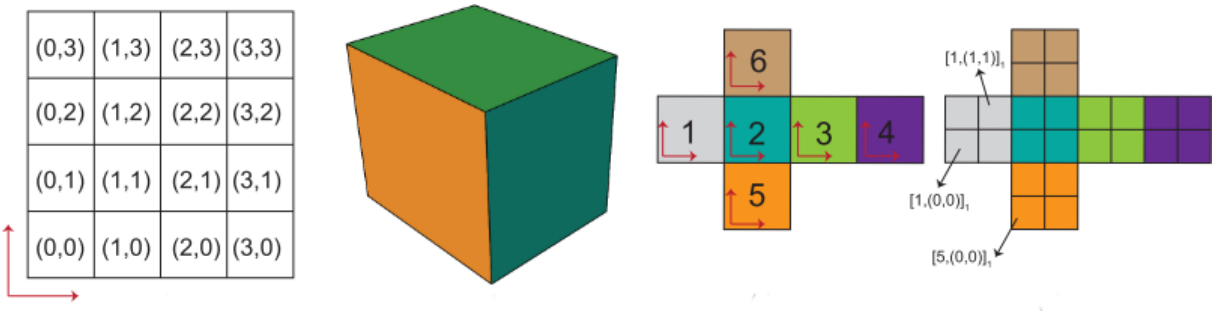


*Figure 16. Axis-based indexing (Source: Amiri et al., 2015).*

Another interesting, straightforward method has been proposed by (Mocnik, 2019) for aperture 3 hexagonal partitioning: In their indexing method each cell ID encodes the geographic coordinates of its centroid (latitude and longitude), the resolution and the shape of the cell (pentagon or hexagon). Such an index method provides human-readable cell IDs from which actual location on the globe or geometry of the cell can be easily derived.

Another possible approach is not using an indexing system beyond applying simple unique integers to all cells – in this case siblings and neighboring relationships as well as spatial queries would purely be based on spatial relations modeled by the means of relational database tables. Such indexing is used by (Adams, 2017) for the global gazetteer called Wãhi. For the hexagonal grids this means six adjacent cell identifiers are stored for each cell, and for triangular grids three adjacent cell identifiers are stored. Hierarchical parent-child relationships between cells at different levels are stored in another table. These relations are established based on spatial overlap (Adams, 2017).

## 1.4 DGGS advantages and limitations comparing to traditional coordinate systems and data structures

Traditional spherical longitude/latitude and planar coordinate systems were developed during the era of paper maps. They are suitable for navigation and spatial analysis on local scale, but not so adequate for geospatial computing and statistical analysis on the global scale (Purss et al., 2019; Sahr, 2011). Global grids based on the geographic coordinate system, such as well-known graticule have many practical benefits. The geographic coordinate system has been used extensively since way before the computer era and is therefore the basis for a wide variety of existing data sets, processing algorithms and computer software as well. Grids based on square partitions are by far the most familiar to users, and they map efficiently to common data structures and display devices (Gregory et al., 2008; Sahr et al., 2004).

However, grid systems created on the base of the latitude–longitude graticule do not have equal-area cell regions, which complicates statistical analysis on these grids. The cells become increasingly distorted in area, shape and inter-cell distance to the north and south from the equator. The areas of extreme distortions appear close the north and south poles where cells are triangles, not squares as they appear on the plane. These polar singularities have forced geospatial community to adapt special grids and projections for the polar regions (Amiri et al., 2015). Attempting to use any of these representations on truly global big data sets exposes their internal geometric weaknesses, which include areas of extreme distortion, singularities, and data under/over-sampling (Purss et al., 2019).

As it was mentioned previously, square grids in general do not exhibit uniform adjacency - each square grid cell has four neighbors with which it shares an edge and whose centers are equidistant from its center. Each cell, however, also has four neighbors with which it shares

only a vertex and whose centers are a different distance from its center than the distance to the centers of the edge neighbors. This complicates the use of these grids for such applications as discrete simulations (Sahr et al., 2004).

Another unique property of DGGSs which frequently discussed in the literature is explicit spatial uncertainty (Bush, 2017a; Robertson et al., 2020; Tong et al., 2013) While all spatial data are collected with some degree of accuracy, vector geometry types have this information only in metadata, which is ignored in further analysis in the most cases. Explicit assignation of spatial uncertainty to DGGS cell resolution provides an easy mechanism to ensure that data do not demonstrate false precision (Robertson et al., 2020). No point can be infinitely accurate, so there will always be an accuracy associated with each point. In a DGGS this accuracy is manifested by the area of the cell (Bush, 2017a).

There is a substantial body of studies which discuss advantages of DGGSs data model from the perspective of computation effectiveness in the era of Big Data. Zhenlong et al., (2020) formulated the most important advantages of DGGS related to Big Data handling: (1) the DGGS provides a single framework for the seamless integration of different distributed global geospatial data from different sources and domains; (2) the DGGS works with high-performance computing to handle big data extremely well because data managed with the DGGS is already decomposed into discrete domains and can be processed in parallel; and (3) by providing a single framework, the DGGS benefits interoperability among different tools and geoprocessing technologies and is a promising solution to build interoperable data systems.

Traditional representation of vector data as tuples of decimal numbers computationally more expensive in spatial analysis algorithms and potentially introduce rounding error (Sahr, 2011). Traditional systems are not well suitable for spatial data storage and interaction on global scale. Majority of algorithms in spatial analysis require input data in projected planar coordinate systems which are either location-specific or introduces distortions if they are global. The global scope of a DGGS immediately means that algorithms defined for one region of the DGGS reference frame become applicable anywhere within the DGGS without necessity to take into account the variances in coordinate systems (Sahr, 2011). If DGGS also built with equal-area partitioning, statistical spatial analysis can be replicated consistently anywhere on the Earth, independent of resolution or scale. Robertson et al., (2020) illustrated how DGGS can provide a single spatial representation for geospatial data. This means that algorithms can be designed on this representation and can therefore incorporate a wide variety of native geospatial data

types. In the examples and case study explored here, we have used polygon, raster, network, and point object data.

Kimerling et al., (1999) formulates some key advantages of DGGSs particularly for environmental datasets. A highly regular global grid would allow us to:

1) Summarize and organize the multiple, non-uniformly spaced measurements

over the globe;

2) Calculate gradients (e.g., for budget calculations);

3) Make comparisons of time-series of globally distributed data (e.g., for detecting climate changes);

4) Make statistically meaningful regional com- parisons of globally distributed data;

5) Compare and combine data sets taken at different spatial resolutions, such as data from multiple satellite measurements and field verification data;

6) Improve the operation of numerical models based on finite difference equations; and

7) Document the precision as well as the location of spatial data on the globe.

## 1.5 DGGS implementations and use cases

There are numerous implementations of different DGGS can be found in scientific literature over the last 40 - 50 years. Although being constructed on a strong theoretical foundation these implementations seldomly are able to find its way outside of particular scientific institution or specific area of implementation towards wider public as a general-purpose spatial model. Mahdavi-Amiri et al., (2015) gives and overview on such state-of-the-art DGGS which impressive variety is based on design options described in previous subchapter. Most common base architectures for majority of described systems are Quaternary Triangular Mesh (QTM), Icosahedral Snyder Equal Area Aperture 3 Hexagonal (ISEA3H), systems based on the cube as basic solid with rectangular congruent refinements and modified latitude/longitude partitions.

More interesting from practical perspective is commercial and open-source general-purpose oriented DGGS-based software products. Such products along with one or another implemented DGGS model also have a wrapper or API of convenient methods which allow to actually work with underlying DGGS and use it for any kind of relevant analytical spatial tasks. Such APIs are usually offered in several programming languages which makes it easier to use these

products for variety of applications. The further focus of this work will be mostly on such products, following subchapter is dedicated to the description of the most popular of them based on technical specifications and usage experience.

# 2 Methodology

## 2.1 Open-source DGGS implementations

### 2.1.1 Uber H3

Uber Technologies Inc, (2018) developed Hexagonal Hieratical Geospatial Indexing System (H3), grid system for optimization of ride pricing and dispatch, for visualizing and exploring spatial data. H3 is internally used in different business processes to analyze geographic information to set dynamic prices and make other decisions on a city-wide level. In 2018 Uber open sourced H3 basic C++ library and JavaScript bindings which was followed by its rapid adoption in geospatial community. Up to date there are adoptions of H3 library for all popular programming languages such as Python, R, C#, Java, JavaScript and data base management systems like BigQuery and PostgreSQL.

The underlying DGGS of H3 library is based on aperture 7 hexagonal tessellation of the icosahedron. The orientation of base icosahedron is chosen based on the business specifics of the company – following Fuller's icosahedron orientation all vertices are placed in the water to avoid pentagonal distortions in the areas of company's main interest – cities. Choice of hexagons as a partition shape also dictated by business logic – hexagons due to uniform adjacency are better suitable for modeling of the movements through the grid. Chosen projection method – Gnomonic projection of the faces of icosahedron assumes no shape distortion but quite significant distortions of areas, which was shown by Kimerling et al., (1999). Two indexing methods are used – axis-based for indexing cells on planar faces of the icosahedron and hierarchy-based, which is easily reachable in aperture 7 hexagonal partition due to approximately congruent resolutions. Combination of two indexing schemes allows flexible and computationally efficient methods for neighboring and hierarchical traversing (Uber Ingenering Inc, n.d.).

The library and all implementations to other languages have a set of convenient methods to work with H3. The methods provide conversions to and from geographic coordinates, hierarchical and neighboring traversing, some basic proximity analysis and some auxiliary methods for getting the information on underlying DGGS structure.

### 2.1.2 Google S2

Another example of well-developed and widely used open-source software with DGGS core in it is S2 (Google groups, n.d.). Although developed by Google engineers, it is not stated

anywhere in specifications how this technology is being used in internal business processes. The core library is written in C++, bindings available for Python, Java and Go, unofficial port also exists for JavaScript. S2 is not only DGGS implemented in the software package it is a library for operations on spherical geometry that aims to have the same robustness, flexibility, and performance as the very best planar geometry libraries (Google groups, n.d.). For the scope of this work only DGGS part of the library's API has been investigated.

S2 uses the cube (the Earth Cube) as a base polyhedron oriented in a way that centroids of the two faces located at the poles. There is no technical information on type of projection which is used to map the faces of the cube to the sphere in documentation. However, some properties of the projection will be discussed later in this work. Obviously, chosen partition shape for the cube is square with aperture 4 refinement. For indexing Hilbert space-filling curve is used to recursively index the cells for all resolutions on 6 faces of the initial cube.

Apart from numerous methods for spherical geometry operations S2 API has a set of convenient methods to work with underlying DGGS for grid traversing and conversion to and from geographic coordinates.

### 2.1.3 RiskAware OpenEAGGR

Open Equal Area Global Grid (OpenEAGGR) library developed and open-sourced by RiskAware Ltd in 2017. The OpenEAGGR software library is an implementation of a Discrete Global Grid System (DGGS) which models the Earth's surface as a network of equal area cells (Bush, 2017a). The library is written in C++ with available APIs for Python, C and Java. Also available as plugins for PostgreSQL and Elasticsearch database management systems.

The library currently has two DGGS implementations available both built on an icosahedral base using the Snyder Equal Area projection (ISEA). ISEA4T extends the globe and projection to use a hierarchical equilateral triangle grid with aperture 4. ISEA3H uses a hexagonal grid of aperture 3 with offset coordinate cell indexing (Bush, 2017b). The API has a functionality for the grid traversing, conversion to and from geographic coordinates and some methods for spatial analysis (intersections, rudimentary proximity)

Currently only ISEA4T implementation is fully functional, ISEA3H version has some known issues with hierarchical traversing.

### 2.1.4 rHEALPix

Developed by Landcare Research New Zealand rHEALPix (rearranged Hierarchical Equal Area isoLatitude Pixelization) DGGS and software package has strong scientific background and mentioned in many scientific publications. This DGGS is based on HEALPix system by Krzysztof M. Górski from Theoretical Astrophysics Center in Copenhagen used by astronomic community for measurement and interpretation of cosmic background radiation (Bowater & Stefanakis, 2018).

The rHEALPix DGGS constructed as a mapping of an ellipsoid on a cube, followed by a symmetric hierarchical partitioning of the cube faces, followed by the inverse mapping of the result back onto the ellipsoid (Gibb et al., 2016). As a shape for partition 4 variations are used depending on the latitude - the quad cell (quadrilateral), the dart cell (triangular), skew-quad cell (quadrilateral), and cap cell (circular). All 4 shapes are congruent and form an aperture 9 refinement. Projection method in the system is rHEALPix projection which preserves area, although partition cells non-conformal the area is equal through the resolution level (Gibb et al., 2016). As a base for indexing Z (Morton) space-filling curves are used for every face of the cube.

The rHEALPix DGGS available as a Python package, no adaptations for other programming languages exist for now. The package provides basic functionality like conversion to and from geographic coordinates and grid traversing.

### 2.1.5 DGGRID

DGGRID is an open-source command-line application written in C++, developed primarily by Kevin Sahr, Southern Oregon University. In contrast with previously described software products DGGRID focus on construction of icosahedral DGGSs rather than usage of particular one. The functionality allows to manually define set of design parameters discussed previously or use predefined set. The choice of the base polyhedron is limited to the icosahedron only, orientation can be set manually. For the partition shape all shapes which are able to tesselate an icosahedron can be chosen – hexagons, triangles, diamonds. Available projection methods are Icosahedral Snyder Equal Area or Fuller-Gray.

DGGRID functionality allows to export generated DGGS to standard GIS formats one resolution per run covering whole globe or defined area, binning points into DGGS cells and

perform very basic neighbors and parent-child traversing. To fully utilize the potential of the tool some additional development of convenient wrappers is required.

## 2.2 Development of standard APIs extension scripts for DGGSs libraries

To conveniently work with DGGS software products investigated in this work several sets of helper scripts which utilizes and extends native APIs have been developed. All discussed libraries have Python interfaces, so the scripts are written as Python modules, separate for each individual software library. Every module has similar functionality, but specifics of API implementation had to be taken into consideration during programming. The following part is the description of developed methods, the source code can be found in a GitHub repository (https://github.com/liquidsun/dggs_work).

*Local covering with DGGS cells*. Get DGGS cells indexes of given resolution which cover given latitude/longitude extent as a Pandas DataFrame.

*Global covering with DGGS cells.* Get DGGS cells indexes of given resolution for the whole globe as a Pandas DataFrame.

*Cells geometry generation.* Create cells polygons or cells centroids for given cells indexes as Shapely geometry column of a Geopandas Geodataframe

*Raster data ingestion into DGGS cells.* Assign raster pixels values to DGGS cells at given resolution as a new column in given Pandas Dataframe

*Vector (points) data ingestion into DGGS cells.* Assign points attribute values to DGGS cells at given resolution as a new column in given Pandas Dataframe

*Vector (polygons) data binning into DGGS cells.* Assign polygons attribute values to DGGS cells at given resolution as a new column in given Pandas Dataframe

*Aggregation of cells with numeric/categorical data to coarser resolution (downsampling) and disaggregation to finer (upsampling).* Assign aggregated values from children cells to parent cell or value from parent cell to children. Return Pandas DataFrame with aggregated or disaggregated cells.

*Export to Geojson/geopackage/PostgreSQL or any other format.* Set of export methods to be able to export DGGS cells geometry or cells indexes with ingested data to PostgreSQL database, any GIS format or exchange format

Apart from native DGGSs libraries the following Python packages have been used:

- Geopandas, Shapely, PyProj, Fiona for vector data processing;
- Rasterio for raster data processing;
- PySAL for spatial analysis and classification of numeric ranges;
- Pandas for tabular data processing;
- Matplotlib, Seaborn and GeoViews for visualizations.

Almost the whole further analytical and data manipulation work has been conducted in Jupyter Notebooks which also available in the repository (https://github.com/liquidsun/dggs_work), so the results are reproducible for any input data and DGGS implementation discussed in this thesis.

## 2.3 Comparison of geometric properties of open-source DGGS implementations.

Numerous studies have been conducted on comparison of geometric properties of different global grids (Gregory et al., 2008; Kimerling et al., 1999; Lei et al., 2020; White et al., 1998). In this work some of the properties of underlying DGGSs of open-source software products described in the previous chapter will be compared. These DGGSs implementations have different geometric properties due to the differences in DGGSs structure. For the purposes of this work cells areal equality and cells compactness has been chosen for the evaluation of an open-source DGGS library as the most important for environmental datasets with the main focus on spatial statistics.

The ideal comparison of global grid geometrical properties would consist of computing surface area, compactness, and other metrics at each level of grid resolution down to the spatial resolution corresponding to the smallest grid cell expected to be used for global analysis (Kimerling et al., 1999). Following this suggestion, metrics computation for the comparison was conducted in two different ways. For comparison of the chosen metrics dynamic throughout DGGS resolution levels this metrics were calculated for the whole set of cells on each resolution up to cell area of 1 square km. For visual evaluation of the metrics' spatial distribution and location of distortion anomalies fine enough (up to 400000 cells) resolution level has been chosen for each DGGS. The metrics then were calculated for every cell and results stored in cell's attributes together with an index and a geometry for further visualization.

In order to compare DGGSs with different properties these metrics' values should be standardized to a common scale. Following Kimerling et al. (1999), to check equal area criteria

all cells areas were normalized by dividing it by an average area of all cells. Ideally, for truly equal area DGGS implementation the value should be as close to 1 as possible. For area comparison between resolution levels standard deviation and range were calculated.

Compactness is calculated as perimeter to area ratio. For calculation of standardized compactness methodology from White et al. (1998) was used. The metric is defined as Zonal Standardized Compactness (ZSC) - the ratio of cell perimeter to cell area standardized by dividing this ratio by the highest value of compactness obtainable on the sphere, that is, by the perimeter to area ratio of a spherical zone, bounded by a small circle, having the same area as the given cell. Computationally, this measure reduces to equation 1,

$$ZSC = sqrt(4\pi a - a^2/r^2)/p \qquad (1)$$

where *a* is the cell area, *p* is the cell perimeter, and *r* is the radius of the sphere. The resulting values is then dimensionless numbers between 0 and 1 with more compact shape closer to 1.

To calculate described metrics, analyze and visualize results following workflow of operations has been conducted for every studied DGGS library.

1) Generate global coverage cell IDs for a sufficient resolution
2) Generate cells polygons for these IDs. All libraries output cells geometry in WGS84 coordinate reference system
3) Clean invalid cells geometry. Due to specifics of Shapely library, polygons which cross $180^o$ meridian got invalid geometry, and so must be filtered out.
4) Calculate cells area and perimeter in planar units. To avoid complicated computations on the sphere cells polygons had to be projected from WGS84 to planar coordinate system. To get metrics' precision higher than global equal-area projections allow, every cell has been projected individually with Lambert Azimuthal Equal Area projection using cell's centroid as an origin of the projection. With an individual cell area around 200 – 250 square kilometres shape distortions caused by the projection are neglectable for the current study. After being projected, cell area and perimeter were calculated in planar units and stored as additional attributes.
5) Calculate standardized area and Zonal Standardized Compactness for every cell as described above and store the values as additional attributes
6) Visualize results as maps and graphs

## 2.4 Vector/raster values ingestion. Generalization data loss comparison

### 2.4.1 Vector/raster data ingestion

The most important functionality for any DGGS implementation is data ingestion (sampling). For ingestion of raster data several strategies can be used. The most straightforward approach is to use coordinates of raster pixels centroids to get corresponding DGGS cells indexes on desired resolution. Then values can be aggregated by cell indexes using any desired statistical aggregation method (mean, max, min etc in case of numerical or mode in case of categorical values).

If any manipulations with initial raster is not desired, first option is to sample raster dataset using cells centroids. This method allows avoiding expensive polygonal geometry generation if there is a possibility to generate only cell centroids in particular DGGS library. The downside of this method is necessity to assure that sampling DGGS resolution will match resolution of the ingested raster (Figure 17). Coarser DGGS resolution will lead to data loss due to less frequent sampling points (cells centroids) distribution than raster pixels. In case of significantly finer resolution there will be an unnecessary overhead during sampling process and loss in performance of the algorithm. Simple automatic resolution detection algorithm suggested and implemented for some of the scripts. First, input raster pixel size is retrieved from metadata. Then for a range of DGGS resolutions average cell edge size is calculated and compared with raster pixel size. If cell edge size is less than given desired pixel size/edge size ratio then this DGGS resolution will be chosen for the data ingestion. With this ratio about 1/2 or 1/3 all input raster pixels will be sampled by DGGS cell centroids, so no data will be lost.

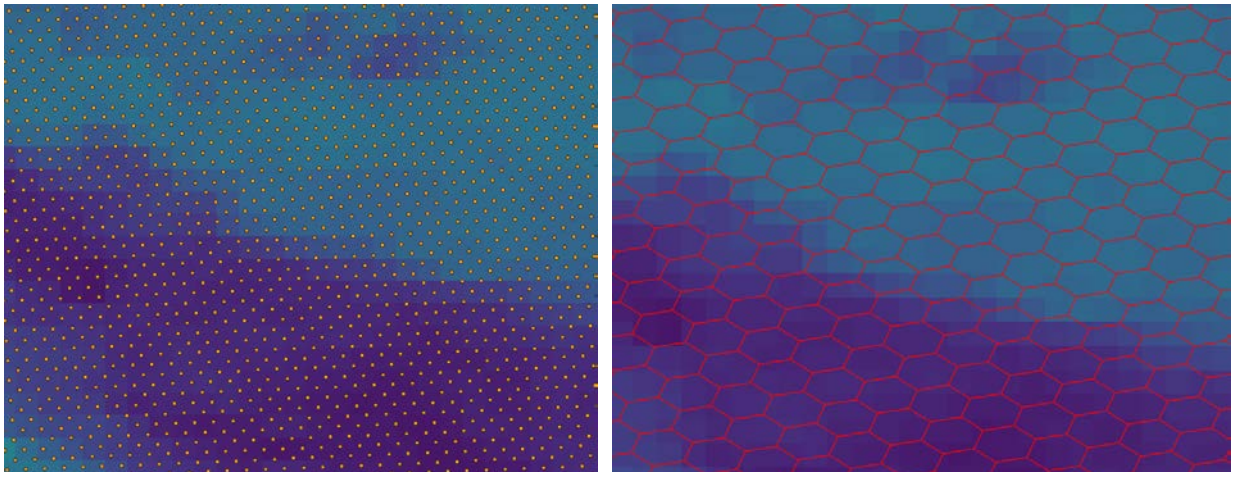

*Figure 17. Raster data ingestion strategies. Left – using cells' centroids, right – using cells' polygons*

Another option is to use cells polygons with any raster zonal statistic algorithms (Figure 17). This method might be more computationally expensive as it is necessary first to construct full

cells' geometry. Constructed vector geometry then most likely will have to be rasterized as zonal statistic algorithms works with raster datasets internally. Significant advantage of this approach is a possibility to use any desired resolution of DGGS for ingestion with any desired statistical aggregation method of raster data into DGGS cell.

Ingestion of vector points is the simplest scenario as all DGGS implementations have methods for conversion of geographic coordinates to cells indexes. The procedure is the same as raster pixels centroids ingestion scenario described above.

Polygonal vector data ingestion is more complicated due to necessity in careful selection of DGSS resolution level to avoid data loss due to undersampling. Basic strategy is to get DGGS cells centroids on manually defined fine enough resolution and use any spatial intersection algorithms to assign desired polygon attributes to cells. To automatically select appropriate resolution following algorithm might be used here. First, calculate an area of the smallest polygon in the input vector dataset. Then, get an average cell area of a range of DGGS resolutions. If cell ratio of polygon area/cell area is smaller than desired value, then corresponding resolution is appropriate for the data ingestion.

### 2.4.2 Aggregation, disaggregation, DGGS hierarchy traversing

Data for ingestion might come from variety of sources of rasters with different resolution and vectors with different spatial accuracy. As a data integration platform, DGGS implementation has to be able to transform ingested data into common resolution. In order to easily switch between resolutions indexing system of underlying DGGS ideally should allow parent-child traversing and appropriate methods implemented in the API. Then moving to coarser resolution is simple aggregation by parent cell index using any desired statistical aggregation method for stored in children cells data values. And moving to finer resolution can be done either by simplistic parent cell disaggregation with setting parent cell data value to all children or using some more sophisticated statistical upsampling methods. This transformation to common resolution can be done either during ingestion of the data or during further manipulations with data on-demand. The choice of the approach depends on database architecture where physical instance of DGGS will be stored.

Crucial property of DGGS for these transformations is the aperture of the refinement, The bigger the aperture number the bigger the areal difference between parent and children cells. Ideally, lower aperture is preferred as it results in more smooth transitions between resolutions

with lower data loss due to aggregation and generalization (Amiri et al., 2013). For every analyzed DGGS effect of transition from the resolution of initial ingestion to several coarser resolutions was evaluated. The metric for the evaluation is the change in area of the distribution of a data value in percentage between initial resolution and coarser resolutions for fixed testing region of Tartu city and surroundings. Datasets of different origins with categorical and numerical data were used: CORINE land cover as a vector dataset and MERIT DEM elevation as a raster dataset.

The datasets have been ingested into every explored DGGS at the resolution 2-3 times lower than native to avoid data loss due to under sampling. So, for elevation raster with a pixel size of 100 meters resolutions with cell area of around 3000-5000 square meters were chosen. Also, to get discrete classes for the continuous raster elevation values were classified into 10 equal intervals. After that, by either provided DGGS hierarchy traversing methods or by simple spatial aggregation if no such methods exist data from initial resolution has been downsampled into 4 subsequent coarser resolutions. It is important to keep in mind that due to different aperture only first resolution on which data has been ingested has a relatively similar cell size. Coarser resolutions will be different in the cell sizes depending on an aperture number. During the aggregation process areas of CORINE land cover and elevation classes has been calculated. Further, the absolute difference in percent between area values on every resolution and areas for the same classes from the initial datasets were calculated. Finally, for DGGSs libraries comparison mean value of all differences for all land cover or elevation classes has been used.

## 2.5 DGGS-based data cube construction

For data ingestion into DGGS, further analysis and visualization open access global environmental data was used. As testing area territory of Estonia is chosen due to availability of high-quality data and computational limitations of used hardware to process global datasets. Following data sources were used:

1) CORINE Estonian land cover data (https://land.copernicus.eu/pan-european/corine-land-cover)
2) MERIT DEM as elevation data (http://hydro.iis.u-tokyo.ac.jp/~yamadai/MERIT_DEM/)
3) Estonian soil map as soils data (https://geoportaal.maaamet.ee/eng/Spatial-Data/Estonian-Soil-Map-p316.html)
4) CHELSEA temperature and precipitation data (http://chelsa-climate.org/)
5) WorldPOP population data for Estonia (https://www.worldpop.org/)

Initial datasets have different resolution and data model. Elevation, temperature, precipitation and population datasets are rasters, land cover and soils are vectors. In order to avoid information loss as much as possible DGGS resolution close or lower than a dataset's native has been used for an initial ingestion. Then, using library's methods of hierarchical traversing datasets with finer resolution have been downsampled to one common resolution.

To illustrate practical application of a DGGS for constructing a data cube of environmental data these 6 different datasets has been ingested into H3 DGGS. The choice of this DGGS library is dictated by its popularity, convenient API and JavaScript binding which allows to visualize the cells in the browser WEB-application – Kepler.gl. After constructed, resulting dataset is a simple table with one column contains DGGS cell indexes and other columns with different data values. These data values are spatially aligned and can be immediately used for some spatial statistical analysis. Plus, spatial and hierarchical relationships between DGGS cells encoded in cell indexes open a possibility to perform aggregations, disaggregations and neighborhood manipulations without computationally expensive spatial operations. If needed, this tabular data structure can be easily divided into several domain-specific tables with cell indexes and data values of the same type for distributed computations or to store in relational database.

# 3 Results

## 3.1 Cells area and compactness properties of DGGSs in open-source software

The empirical results of the investigation of the DGGSs' cells area and compactness properties are described in the following chapter. For each open-source library the resolution which contains 200 - 400 thousands cells with cell area 1500-2500 km$^2$ in the global coverage has been chosen.

### 3.1.1 Uber H3

The H3 DGGS cells properties are analyzed on the resolution level 4, the total number of cells is 287525.

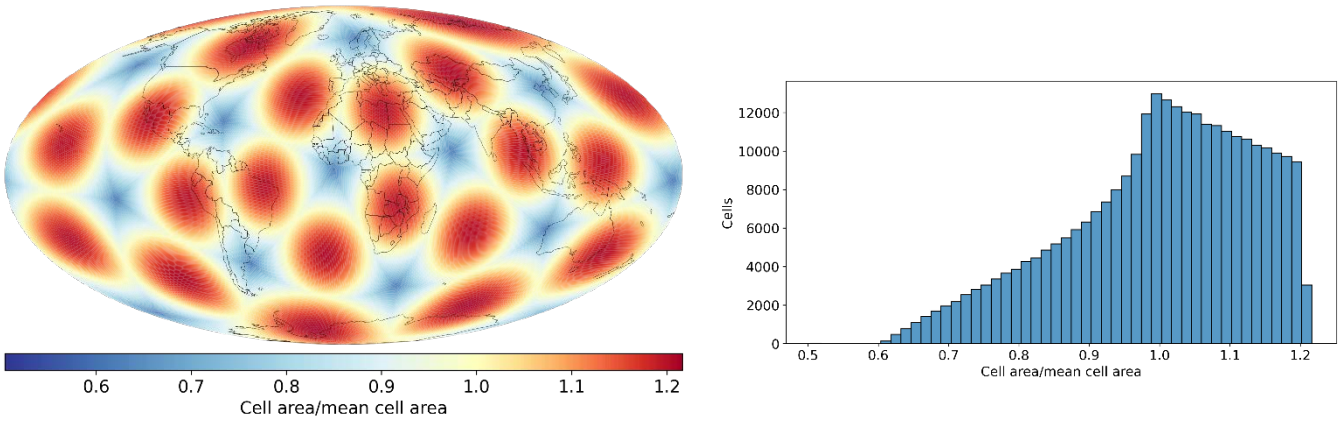


*Figure 18. Left – map of H3 DGGS cells' normalized area in Mollweide projection, right – histogram of the same values*

Looking at the cells' area distribution (Figure 18) it is clear that H3 DGGS cells do not have an equal area across the whole global tessellation. The distortions follow the geometry of the underlying icosahedron – the area changes from central parts of faces towards edges. The biggest difference thus occurs between cells laying closer to the faces' centroids and cells at the vertices – almost 2 times. This is the consequence of the usage of the Gnomonic projection of

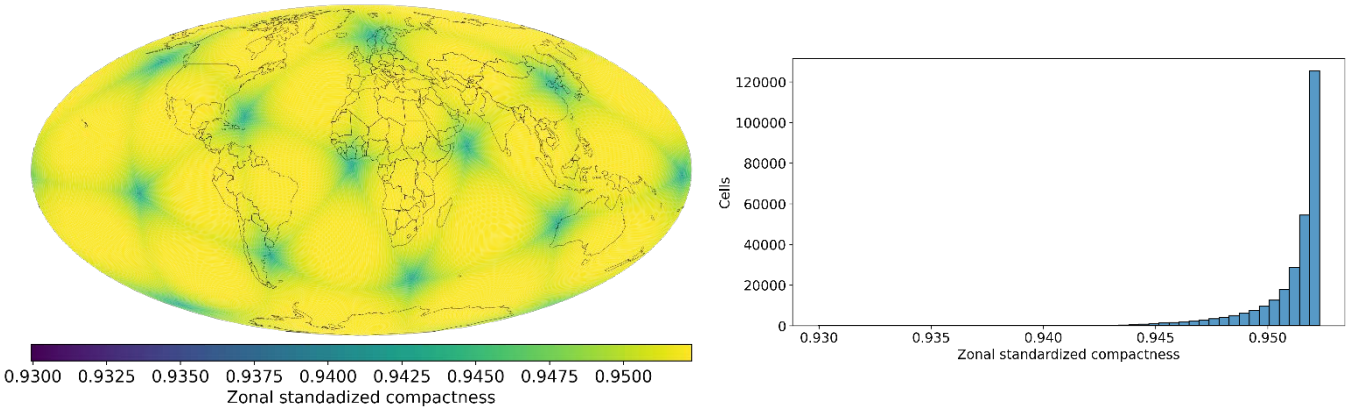


*Figure 19 . Left – map of H3 DGGS cells' compactness values in Mollweide projection, right – histogram of the same values*

the icosahedron's faces. This distribution follows the pattern described by Kimerling et al., (1999). However, cells shape is well-preserved with some minor distortions in compactness at the icosahedron's vertices (Figure 19). These distortions then rapidly even-out towards the faces forming almost a uniform overall pattern.

### 3.1.2 Google S2

The S2 DGGS cells properties are analyzed using the library's resolution 8 with a global coverage of 392703 cells.

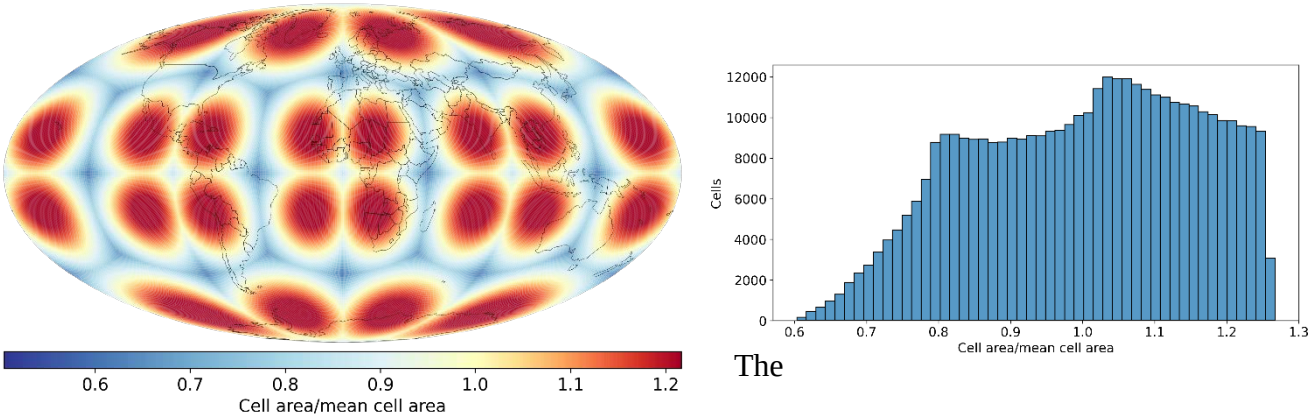


*Figure 20. Left – map of S2 DGGS cells' standardized area in Mollweide projection, right – histogram of the same values.*

The difference in area values in the S2 DGGS (Figure 20) is more than 2 times and with less homogeneity in spatial distribution than H3. The underlying geometry of the base solid, the cube, affects the distribution of area values which can be seen on the picture. The projection type which was used to map the initial tessellation of the cube back to the sphere is not defined in technical specification. But, looking at the distribution of areal distortions it is possible to assume that it is not a Gnomonic projection – every face was subdivided into 4 smaller

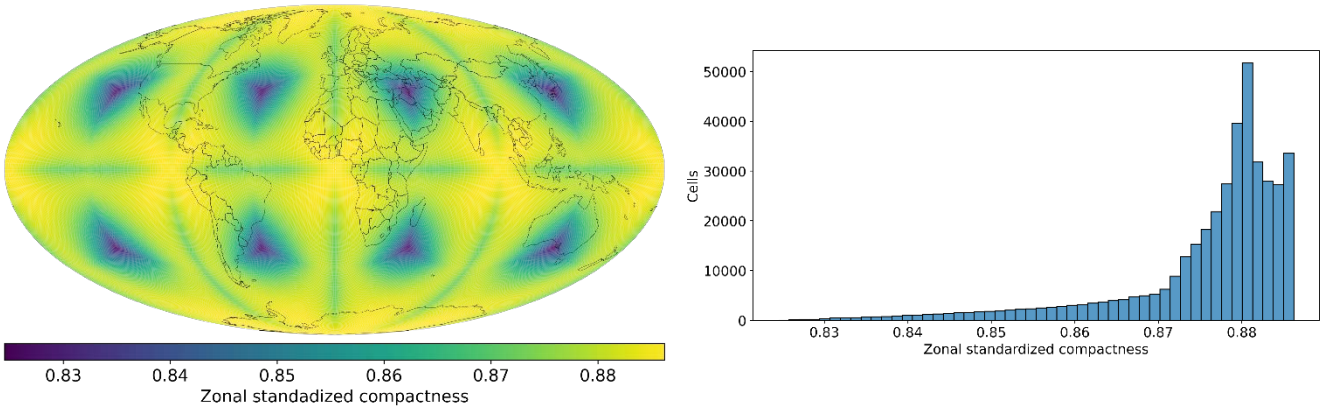


*Figure 21. Left – map of S2 DGGS cells' compactness values in Mollweide projection, right – histogram of the same values.*

segments. The segments seem to be formed by intersection of the Equator, the Prime Meridian and two more meridians an equal number of degrees apart from the prime. In the resulted area

values distribution, the smallest area values related to the edges of the cube's faces and the biggest to the central parts of the segments. The cells compactness is generally lower here due to a rectangular shape and the distribution is less uniform (Figure 21) comparing to hexagonal tessellation of the H3 DGGS. The lowest compactness values related to regions closer to vertices of the cube, the highest to the central parts of the faces.

### 3.1.3 rHEALPix

The rHEALPix cells properties are analysed on the resolution 5, the total number of cells is 352654.

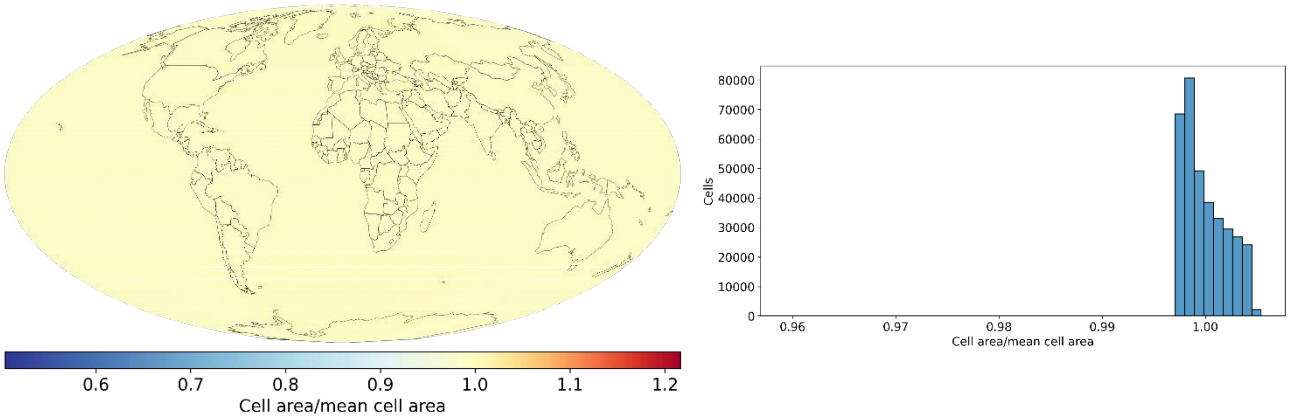


*Figure 22. Left – map of rHEALPix DGGS cells' standardized area in Mollweide projection, right – histogram of the same values*

Although the base solid of the DGGS is the cube, usage of 4 different shapes for a tessellation and equal area rHEALPix projection allow to achieve almost uniform distribution of the global area values (Figure 22). Just some minor (less than 0.01 of normalized area units) differences

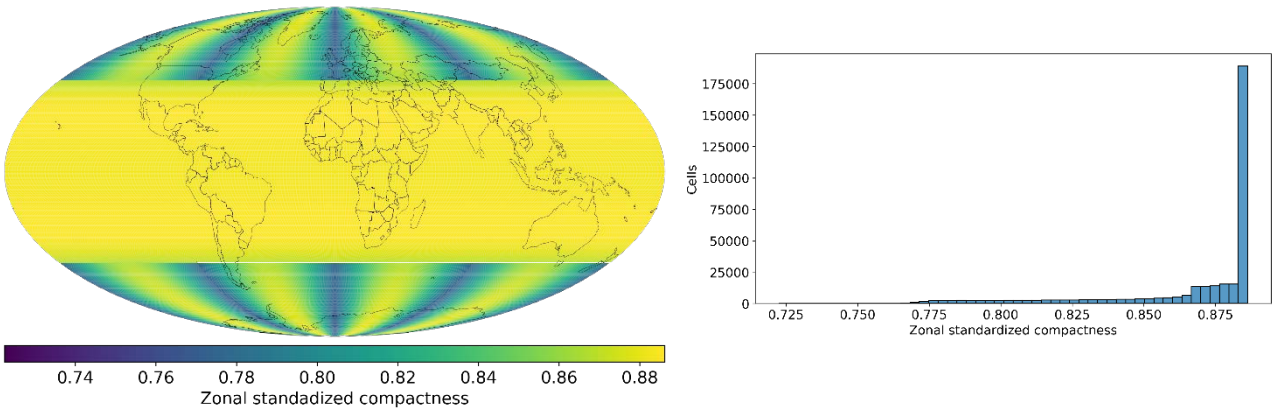


*Figure 23. Left – map of rHEALPix DGGS cells' compactness values in Mollweide projection, right – histogram of the same values*

are still present between average and higher latitudes. The cells compactness values are

allocated based on the distribution of the cells shape – the highest values related to squares in the average latitudes, the lowest – to triangular cells of the higher latitudes (Figure 23).

### 3.1.4 OpenEAGGR

The properties of the ISEA aperture 4 triangular DGGS (ISEA4T) version of the OpenEAGGR library at resolution 7 was analyzed. Total number of cells in global tessellation at this resolution is 327680.

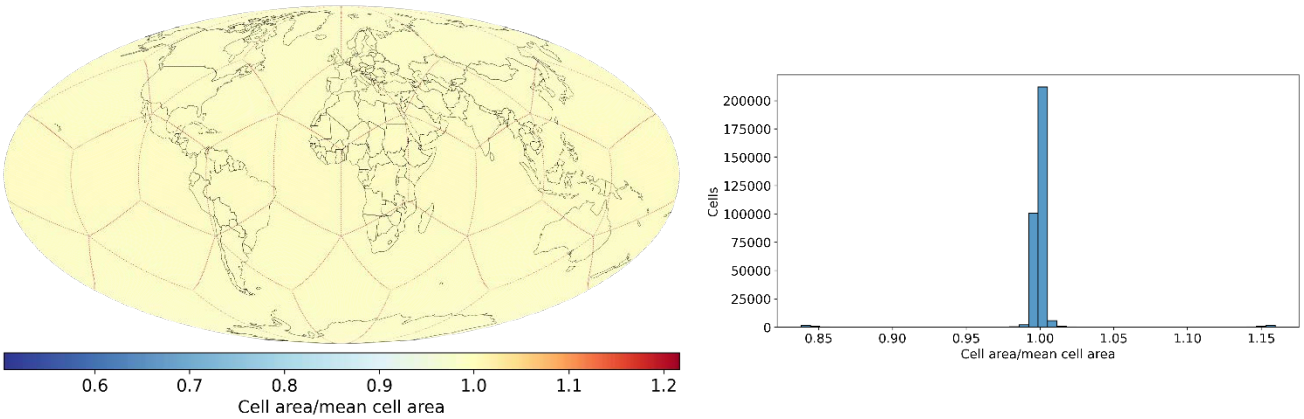


*Figure 24. Left – map of OpenEAGGR DGGS cells' standardized area in Mollweide projection, right – histogram of the same values*

Figure 24 clearly illustrates uniform distribution of cells' area values provided by the ISEA projection. Some observable outliers might be the issues of the particular implementation in the OpenEAGGR library or artifacts of the cells generation and area calculations processes. Lower

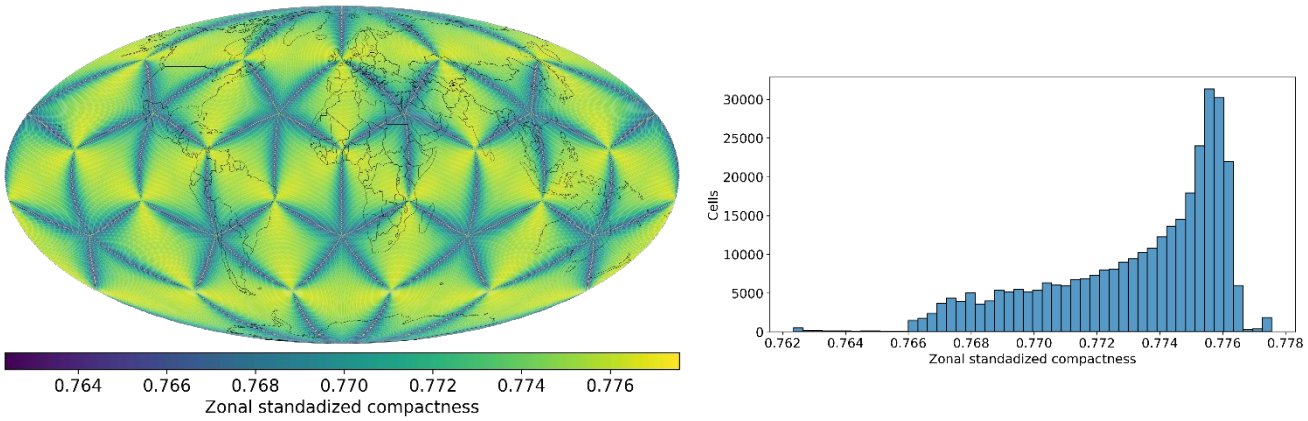


*Figure 25. Left – map of OpenEAGGR ISEA4T DGGS cells' compactness values in Mollweide projection, right – histogram of the same values*

area values are related to the vertices of the icosahedron, higher values – to edges of its faces. Significantly lower compactness values comparing to previously described DGGSs (Figure 25) are result of triangular shape of the cells of the tessellation. Specific irregular pattern of the

values' distribution is common for ISEA projection – shapes are distorted in order to preserve area. On each face of the icosahedron the shape distortion is distributed along the lines which connect the face's centroid with its vertices.

### 3.1.5 DGGRID

As it was mentioned previously, the DGGRID tool can be used to generate various DGGSs with an icosahedron as the base solid with flexibility in selection of other basic parameters. Area and compactness properties have been calculated for all pre-set types available in the tool, but only 4 important for the comparison with other libraries described in the following sections.

*DGGRID ISEA aperture 7 hexagonal DGGS (ISEA7H)*

This DGGS type is close to Uber H3 with a difference in projection type – ISEA instead of Gnomonic projection. The properties were analyzed at resolution 5 with 167636 cells in the global coverage.

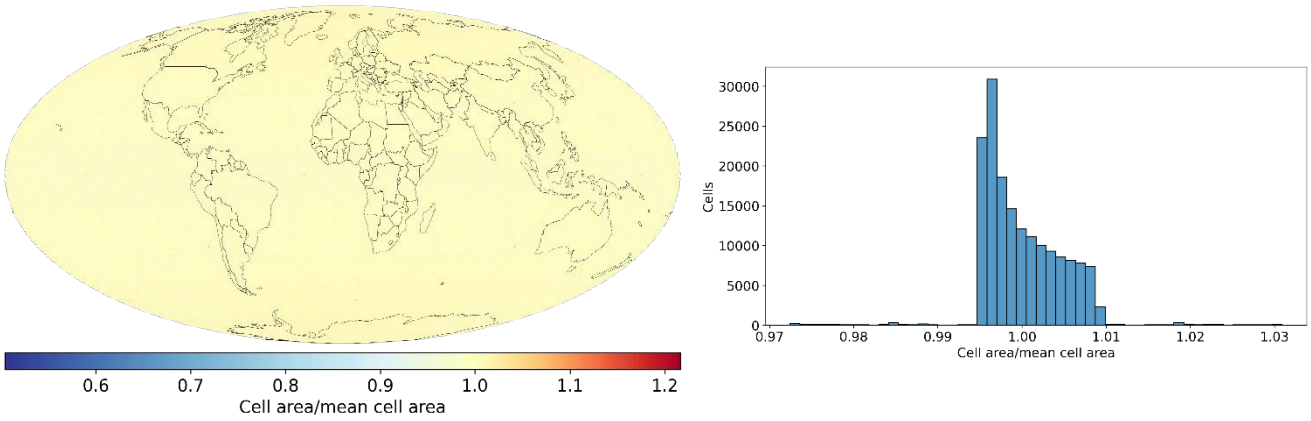


*Figure 26. Left – map of DGGRID ISEA7H DGGS cells' standardized area in Mollweide projection, right – histogram of the same values*

Figure 26 illustrates close to uniform distribution of area values with some outliers. The outliers with values lower than the mean are related to pentagon-shaped cells at the icosahedron's vertices. The outliers with higher values most likely caused by peculiarities in the Python libraries which were used for geometry generation and area calculations. But compactness has the same ISEA distortion pattern (Figure 27) as already has been demonstrated above.

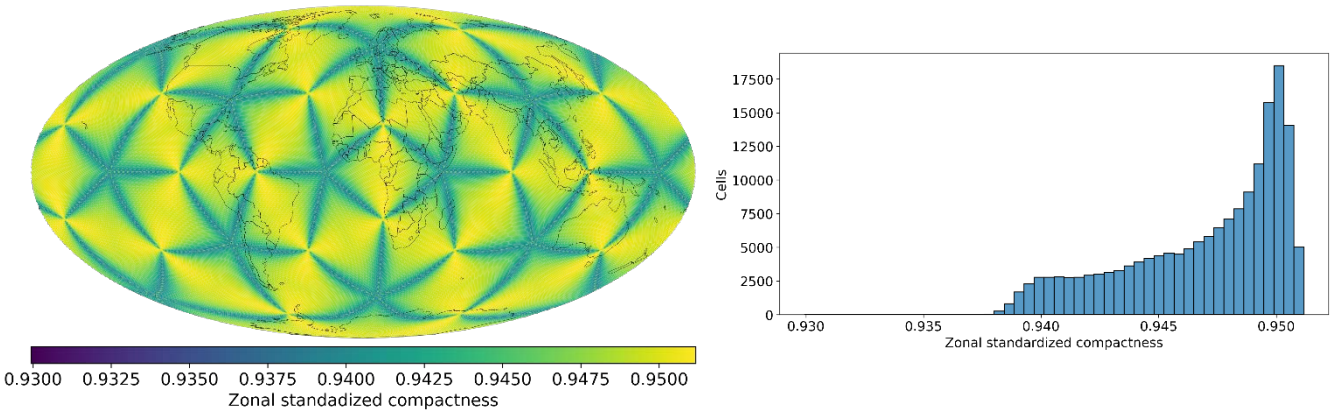


*Figure 27. Left – map of DGGRID ISEA7H DGGS cells' compactness values in Mollweide projection, right – histogram of the same values*

### *DGGRID Fuller-Gray aperture 7 hexagonal DGGS (FULLER7H)*

DGGRID FULLER7H DGGS is another variation of aperture 7 hexagonal partition with different set of properties which is a consequence of Fuller-Gray projection method. The properties were analyzed at resolution 5 with 167636 cells in the global coverage.

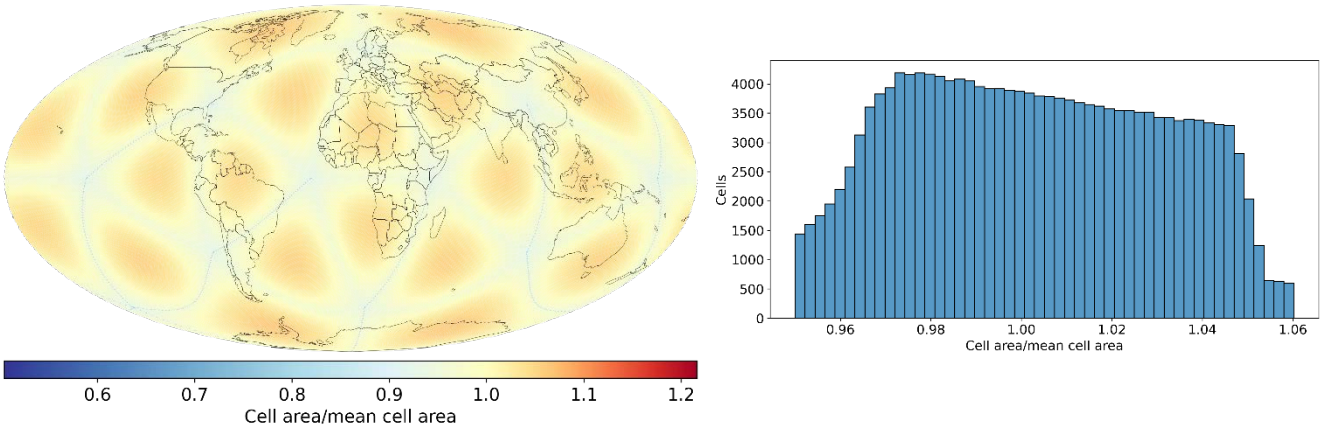


*Figure 28. Left – map of DGGRID FULLER7H DGGS cells' standardized area in Mollweide projection, right – histogram of the same values*

Figure 28 illustrates normalized area values distribution pattern similar to Uber H3, but range in values is significantly lower and the distribution closer to uniform. Lower area values concentrated at vertices and edges of the icosahedron's faces, higher values at central parts of faces. Distribution of compactness values (Figure 29) is more uniform comparing to ISEA variant but still lower than with Gnomonic projection. Thus, these results support the fact that

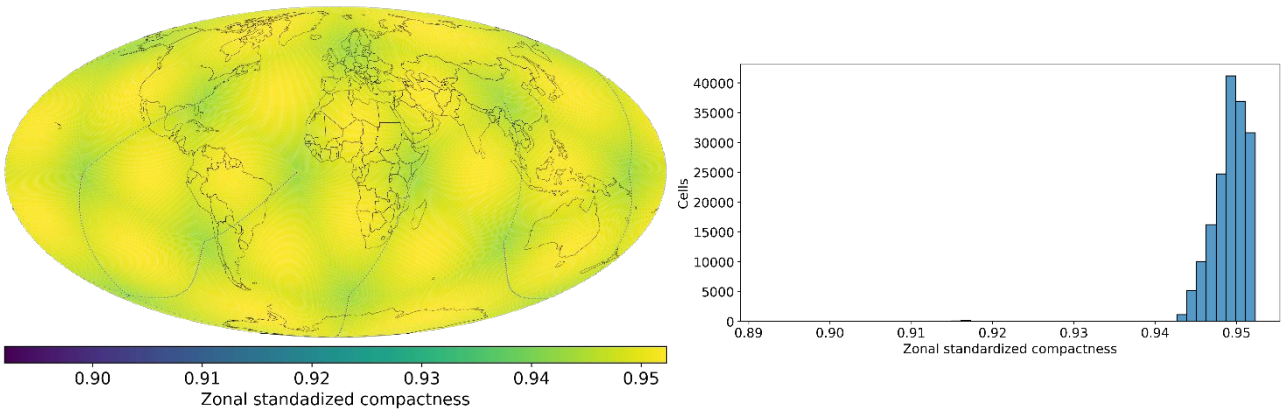


*Figure 29. Left – map of DGGRID FULLER7H DGGS cells' compactness values in Mollweide projection, right – histogram of the same values*

Fuller-Gray projection gives a compromise option in terms of area versus shape preservation between ISEA and Gnomonic projections.

*DGGRID ISEA aperture 4 diamond DGGS (ISEA4D)*

The main difference of ISEA4D DGGS and the reason to include it in this overview is partition shape - diamonds or rhombuses. As it was mentioned before, a diamond is a shape identical in its properties to a square, thus most common and well-developed indexing and analytical algorithms can be utilized for this type of DGGS. The properties were analyzed at resolution 7 with 163432 cells in the global coverage.

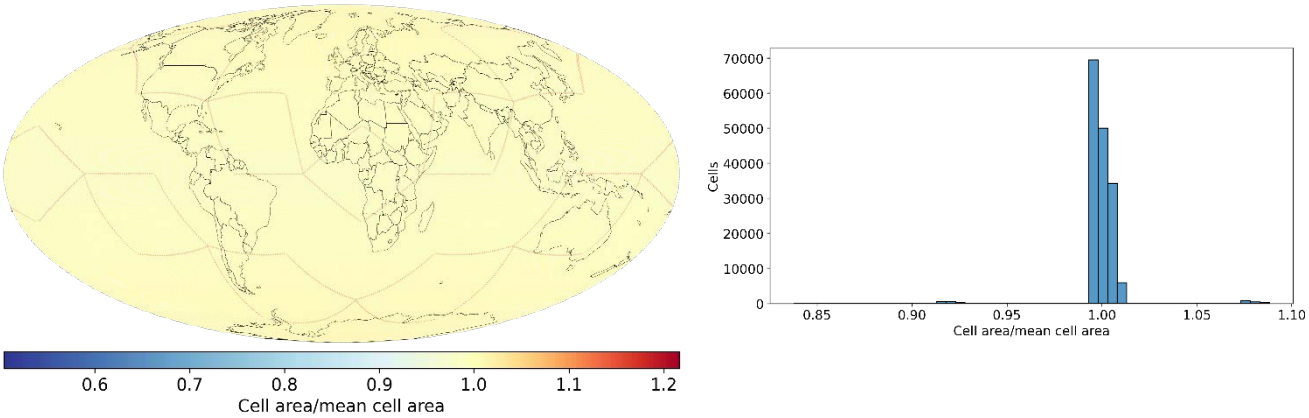


*Figure 30. Left – map of DGGRID ISEA4D DGGS cells' standardized area in Mollweide projection, right – histogram of the same values*

Figure 30 illustrates already well familiar uniform distribution of area values provided by ISEA projection. Still, vertices and edges of the underlying icosahedron tend to be regions of maximal distortions where all outliers are concentrated. Compactness values distribution has quite

unexpected pattern (Figure 31) which differs from hexagonal and triangular versions of ISEA-based DGGS. Obviously, absolute compactness values are lower than in hexagonal partitions but close to squares.

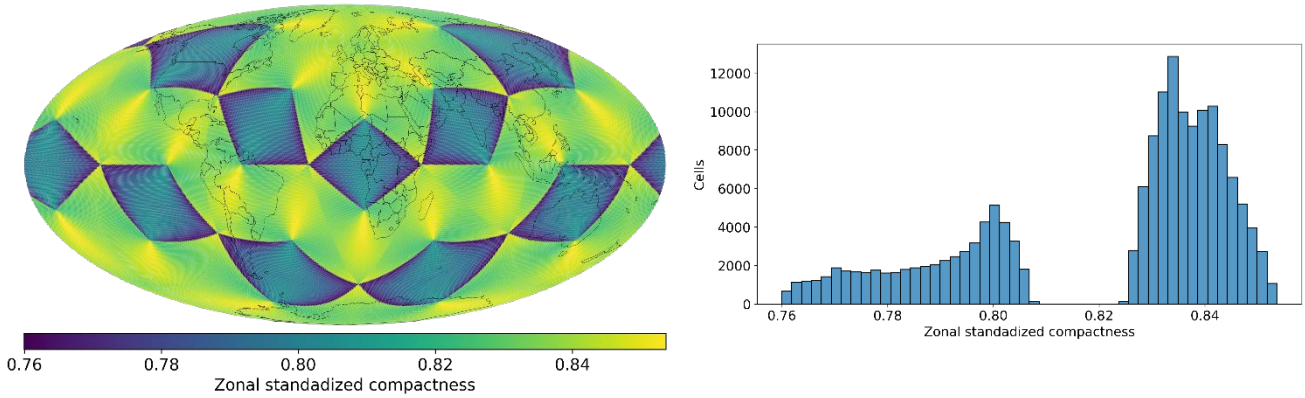


*Figure 31. Left – map of DGGRID ISEA4D DGGS cells' compactness values in Mollweide projection, right – histogram of the same values*

*DGGRID Fuller-Gray aperture 4 diamond DGGS (Fuller4D)*

Fuller4D DGGS version again utilizes diamonds as partition shapes and Fuller-Gray projection. The properties were analyzed at resolution 7 with 163432 cells in the global coverage.

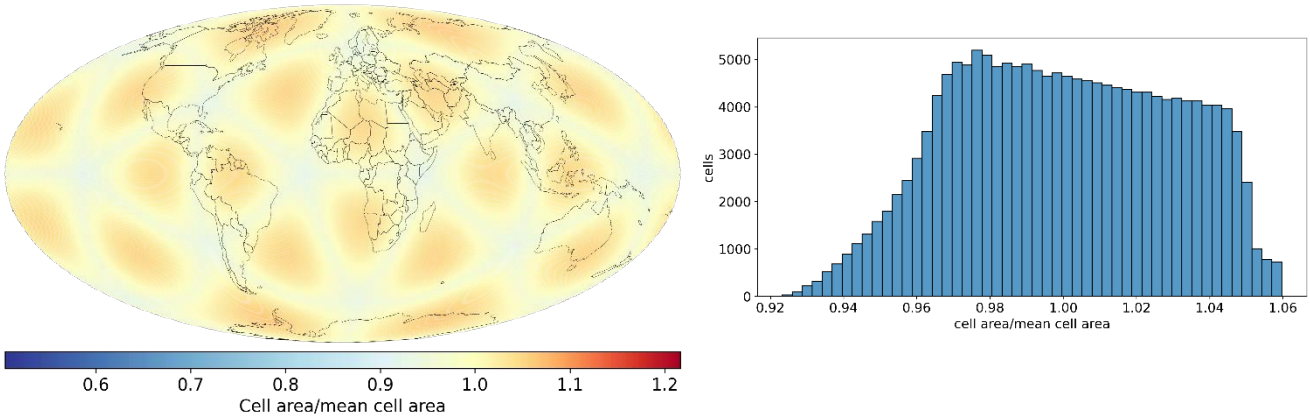


*Figure 32. Left – map of DGGRID Fuller4D DGGS cells' standardized area in Mollweide projection, right – histogram of the same values*

Figure 32 shows non-uniform distribution of area values with same pattern as in ISEA or Gnomonic versions. Maximal values tend to concentrate closer to centroids of the faces, minimal to the edges and vertices of the base icosahedron. Compactness values distribution

(Figure 32) again illustrates different pattern comparing to hexagonal or triangular version of DGGS with Fuller-Gray projection.

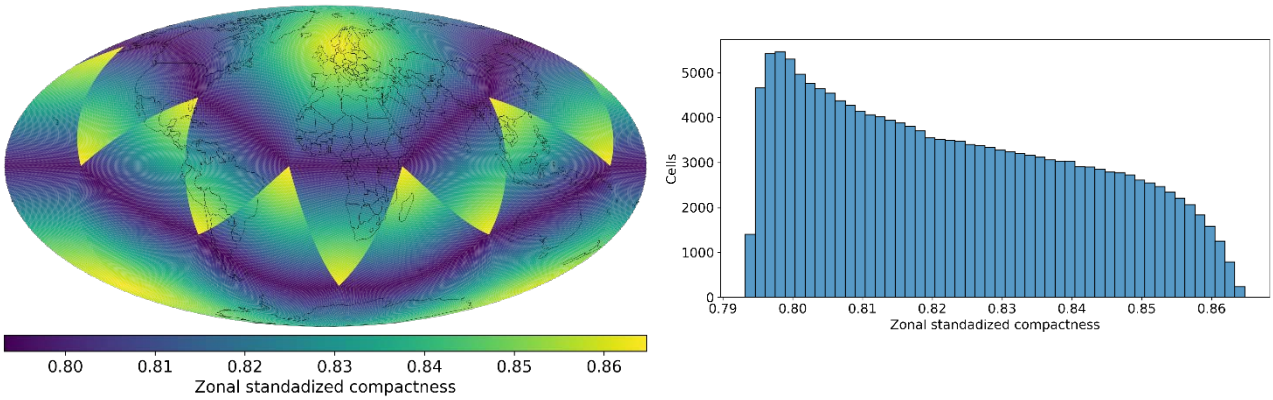


*Figure 33. Left – map of DGGRID Fuller4D DGGS cells' compactness values in Mollweide projection, right – histogram of the same values*

### 3.1.6 Normalized area and compactness values comparison between DGGSs

Normalized values of area and compactness allow to compare these properties for different DGGSs. Figure 34 shows area values for all described DGGSs as a series of boxplot diagrams. The graph clearly illustrates the superiority of rHEALPix and ISEA-based DGGSs in terms of area preservation on global scale. Almost 99.3 % of cells areas are close to the mean value in such DGGSs with a small number of outliers related to either to pentagons in hexagonal partitions or to artifacts of the libraries used for the calculations and cells geometry construction.

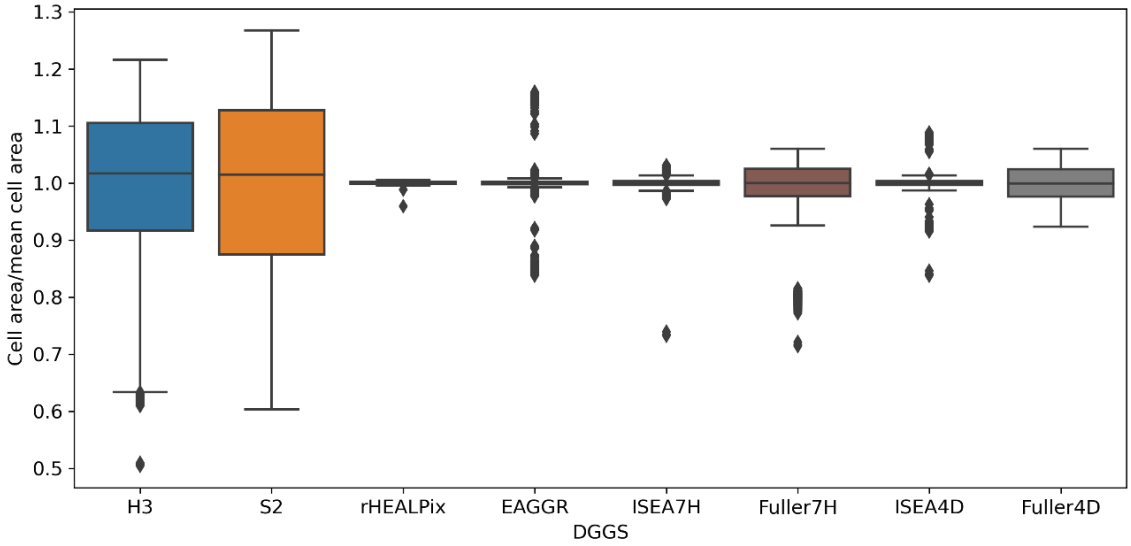


*Figure 34. Distribution of cells' standardized area values in different DGGS*

The same type of graph for compactness values (Figure 35) shows that hexagonal DGGSs have the best compactness while DGGSs with triangular shape of partition are the least compact. Gnomonic projection of the H3 library provides the highest compactness among other hexagonal versions with the smallest range. Although considering its significant area distortions Fuller-Gray projection might be still a better compromise even with slightly lower compactness values and bigger range. Square-based DGGSs have surprisingly higher absolute compactness values and smaller ranges comparing to diamonds-rhombuses.

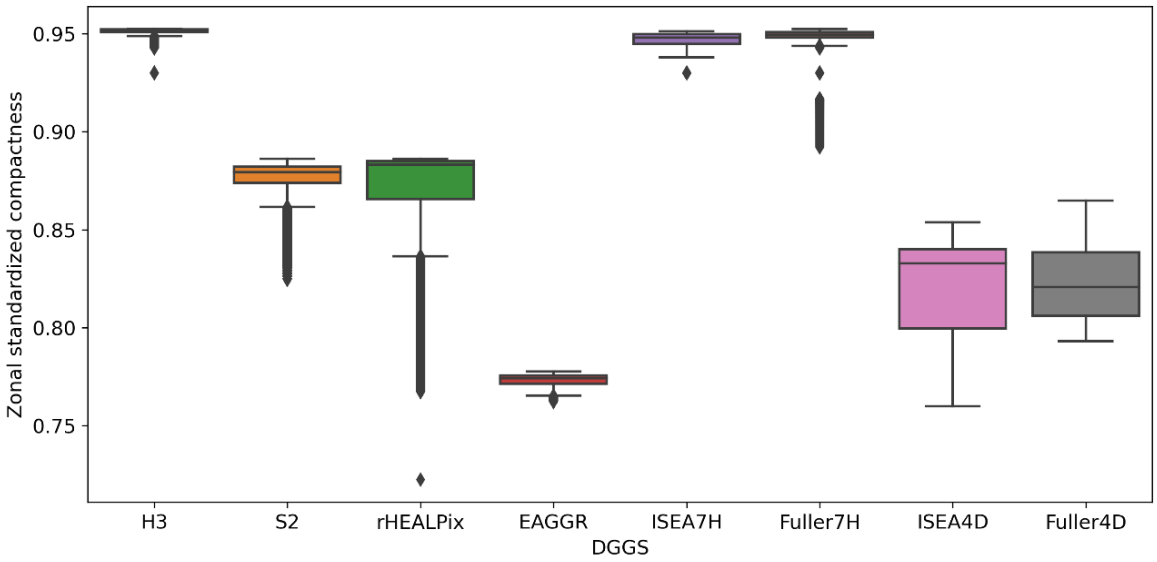


*Figure 35. Distribution of cells' compactness values in different DGGS*

## 3.2 Comparison of a resolution change dynamic between DGGSs

Figure 36 demonstrates the dynamic of mean absolute difference (in percent) of an area of CORINE landcover classes on a test region between initial vector source and same data ingested into different DGGS depending on level of aggregation into coarser resolution.

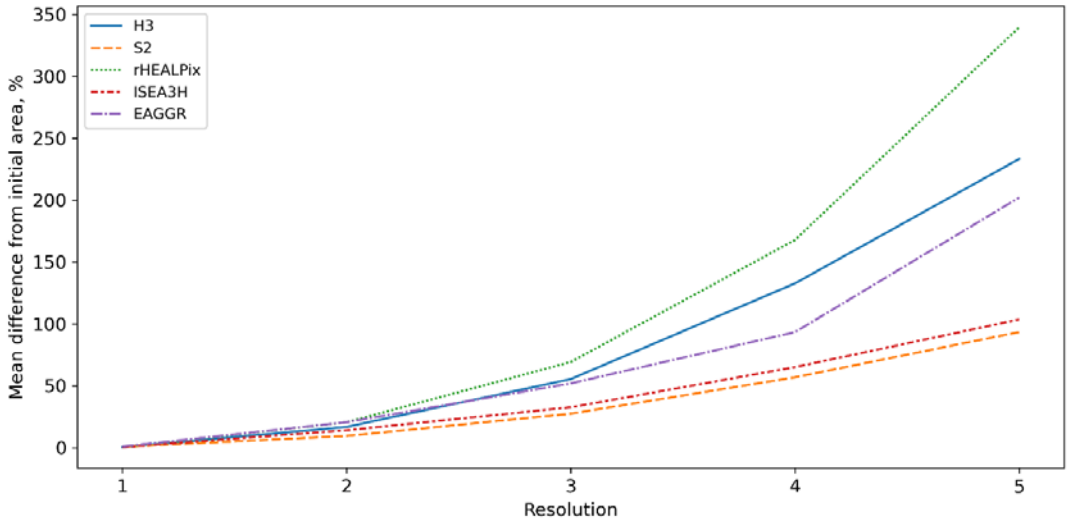


*Figure 36. Change in mean absolute difference in area from initial landcover vector dataset*

It is clearly seen on the graph that DGGSs with smaller aperture provide a smoother transition to coarser resolutions. rHEALPix DGGS with aperture 9 demonstrates the most rapid change in the difference, starting from second coarser resolution from the initial it has the highest value. It is interesting, that ISEA3H DGGS has slightly higher difference values through the resolutions than S2 with aperture 4. Most likely it is the result of incongruent nature of aperture 3 hexagonal partition and not sophisticated enough methods which has been used for transition to coarser resolutions to approach this feature.

Figure 37 shows the dynamic of mean absolute difference of an area of the MERIT DEM raster with values classified into 10 equal interval classes and same data ingested into different DGGSs. Not so clear and straightforward picture is seen here in a relation of an aperture value to a difference in area. Although rHEALPix with aperture 9 shows the most rapid change, position of H3 with aperture 7 on lowest and second lowest values under aperture 3 and 4 contradicts the logic behind transitions between resolutions with different aperture level. Most likely the reason behind is chosen methodology to compare transitions based only on area difference or its particular implementation for raster data sources.

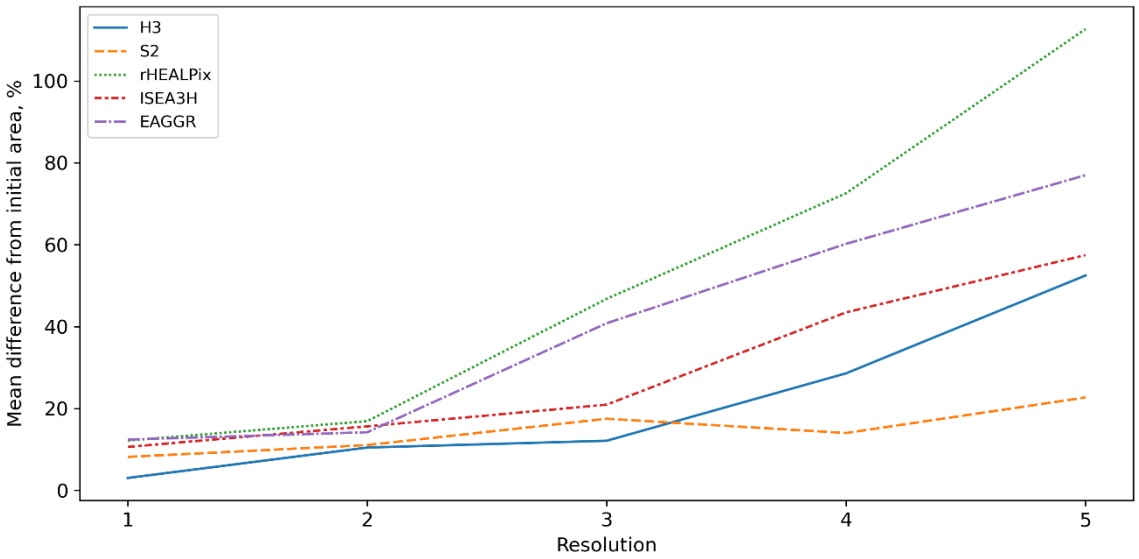


*Figure 37. Change in mean absolute difference in area from initial elevation raster dataset*

Despite some uncertainty in quantitative methods for the comparison, visual analysis of the same resolutions of DGGSs shows expected behavior. Figure 38 shows CORINE land cover classes ingested into studied DGGSs and aggregations into subsequent coarser resolutions. Ingestion and aggregation of 10 classes of the MERIT DEM raster elevation values is shown on figure 39. Here it is clearly seen that with approximately same cells area on initial resolution for every DGGS further aggregation into coarser resolutions causes distortions and data loss faster for DGGS with bigger aperture number. For example, in rHEALPix DGGS 4 and 5 aggregations initial picture is already hardly readable. In contrast, picture from the same aggregation steps in ISEA3H DGGS still have distinguishable general pattern of landcover or elevation classes distribution.

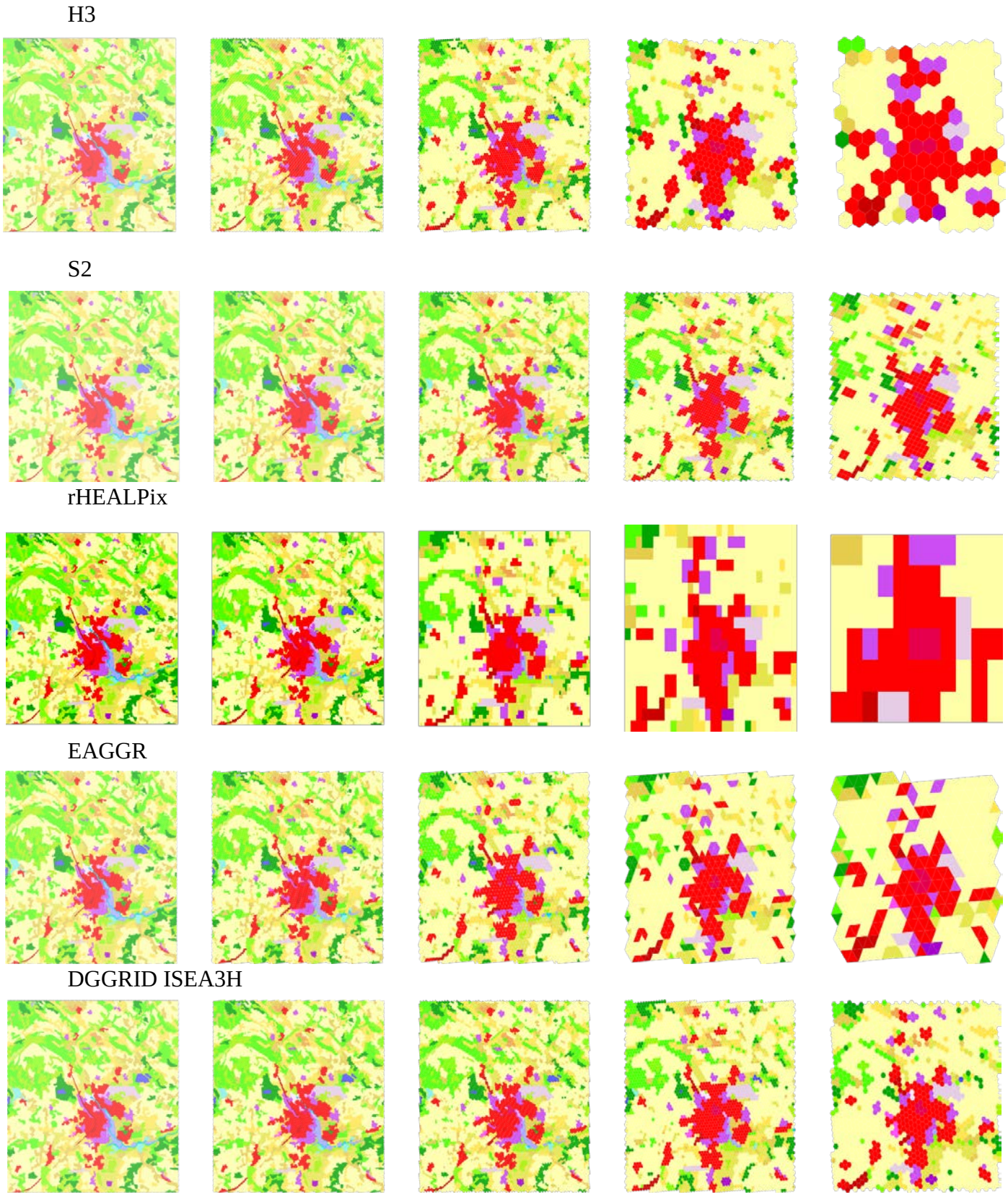


*Figure 38. CORINE Land Cover data ingested in various DGGSs with further downsampling into coarser 4 subsequent resolutions. Tartu city area.*

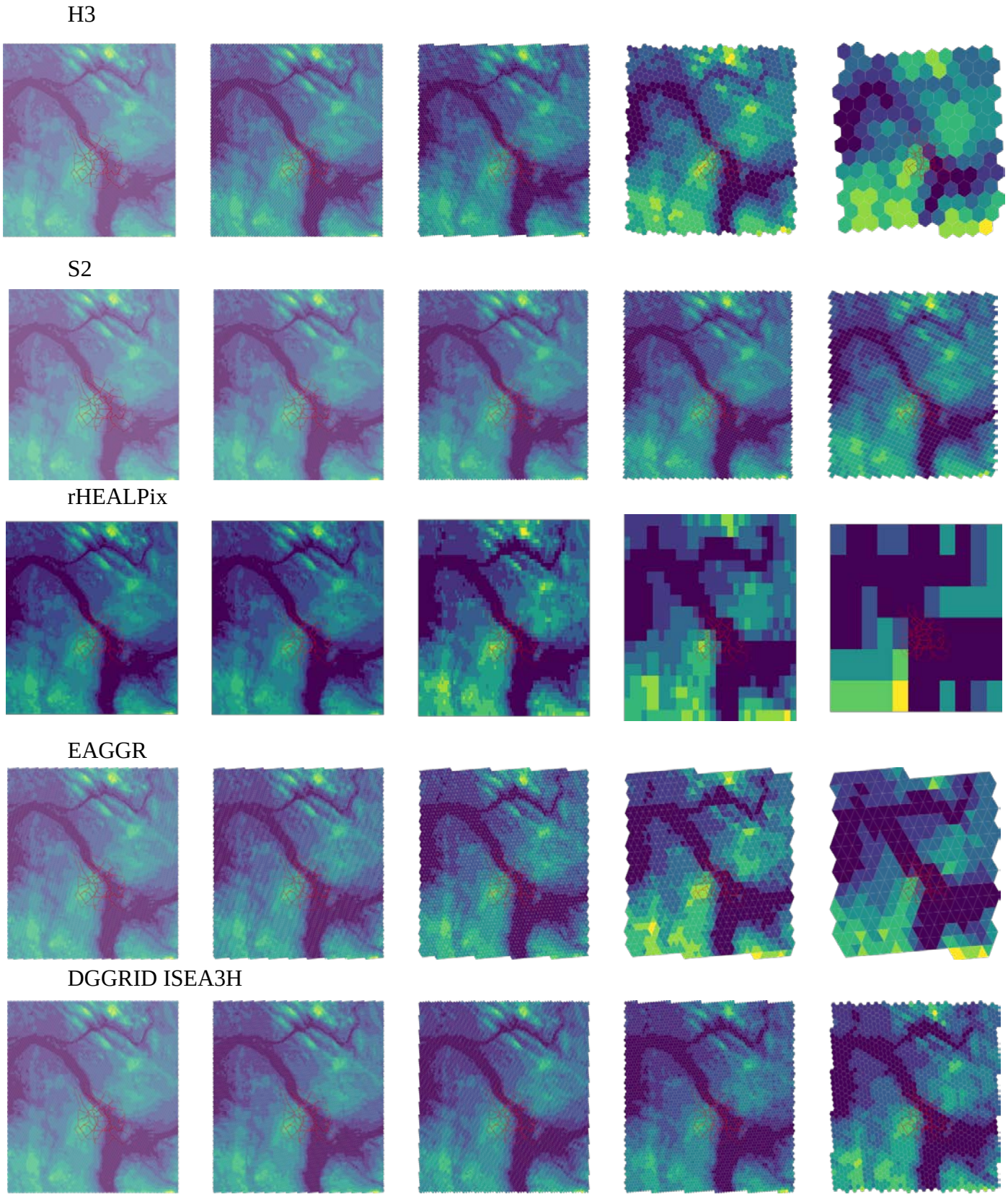


*Figure 39. MERIT DEM elevation data ingested in various DGGSs with further downsampling into coarser 4 subsequent resolutions. Tartu city area*

## 3.3 Construction of a DGGS-based data cube for the test territory

Resulted dataset covers Estonian territory with 124767 cells with an area of approximately 0.7 square kilometers each. Table 1 shows a 10-cells sample from this dataset. With spatial variable encoded as a cell index it is trivial to perform any further statistical analysis on other variables utilizing common data manipulation methods without any special spatial extensions or software. To illustrate convenience and potential of such data structure several queries has been performed.

*Table 1. 10-cells sample of the dataset with various data ingested into H3 DGGS cells covering Estonian territory at resolution level 8.*

| cell_id | elevation | land_cover | temperature | precipitation | population | soils |
|---|---|---|---|---|---|---|
| 88113096a7fffff | 64.04313847 | 311 | -52 | 611 | 0.012465258 | 7 |
| 88089bc155fffff | 37.11115479 | 242 | -35 | 640 | 29.31609726 | 2 |
| 8811368961fffff | 49.25409792 | 231 | -48 | 643 | 0.644336998 | 5 |
| 88089e2f6dfffff | 5.64717522 | 242 | -15 | 615 | 2.621392488 | 9 |
| 881f65b859fffff | 71.59759895 | 112 | -46 | 683 | 47.97258759 | 4 |
| 881135d321fffff | 91.22940297 | 312 | -49 | 653 | 0 | 4 |
| 881f653445fffff | 74.79504846 | 312 | -48 | 682 | 0 | 1 |
| 8811365419fffff | 148.282869 | 243 | -53 | 717 | 3.355177879 | 4 |
| 8811344465fffff | 34.11599537 | 313 | -50 | 646 | 0.027557198 | 14 |
| 881135d189fffff | 87.06087229 | 211 | -49 | 621 | 0.92075038 | 2 |

Figure 40 illustrates relations of elevation and precipitation variables. While ingested data already implicitly spatially aligned in cells such relations can be visualized directly from the dataset without any specific queries.

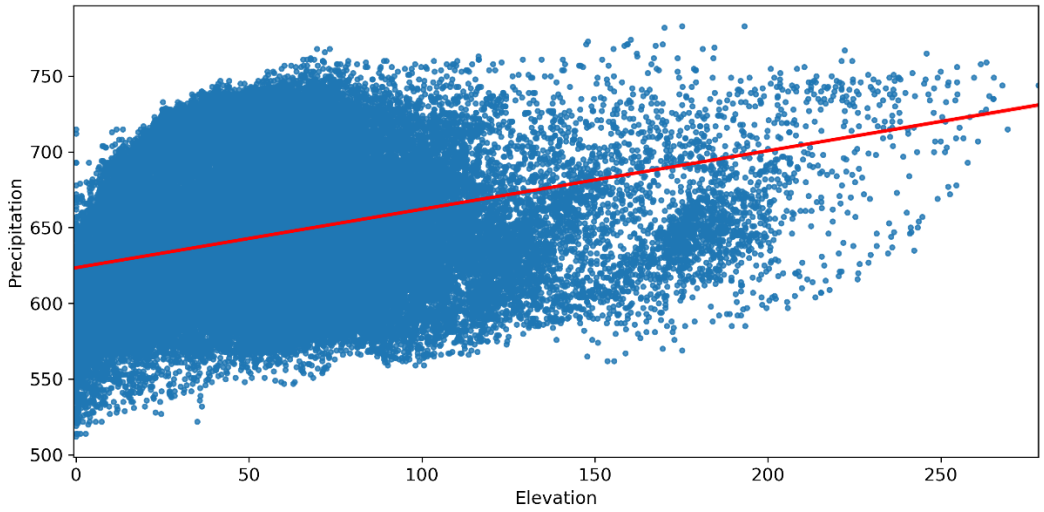


*Figure 40. Scatterplot of elevation and precipitation data ingested into H3 DGGS cells covering Estonian territory at resolution level 8 with regression line.*

Figure 41 shows number of people living in areas within particular elevation ranges. This is the result of simple group by operation with summation aggregation function.

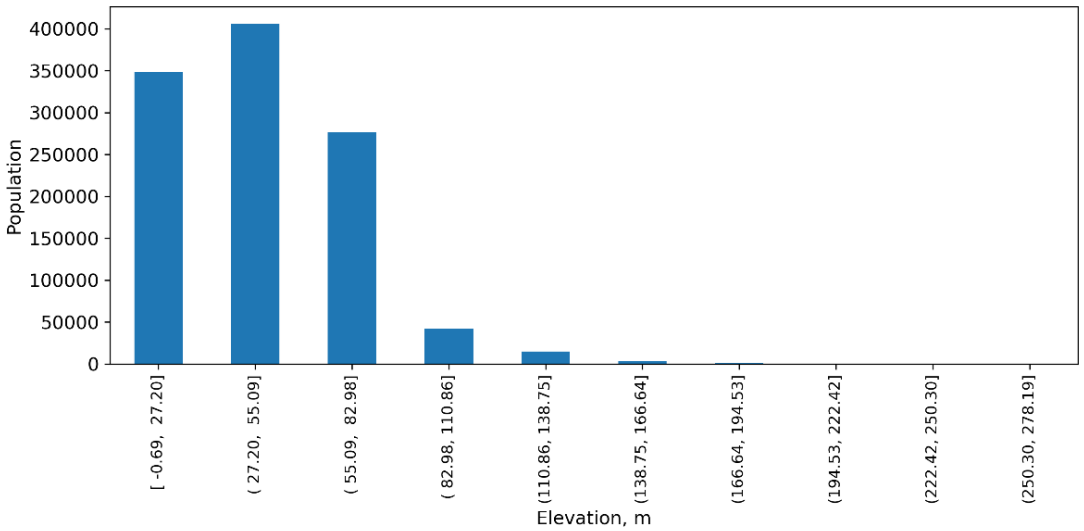


*Figure 41. Number of people living in different elevation ranges. From elevation and population data ingested into H3 DGGS cells covering Estonian territory at resolution level 8.*

JavaScript implementation of H3 library provides the possibility to visualize DGGS cells and other ingested variables in WEB-applications without necessity to transfer cells' geometry. Figure 42 shows elevation and soil variables from the dataset described above visualized in Kepler.gl application.

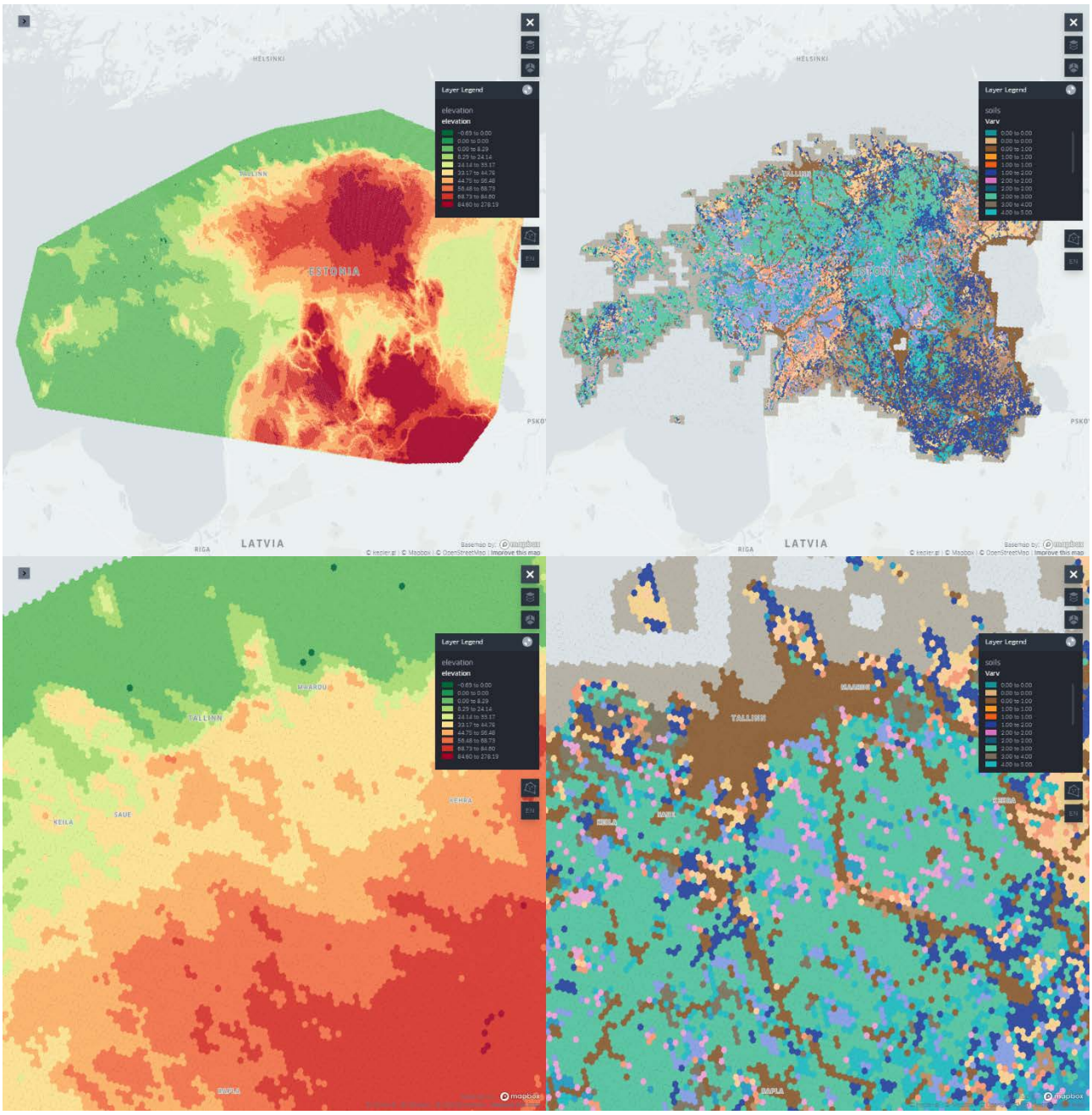


*Figure 42. Elevation and soil data elevation and population data ingested into H3 DGGS cells covering Estonian territory at resolution level 8 visualised in Kepler.gl WEB-application.*

It is worth mention that size of this whole dataset is roughly 8 mb, which is significantly smaller than any vector or raster representation of the same data.

# 4 Discussion

Growing frequency of publications related to Discrete Global Grids Systems and their usage for spatial data management in global scale indicates surging interest in the topic. It also can be seen in ongoing adoption of the global grid concept outside of scientific community by businesses and wider public. Examples of such adoption are DGGS-based solutions Uber H3 and Google S2 which are in intensive use outside of initial business domain where they have been developed and used before open-sourced (Addresscloud, Arup to name just a few). Data cubes, on the other hand, are already well-established new paradigm of managing remote sensing data in Earth Observation domain. Different country-level data cube – based platforms already functional or under development. (Swiss Data Cube, The Australian Geoscience Data Cube, Open Data Cube). However, Purrs et al. (2019) points out that data cube implementations based on traditional projected coordinate systems inherited fundamental flaws and their usage limited to local, country-level scale.

Open-source DGGS implementations were explored in this work from the perspective of integration of different datasets into a data cube. The main requirements to a DGGS implementation for this purpose is equal area of cells throughout resolution level on a global scale and a set of convenient methods to work with underlying grid structure. During practical interaction with the explored libraries minimum set of crucial methods for successful work with DGGS has emerged:

1) Conversion from point data with geographic coordinates to a cell index on given resolution level;
2) Conversion from a cell index to geographic coordinates of the cell centroid;
3) Construction of cell shape geometry as a polygon or a linestring in common GIS formats
4) Coverage of given extent in geographic coordinates or globally by cells indexes on a given resolution level
5) Methods for hierarchical and neighbor traversing

This is the minimum required functionality of a DGGS library which allows to ingest raster or vector data, transform it to a common resolution, store, analyze and visualize.

Among explored libraries Uber H3 provides the most convenient and rich API. It also has implementations and bindings to many popular programming languages and data manipulation platforms including Python, C#, Java, JavaScript, R, PostgreSQL, BigQuery. Unfortunately, as

it has been shown that the projection method which was used to construct H3 DGGS does not provide equal-area cells throughout global coverage. It largely complicates the library's usage for environmental statistics which would involve area calculations on a global scale.

Second most convenient for usage and well-developed DGGS implementation, Google S2, also has shown to have even higher area distortions. The library also has bindings to several popular programming languages and databases which guarantees streamlined data ingestion, storage, analysis and visualization workflow. Rectangular shape of the cells and lower aperture level which provides smoother transitions between resolution levels also makes it DGGS library of choice for some applications comparing to H3.

rHEALPix DGGS library benefits from underlying equal-area rHEALPix projection and indeed provides cells of the same area on global level. The major drawback of the library is Python as its main language of implementation which might be the reason why it is significantly slower comparing to libraries written in compiled languages (C, C++, Java). Also, there is no well-established community behind the library and no ports to other languages. Another issue which might be crucial for some applications is the aperture 9 property for the partitioning, so the difference in cells sizes between finer and coarser resolution is significant.

OpenEAGGR library's underlying DGGS is based on the ISEA projection which guarantee equal-area cells across global coverage. The has a minimum necessary set of methods for convenient DGGS manipulations. Only the ISEA4T version of OpenEAGGR DGGSs is fully functional, which limits the implementation of this library for usage as a basis for an environmental data cube due to the fact that triangular shape of cells not ideal choice in terms of shape's compactness. Although limited use of the hexagonal DGGS version (ISEA3H) to construct cells coverages and convert to/from geographic coordinates is possible, the library has known flaws in hierarchical traversing algorithms – its impossible to get children cells unambiguously using provided methods.

DGGRID is a great tool for constructing various DGGS with desired partition shape using ISEA projection for area preservation or Fuller-Gray if shape preservation is a desired property. The tool can be used for exploration purposes to investigate properties of different DGGS as it was done in this study, but to utilize it for any practical real-life implementation would require a further development of wrapper methods to simplify data ingestion and DGGS cell traversing workflow, because it currently is a standalone application and not a reusable library.

To conclude, among explored open-source DGGS implementations the most well-maintained and adopted by wider community libraries are lacking cells equal area property required for proper global geostatistical analysis. On the other hand, libraries which provides equal area cells are either complicated to use or have a poor community support. In order to use these libraries in real-life applications considerable additional development is required

Nevertheless, H3 library used in this study for a data cube construction might serve as a good example of what potential DGGS concept has when a particular implementation is at least to some extent widely adopted and standardized. Unique cells indexes which unambiguously encode the same location throughout different platforms and technologies opens a perspective of 'geometry-free' spatial data manipulations. One of the benefits of this feature has been illustrated with visualization of integrated Estonian dataset in Kepler.gl WEB-application. Due to uniqueness and recognition of H3 cell ids it is possible to significantly minimize the size of transferred data. This example also illustrated the simplicity of data discretization for distributed calculations or storage of significantly larger datasets on a Big Data scale. It has also been shown that with developed DGGS library as a basis for data cube it is possible to easily combine data from different sources, with different initial resolution and data formats. This extends a data cube nature from the Earth Observation domain beyond just raster data and a local country level.

There are several limitations of this study which are to be addressed in the future works. First, the usage of a relatively small test region for the data cube construction might be seemed inadequate for the purposes of demonstration of the global-scale nature of DGGS. For the future work, it would be necessary to construct similar data structure on a truly global level with integration of various country-level and global datasets. Second, in this work very basic analytical queries have been performed to demonstrate convenience of the DGGS-based data model. With constructed DGGS-based global-scale data cube more sophisticated analysis and modeling should be performed to illustrate the full potential of this approach. Third, time dimension of the data cube concept has not been utilized in the demonstration, but can be easily added as another variable.

Finally, for adoption and popularization of DGGS in general it is necessary to initiate development and implementation of DGGS-based tools and plugins in desktop or WEB-based GIS software in user-friendly, more familiar for a wider audience interface form.

## Conclusion

The main scope of this thesis was dedicated to Discrete Global Grid Systems – an alternative spatial indexing system and data model. The aim of this thesis was to evaluate applicability of currently available open-source DGGS implementations as a basis for a data cube construction for global-scale environmental datasets management and spatial statistical analysis.

Using 6 different datasets for the test region it has been shown that DGGS might be used as a basis for data cube-like structure with the possibility to easily combine, manage and further analyze data from various sources, with different initial resolution and data formats. Although limited to relatively small case-study area, it is obvious that such DGGS-based data structure, if underlying DGGS is widely adopted and standardized, is well-suitable for data discretization for distributed calculations or storage of significantly larger datasets on a Big Data scale. For the better demonstration, it is necessary to construct similar data structure using truly global datasets.

In this study, following open-source DGGS solutions have been examined for their usability as a basis for construction of a data cube for managing environmental data and perform statistical analysis on the global scale: Uber H3, Google S2, RiskAware OpenEAGGR, rHEALPix and DGGRID. Several geometric properties of these DGGS were empirically explored. As a result, it was concluded that among these DGGS implementations the most well-maintained and adopted by wider community libraries, H3 and S2, are lacking cells equal-area property required for proper global geostatistical analysis. On the other hand, libraries which utilizes DGGSs with equal area cells as rHEALPix or ISEA-based are either complicated to set up and use or have a poor community support.

During practical work with discussed DGGS implementations set of minimum requirements to functionality which would provide the possibility to use it for a data cube construction also has been formulated. In addition to this set of convenient methods, the implementation ideally should have same community support and presence in most popular programming languages and database technologies as Uber H3. Perspective work in this direction would be an extension of DGGRID tool functionality with convenient Python interface for DGGS manipulation and JavaScript adoption for rapid rendering of cells geometry.

## Summary

**Master Thesis in Geoinformatics for Urbanized Society: Integrating environmental datasets into a Data Cube using Discrete Global Grid Systems**

**Ivan Vasilyev**

The main scope of this thesis is Discrete Global Grid Systems – an alternative spatial indexing system and data model. In the era of Big Data and ongoing globalization scientific community, public and private companies who deal with spatial data face new challenges related to processing enormous amounts of data. Several petabytes of geospatial data occupy servers around the world, and more continues to be generated every day (Mahdavi-Amiri et al., 2015). In such conditions, traditional models for storing, managing and analysing spatial information are becoming inadequate due to global extent and increasing demand for distributed computation capacity, as it stated by many authors (Sahr et al.,2003, 2015; Sahr, 2008; Mahdavi-Amiri, 2015; Purss et al., 2019). One of possible solutions to tackle this issue is the use of Discrete Global Grid Systems (DGGSs). In a DGGS, the surface of the Earth is discretized into a set of regular cells. These cells are then addressed using a data structure or indexing mechanism that is further used to assign and retrieve data. The DGGS theory has been discussed for decades in the scientific community, but the interest among wider public has gained momentum in recent years including emergence of numerous DGGS-based software products in various domains (Adams, 2017; Uber Technologies Inc, 2018; Global Grid Systems, 2019; Bush, 2017; Gibb, 2016; Robertson et al., 2020) and the recent introduction of the DGGS abstract specification by Open Geospatial Consortium (OGC) (Peterson, 2019).

Another technological concept in the spotlight of the geospatial community for managing Big Spatial Data is the so-called data cube. Data cubes are n-dimensional arrays that are used to store query-ready spatial-temporal data ordered according to various attribute/coordinate axes, which can be spatial or non-spatial in nature (Alderson, 2020). The widest recognition and adoption of data cubes have been gained in recent years in Earth Observation (EO) domain. There is a growing number of implementations currently referred to as EO data cubes (Swiss Data Cube, The Australian Geoscience Data Cube, Open Data Cube to name just a few) with the goal of optimizing the way to store, manage, provide access to and analyse Big EO Data in a more convenient manner (Augustin et al., 2019).

Most of the raw global data (satellite derived, modelled climate data, land use, GPS-based location data) cannot easily be meaningfully spatially summarized (by area), because typically produced in World Geodetic System 84 (WGS 84) geographic coordinate reference system. This spatial data is further projected to planar reference systems (such as UTM zones or country-specific coordinate systems) to index and align data in order to be able to query, aggregate, summarize and otherwise analyze across raw observation data and thematic layers. This conventional approach of managing spatial data in map-projected data sets have led to a restricted set of data cube implementations that are each tightly coupled to the spatial constraints of the data and how they are stored resulting in barriers to interoperability and analysis on global scales (Purss et al., 2019). The equal area global properties of DGGS and its implicit global scope would allow for building a globally valid interoperable data cube.

The aim of this thesis was to evaluate applicability of currently available open-source DGGS implementations as a basis for a data cube construction for global-scale environmental datasets management and spatial statistical analysis. Following research questions were stated:

1) Is DGGS is suitable for a data cube indexing?

2) What type DGGS is suitable for global-scale data cube construction by its geometric properties among currently available open-source implementations?

3) Which currently available open-source DGGS implementations are suitable for integration with larger software ecosystem to build practical real-world application?

To answer these questions, first, it was necessary to establish theoretical foundation of DGGS. Second, geometric properties of several publicly accessible open-source implementations of DGGS were explored experimentally. Third, DGGS implementation-specific advantages and disadvantages in particular with focus on supporting a data cube implementation were explored. Finally, an example of operational DGGS-based data cube-like structure for test region was demonstrated.

Following open-source DGGS implementations were explored in this work: Uber Hexagonal Hierarchical Spatial Index (H3), Google S2, RiskAware Open Equal Area Global Grid (OpenEAGGR), Rearranged Hierarchical Equal Area isoLatitude Pixelization (rHEALPix) by Landcare Research New Zealand and DGGRID by Southern Oregon University. To operate with underlying DGGSs of the implementations provided APIs were used in Python programming language environment. Due to difference in provided APIs functionality native

functions were extended with set of uniform convenient methods as Python packages. Uber H3 library appeared to be the most maintained by the community, straightforward in setup in Windows and Linux environments and provide well-developed APIs in many popular programming languages.

For evaluation and comparison of the geometric properties of the DGGSs in these libraries two metrics were chosen, following Kimerling et al., (1999): normalized area and Zonal Standardized Compactness (ZSC). Additionally, influence of a DGGS partition aperture level on a cell area change dynamic between DGGS resolutions was examined. The study empirically shown that rHEALPix and DGGSs which are based on Icosahedral Snyder Equal Area projection provide cells of an equal area across the global coverage. In contrast, H3 and S2 DGGSs have significant areal differences in the global coverage. Regarding compactness, it was illustrated that hexagonal partitions of H3 and DGGRID-generated hexagonal DGGS provide the best possible compactness. However, the highest compactness values were achieved by H3 DGGS's Gnomonic projection which does not provide equal-area property. The smoothest transition between resolutions among explored implementations is provided by DGGRID-generated ISEA3H DGGS with aperture value of 3. The worst performance in this context had rHEALPix DGGS with aperture 9.

The conclusion of this study is that among explored open-source DGGS implementations the most well-maintained and adopted by wider community libraries are lacking cells equal area property required for proper global geostatistical analysis. On the other hand, libraries which provides equal area cells are either complicated to use or have a poor community support. In order to use these libraries in real-life applications considerable additional development is still required.

For the practical demonstration of a possibility to integrate datasets from different sources and of different origins into DGGS-based data cube-like structure 6 openly available datasets for Estonian territory were loaded into H3 DGGS. The choice of this DGGS for the demonstration was dictated by the fact that H3 library might serve as a good example of what potential DGGS concept has when a particular implementation is widely adopted and standardized. Unique cell indexes which unambiguously encode the same location throughout different platforms and technologies opens a perspective of 'geometry-free' spatial data manipulations. Due to uniqueness and recognition of H3 cell ids it is possible to significantly minimize the size of transferred data for WEB-applications. This example also illustrated the simplicity of data

discretization for distributed calculations or storage of significantly larger datasets on a Big Data scale. It has also been shown that with developed DGGS library as a basis for data cube it is possible to easily combine data from different sources, with different initial resolution and data formats. This extends a data cube nature from the Earth Observation domain beyond just raster data and a local country level.

# Kokkuvõte

## Linnastunud ühiskonna geoinformaatika magistritöö: keskkonnaandmestike integreerimine andmekuupi jagatud globaalsete võrkude süsteemide (DGGS) abil

**Ivan Vasilyev**

Selles töös keskendutakse peamiselt jagatud globaalsete võrkude süsteemidele (DGGS) – alternatiivsele ruumilise indekseerimise süsteemile ja andmemudelile. Suurandmete tehnoloogia ning teaduskogukonna pideva globaliseerumise ajastul seisavad ruumiliste andmetega töötavad avaliku sektori ja eraettevõtted silmitsi uute väljakutsetega, mis on seotud hiiglaslike andmehulkade töötlemisega. Maailma serverites on mitu petabaiti georuumilisi andmeid ning neid genereeritakse iga päev juurde (Mahdavi-Amiri et al., 2015). Sellistes tingimustes muutuvad traditsioonilised ruumilise teabe hoidmise, haldamise ja analüüsimise mudelid globaalse ulatuse ning üha suureneva nõudluse tõttu jaotatud arvutusjõudluse järele ebapiisavaks, seda on arvanud paljud autorid (Sahr et al., 2003, 2015; Sahr, 2008; Mahdavi-Amiri, 2015; Purss et al., 2019). Üks võimalikest lahendustest selle probleemiga tegelemiseks on jagatud globaalsete võrkude süsteemide (DGGS) kasutamine. DGGSi korral on Maa pind jaotatud tavaliste võrguruutude kogumiks. Seejärel määratakse nendele võrguruutudele andmestruktuuri või indekseerimismehhanismi abil aadressid, mida kasutatakse andmete suunatuseks ja välja otsimiseks. DGGSi teooriast on teaduskogukonnas räägitud juba kümneid aastaid, ent laiema avalikkuse huvi on hakanud hoogu koguma viimastel aastatel, muuhulgas on mitmesugustesse erinevatesse valdkondadesse tekkinud mitmeid DGGS-põhiseid tarkvaratooteid (Adams, 2017; Uber Technologies Inc, 2018; Global Grid Systems, 2019; Bush, 2017; Gibb, 2016; Robertson et al., 2020) ning standardiseerimisorganisatsioon Open Geospatial Consortium (OGC) (Peterson, 2019) võttis hiljuti kasutusele DGGSi abstraktse spetsifikatsiooni.

Teine praegu georuumilises kogukonnas tähelepanu keskmes olev tehnoloogiline kontseptsioon ruumiliste suurandmete haldamiseks on niinimetatud andmekuup. Andmekuubid on n-mõõtmelised massiivid, mida kasutatakse päringuvalmis ruumilis-ajaliste andmete säilitamiseks järjestatuna vastavalt mitmesugustele omadustele/koordinaattelgedele, mis võivad olla ruumilised või mitteruumilised (Alderson, 2020). Andmekuupe on viimastel aastatel maapinna kaugseire (EO) domeenis laialdaselt tunnustatud ning kasutusele võetud. Praegu EO andmekuupideks nimetatavate rakenduste (muuhulgas näiteks Swiss Data Cube,

The Australian Geoscience Data Cube, Open Data Cube) arv aina kasvab, et optimeerida EO suurandmete mugavamat säilitamist, haldamist, neile juurdepääsu tagamist ning nende analüüsimist (Augustin et al., 2019).

Enamikku töötlemata globaalsetest andmetest (satelliitidelt saadud, modelleeritud kliimaandmed, maakasutuse andmed, GPS-põhised asukohaandmed) ei saa piirkondlikult summeerida, sest tavaliselt genereeritakse need geograafiline koordinaatide referentssüsteemi standardiga World Geodetic System 84 (WGS 84). Need ruumilised andmed projitseeritakse andmete indekseerimiseks ja ühtlustamiseks tasandilistesse referentssüsteemidesse (näiteks UTM-tsoonid või riigipõhised koordinaatide süsteemid), et töötlemata seireandmeid ja temaatilisi kihte oleks võimalik päringutes kasutada, agregeerida, kokku võtta ja muul viisil analüüsida. Selle tavapärase ruumiandmete kaardile projitseeritud andmekogumitena haldamise tulemusena on tekkinud andmekuubi rakenduste kogum, see on tihedalt seotud andmete ning nende säilitamise ruumiliste piirangutega, mis põhjustab globaalsel skaalal takistusi koostalitusvõime ja analüüsi osas (Purss et al., 2019). DGGSi õigepindsus ning selle globaalne rakendatavus võimaldaks luua globaalselt kasutatava koostalitlusvõimelise andmekuubi.

Uurimustöö eesmärk oli hinnata praegu kättesaadavaid avatud lähtekoodiga DGGS-rakendusi globaalsel skaalal keskkonnaandmestike haldamiseks ja statistiliseks ruumianalüüsiks andmekuubi genereerimise alusena. Püstitati järgmised uurimisküsimused.

1. Kas DGGS sobib andmekuubi indekseerimiseks?
2. Missugust tüüpi DGGS praegu kättesaadavatest avatud lähtekoodiga rakendustest sobib geomeetriliste omaduste poolest andmekuubi genereerimiseks globaalsel skaalal?
3. Missugused praegu kättesaadavad avatud lähtekoodiga DGGS-rakendused sobivad suurema tarkvara integreerimiseks ökosüsteemiga, et luua praktiline pärismaailmas kasutatav rakendus?

Nendele küsimustele vastamiseks tuli esmalt selgitada välja DGGS-i teoreetiline alus. Teiseks uuriti eksperimentaalselt mitme avalikult juurdepääsetava avatud lähtekoodiga DGGS-rakenduse geomeetrilisi omadusi. Kolmandaks uuriti konkreetse DGGS-rakenduse eeliseid ja puudusi, keskendudes eelkõige andmekuubi rakendamise toetamisele. Viimaks esitleti näitena toimivat DGGS-põhise andmekuubi laadset struktuuri.

Selles töös uuriti järgmisi avatud lähtekoodiga DGGS-rakendusi: Uber Hexagonal Hierarchical Spatial Index (H3), Google S2, RiskAware Open Equal Area Global Grid (OpenEAGGR), Landcare Research New Zealandi Rearranged Hierarchical Equal Area isoLatitude Pixelization (rHEALPix) ja Lõuna-Oregoni Ülikooli DGGRID. DGGSide kasutamiseks, millel rakendused põhinesid, kasutati olemasolevaid APIsid programmeerimiskeele keskkonnas Python. Olemasolevate APIde erinevuse tõttu laiendati funktsionaalsuse põhifunktsioone Pythoni pakettidena ühtsete meetodite kogumiga. Leiti, et teegil Uber H3 on vaadeldud variantidest parim arendajate kogukond, see on Windowsi ja Linuxi keskkondades seadistatav ning omab enamikus programmeerimiskeeltes hästi väljatöötatud APIsid.

Nendes teekides DGGSide geomeetriliste omaduste hindamiseks ja võrdlemiseks valiti Kimerlingi jt (1999) alusel kaks tunnust: normeeritud ala ja tsonaalne standarditud kompaktsus (ZSC). Uuriti ka DGGSi jaotusastme mõju DGGS-resolutsioonide vahelise võrguruuduala muutumise dünaamikale. Uurimustöö näitab empiiriliselt, et rHEALPix ja DGGSid, mis põhinevad Snyderi ikosaheedrilise õigepindsel (ISEA) projektsioonil, tagavad globaalse kaetuse lõikes õigepindsed võrguruudud. H3 ja S2 DGGSid on seevastu globaalse kaetuse korral oluliste pindalaliste erinevustega. Kompaktsuse osas näidati, et H3 ja DGGRIDiga genereeritud kuusnurkse DGGSi eraldused tagavad parima kompaktsuse. Parimad kompaktsuse väärtused saavutati aga H3 DGGSi gnomoonilise projektsiooniga, mis ei taga õigepindsust. Kõige ladusama resolutsioonidevahelise ülemineku annab uuritud rakendustest DGGRIDiga genereeritud ISEA3H DGGS jaotusastmega 3. Kõige nõrgema tulemusega oli selles kontekstis rHEALPix DGGS jaotusastmega 9.

Uurimustöö järeldus on, et uuritud avatud lähtekoodiga DGGS-rakenduste seast kõige laiema arendajate kogukonna poolt kasutusele võetud teekidel ei ole nõuetekohaseks globaalseks geostatistiliseks analüüsiks vajalikku võrguruutude õigepindsust. Need teegid, millel on õigepindsed võrguruudud, on aga keerulised või vähese kasutajatoega.

Erinevatest allikatest ja erineva päritoluga andmestike DGGS-põhise andmekuubi laadsesse struktuuri integreerimise võimalikkuse demonstreerimiseks praktikas laeti kuus Eesti territooriumil avalikult juurdepääsetavat andmestikku üles H3 DGGSi. See DGGS valiti demonstreerimiseks seetõttu, et H3 võib tuua hea näite sellest, missugune on DGGS-kontseptsiooni võimalik potentsiaal siis, kui konkreetne rakendus on laialdaselt kasutusel ning standarditud. Unikaalsed võrguruuduindeksid, mis kodeerivad sama asukohta erinevate

platvormide ja tehnoloogia kaudu üheselt, pakuvad „geomeetriavabade“ ruumiandmete manipulatsioonide võimaluse. H3 võrguruutude IDde unikaalsuse ja äratuntavuse tõttu saab veebirakenduste jaoks ülekantavate andmete suurust oluliselt minimeerida. See näide illustreerib ka suurandmete tasandil hajusarvutuste või andmehalduse eesmärgil diskreetimise lihtsust. Tõestati ka, et kui andmekuubi alusena kasutatakse H3 DGGS-teeki, on erinevatest allikatest pärinevad, erineva esialgse resolutsiooni ja andmeformaadiga andmed kergesti ühendatavad. See tähendab, et maapinna kaugseire valdkonna andmekuup on kasutatav ka suuremõõtmeliste rasterandmete puhul.

## Acknowledgment

First of all, I would like to express my gratitude to my supervisors, Alexander Kmoch and Holger Virro for all the wise guidance and patient support which they have been providing during this work. Also, to Evelyn Uuemaa for sparking the interest and giving initial motivation to work on this topic. I am grateful to my pre-defence opponent, Isaac Newton Kwasi Buo, for his valuable feedback. Of course, I also would like to thank my defence opponent in advance for the time that will be spend on thoroughly reading this thesis and all the comments to be made. Finally, I am deeply thankful to all the people involved in this master's program. It was a truly life-changing experience, I reached most of the goals which I had in mind 2 years ago during the application process, so now it is time to set new ones.

## References


Adams, B. (2017). Wāhi, a discrete global grid gazetteer built using linked open data. *International Journal of Digital Earth*, *10*(5), 490–503. https://doi.org/10.1080/17538947.2016.1229819

Alderson, T., Purss, M., Du, X., Mahdavi-amiri, A., & Samavati, F. (2020). Manual of Digital Earth. In *Manual of Digital Earth*. https://doi.org/10.1007/978-981-32-9915-3

Alderson, T., Purss, M., Du, X., Mahdavi-Amiri, A., & Samavati, F. (2020). Digital Earth Platforms. In *Manual of Digital Earth* (pp. 25–55).

Amiri, A. M., Bhojani, F., & Samavati, F. (2013). One-to-Two Digital Earth. https://doi.org/10.1007/978-3-642-41939-3

Amiri, A., Samavati, F., & Peterson, P. (2015). Categorization and Conversions for Indexing Methods of Discrete Global Grid Systems. *ISPRS International Journal of Geo-Information*, *4*(1), 320–336. https://doi.org/10.3390/ijgi4010320

Augustin, H., Sudmanns, M., Tiede, D., Lang, S., & Baraldi, A. (2019). Semantic earth observation data cubes. *Data*, *4*(3). https://doi.org/10.3390/data4030102

Barnes, R. (2019). Optimal orientations of discrete global grids and the Poles of Inaccessibility. *International Journal of Digital Earth*, 1–14. https://doi.org/10.1080/17538947.2019.1576786

Bauer-Marschallinger, B., Sabel, D., & Wagner, W. (2014). Optimisation of global grids for high-resolution remote sensing data. *Computers and Geosciences*, *72*, 84–93. https://doi.org/10.1016/j.cageo.2014.07.005

Baumann, P. (2017). *The Datacube Manifesto*. 1–2. Retrieved from http://earthserver.eu/tech/datacube-manifesto

Birch, C. P. D., Oom, S. P., & Beecham, J. A. (2007). Rectangular and hexagonal grids used for observation, experiment and simulation in ecology. *Ecological Modelling*, *206*(3–4), 347–359. https://doi.org/10.1016/j.ecolmodel.2007.03.041

Bolstad, P. (2008). GIS Fundamentals: A first text on geographic information systems (3rd ed.). White Bear Lake: Eider Press.

Bowater, D., & Stefanakis, E. (2018). The rHEALPix discrete global grid system: Considerations for Canada. *Geomatica, 72*(1), 27–37. https://doi.org/10.1139/geomat-2018-0008

Bush, I. (2017a). Literature Review & Prototype Evaluation. Issue 1. *Riskaware Ltd*

Bush, I. (2017b). OpenEAGGR Software Design Document. Issue 1. *Riskaware Ltd*

https://doi.org/10.1007/978-3-030-28501-2_34

de Sousa, Luís Moreira, & Leitão, J. P. (2018). HexASCII: A file format for cartographical hexagonal rasters. *Transactions in GIS, 22*(1), 217–232. https://doi.org/10.1111/tgis.12304

Dutton, G. (1989). Planetary modelling via hierarchical tessellation. *Auto-Carto 9. Proc. Symposium, Baltimore, MD, 1989*, *01760*(508), 462–471.

Dutton, Geoffrey. (n.d.). The Making of a Global Grid — And What To Make of It.

Dutton, Geoffrey. (1996). Encoding and Handling Geospatial Data with Hierarchical Triangular Meshes. *Proceedings of the 7th Symposium on Spatial Data Handling*, 505–518. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.461.5603%0Ahttp://www.spatial-effects.com/papers/conf/GDutton_SDH96.pdf

Gibb, R. G. (2016). The rHEALPix Discrete Global Grid System. *IOP Conference Series: Earth and Environmental Science*, *34*(1). https://doi.org/10.1088/1755-1315/34/1/012012

Gibb, R., Raichev, A., & Speth, M. (2016). The rHEALPix DGGS preprint.

Giuliani, G., Camara, G., Killough, B., & Minchin, S. (2019). Earth Observation Open Science: Enhancing Reproducible Science Using Data Cubes. *Data, 4*, 147. https://doi.org/doi:10.3390/data4040147

Goodchild, M. F., & Shiren, Y. (1992). A hierarchical spatial data structure for global geographic information systems. *CVGIP: Graphical Models and Image Processing*, *54*(1), 31–44. https://doi.org/10.1016/1049-9652(92)90032-S

Google groups. (n.d.). S2Geometry. Retrieved November 14, 2020, from

https://s2geometry.io/getS2.html

Gray, R. W. (1995). Exact transformation equations for Fuller's world map. *Cartographica*, (32), 17–25.

Gregory, M. J., Kimerling, A. J., White, D., & Sahr, K. (2008). A comparison of intercell metrics on discrete global grid systems. *Computers, Environment and Urban Systems*, *32*(3), 188–203. https://doi.org/10.1016/j.compenvurbsys.2007.11.003

Guo, H. (2017). Big Earth data: A new frontier in Earth and information sciences. *Big Earth Data*, *1*(1–2), 4–20. https://doi.org/10.1080/20964471.2017.1403062

Jubair, M. I., Alim, U., Röber, N., Clyne, J., & Mahdavi-Amiri, A. (2016). Icosahedral maps for a multi-resolution representation of earth data. *VMV 2016 - Vision, Modeling and Visualization*, (October), 161–168. https://doi.org/10.2312/vmv.20161355

Kimerling, J. A., Sahr, K., White, D., & Song, L. (1999). Comparing Geometrical Properties of Global Grids. *Cartography and Geographic Information Science*, *26*(4), 271–288. https://doi.org/10.1559/152304099782294186

Lei, K., Qi, D., & Tian, X. (2020). A new coordinate system for constructing spherical grid systems. *Applied Sciences (Switzerland)*, *10*(2). https://doi.org/10.3390/app10020655

Liao, C., Tesfa, T., Duan, Z., & Leung, L. R. (2020). Watershed delineation on a hexagonal mesh grid. *Environmental Modelling and Software*, *128*, 1364–8152. https://doi.org/10.1016/j.envsoft.2020.104702

Lin, B., Zhou, L., Xu, D., Zhu, A.-X., & Lu, G. (2018). A discrete global grid system for earth system modeling. *International Journal of Geographical Information Science*, *32*(4), 711–737. https://doi.org/10.1080/13658816.2017.1391389

Luo, J., Zhang, W., Su, J., & Xiang, F. (2019). Hexagonal convolutional neural networks for hexagonal grids. *IEEE Access*, *7*, 142738–142749. https://doi.org/10.1109/ACCESS.2019.2944766

Mahdavi-Amiri, A., Alderson, T., & Samavati, F. (2015). A Survey of Digital Earth. *Computers and Graphics (Pergamon)*, *53*, 95–117. https://doi.org/10.1016/j.cag.2015.08.005

Mahdavi-Amiri, A., Harrison, E., & Samavati, F. (2015). Hexagonal connectivity maps for

Digital Earth. *International Journal of Digital Earth*, *8*(9), 750–769. https://doi.org/10.1080/17538947.2014.927597

Mahdavi Amiri, A., Alderson, T., & Samavati, F. (2019). Geospatial Data Organization Methods with Emphasis on Aperture-3 Hexagonal Discrete Global Grid Systems. *Cartographica: The International Journal for Geographic Information and Geovisualization*, *54*(1), 30–50. https://doi.org/10.3138/cart.54.1.2018-0010

Mocnik, F. B. (2019). A novel identifier scheme for the ISEA Aperture 3 Hexagon Discrete Global Grid System. *Cartography and Geographic Information Science*, *46*(3), 277–291. https://doi.org/10.1080/15230406.2018.1455157

Peterson, P. (2019). Discrete Global Grid Systems – A Framework for the Next Era in Big Earth Data A New OGC Standard Digital Earth Spatial Reference System. (August).

Purss, M. B. J., Gibb, R., Samavati, F., Peterson, P., & Ben, J. (2016). The OGC® Discrete Global Grid System core standard: A framework for rapid geospatial integration. *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 3610–3613. https://doi.org/10.1109/IGARSS.2016.7729935

Purss, M. B. J., Peterson, P. R., Strobl, P., Dow, C., Sabeur, Z. A., Gibb, R. G., & Ben, J. (2019). Datacubes: A Discrete Global Grid Systems Perspective. *Cartographica: The International Journal for Geographic Information and Geovisualization*, *54*(1), 63–71. https://doi.org/10.3138/cart.54.1.2018-0017

Robertson, C., Chaudhuri, C., Hojati, M., & Roberts, S. A. (2020). An integrated environmental analytics system (IDEAS) based on a DGGS. *ISPRS Journal of Photogrammetry and Remote Sensing*, *162*, 214–228. https://doi.org/10.1016/j.isprsjprs.2020.02.009

Sahr, K. (2011). Hexagonal discrete global grid system for geospatial computing. *Archives of Photogrammetry, Cartography and Remote Sensing*, *22*, 363–376.

Sahr, Kevin. (2008). Location coding on icosahedral aperture 3 hexagon discrete global grids. *Computers, Environment and Urban Systems*, *32*(3), 174–187. https://doi.org/10.1016/j.compenvurbsys.2007.11.005

Sahr, Kevin, Dumas, M., & Choudhuri, N. (2011). *The PlanetRisk Discrete Global Grid*

*System*. 1–5.

Sahr, Kevin, White, D., & Kimerling, A. J. (2004). Geodesic Discrete Global Grid Systems. *Cartography and Geographic Information Science*, *30*(2), 121–134. https://doi.org/10.1559/152304003100011090

Seong, J. C., Mulcahy, K. A., & Usery, E. L. (2002). The sinusoidal projection: A new importance in relation to global image data. *Professional Geographer*, *54*(2), 218–225. https://doi.org/10.1111/0033-0124.00327

Sirdeshmukh, N. (2018). Utilizing a Discrete Global Grid System For Handling Point Clouds With Varying Location , Time , and Densities.

Snyder, J. P. (1992). An equal-area map projection for polyhedral globes. *Cartographica*, *29*, 10–21.

Sousa, Lu´ıs M. de. (2018). *Remaining gaps in open source software for Big Spatial Data*. 0–8. Retrieved from https://doi.org/10.7287/peerj.preprints.27215v1

Sousa, Luís Moreira De, & Poggio, L. (2019). *Comparison of FOSS4G Supported Equal-Area Projections Using Discrete Distortion Indicatrices*. 1–13. https://doi.org/10.3390/ijgi8080351

Tong, X., Ben, J., Liu, Y., & Zhang, Y. (2013). Modeling and expression of vector data in the hexagonal discrete global grid system. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, *40*(4W2), 15–25. https://doi.org/10.5194/isprsarchives-XL-4-W2-15-2013

Uber Ingenering Inc. (n.d.). H3. Retrieved November 17, 2020, from https://uber.github.io/h3/#/documentation/core-library/h3-index-representations

Uber Technologies Inc. (2018). H3: Uber’s Hexagonal Hierarchical Spatial Index. Retrieved November 13, 2020, from https://eng.uber.com/h3/

van Wijk, J. J. (2008). Unfolding the earth: Myriahedral projections. *Cartographic Journal*, *45*(1), 32–42. https://doi.org/10.1179/000870408X276594

Vince, A. (2006a). Indexing a Discrete Global Grid. *2*, 3–17.

Vince, A. (2006b). Indexing the aperture 3 hexagonal discrete global grid. *Journal of Visual*

*Communication and Image Representation*, *17*(6), 1227–1236. https://doi.org/10.1016/j.jvcir.2006.04.003

White, D. (2000). Global grids from recursive diamond subdivisions of the surface of an octahedron or icosahedron. *Environmental Monitoring and Assessment*, *64*(1), 93–103. https://doi.org/10.1023/A:1006407023786

White, Denis, Kimerling, A. J., Sahr, K., & Song, L. (1998). Comparing area and shape distortion on polyhedral-based recursive partitions of the sphere. *International Journal of Geographical Information Science*, *12*(8), 805–827. https://doi.org/10.1080/136588198241518

Zhenlong Li, Zhipeng Gui, Barbara Hofer, Yan Li, S. S. and S. S. (2020). Geospatial Information Processing Technologies. In *Manual of Digital Earth* (pp. 191–229).

Zhou, M., Chen, J., & Gong, J. (2016). A virtual globe-based vector data model: quaternary quadrangle vector tile model. *International Journal of Digital Earth*, *9*(3), 230–251. https://doi.org/10.1080/17538947.2015.1016558

**Non-exclusive license to reproduce thesis and make thesis public**

I, Vasilyev Ivan,

1. herewith grant the University of Tartu a free permit (non-exclusive license) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

“Integrating environmental datasets into a Data Cube using Discrete Global Grid Systems” supervised by PhD Alexander Kmoch and MSc Holger Virro

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons license CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive license does not infringe other persons’ intellectual property rights or rights arising from the personal data protection legislation.

Ivan Vasilyev, Tartu, 19.01.2021