F5
FIBERBUNDLE HDF5

Werner Benger
Center for Computation & Technology
Louisiana State University
http://sciviz.cct.lsu.edu/

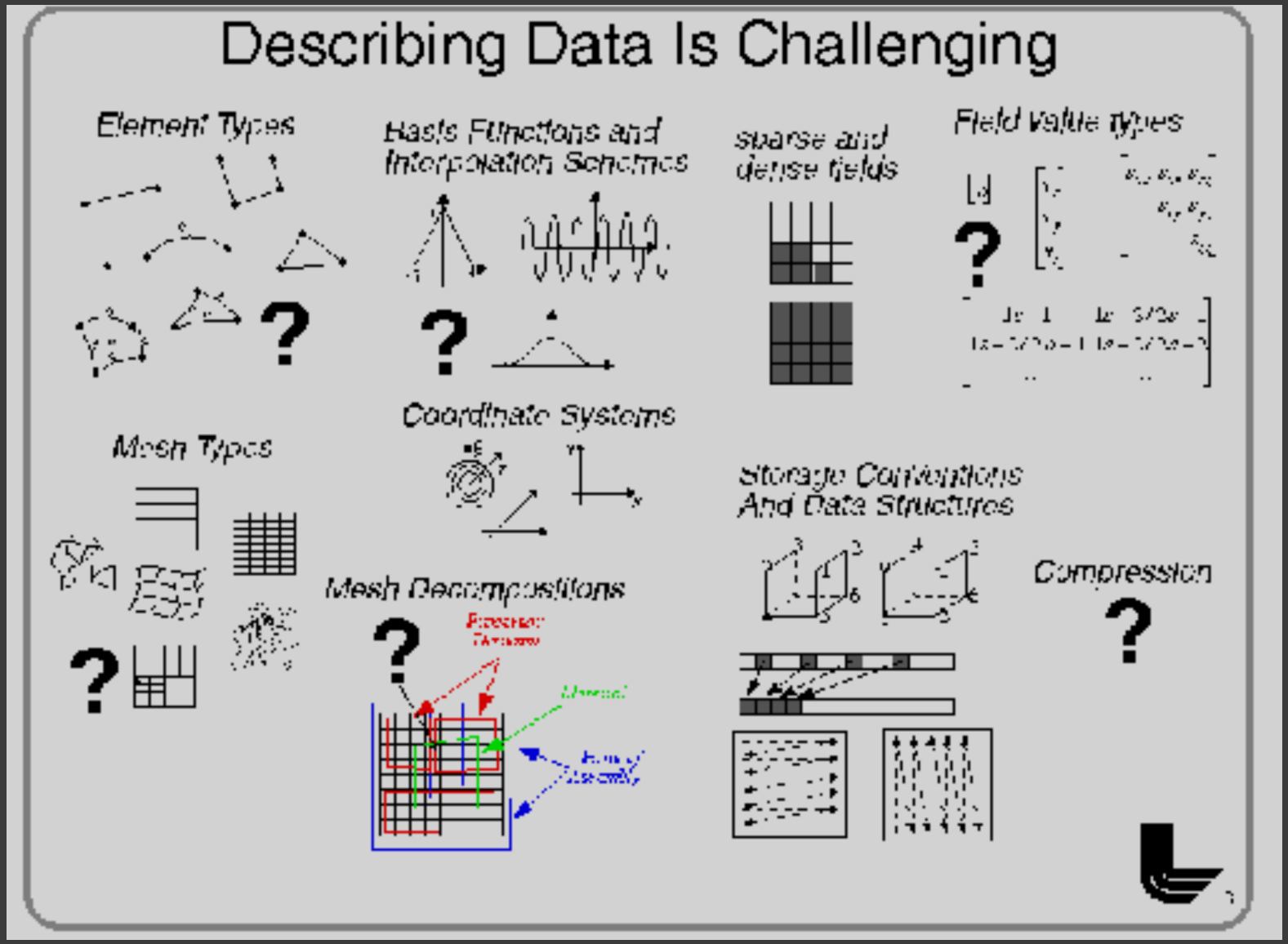
Abstract

“The proper abstractions for scientific data are known. We just have to use them.” (D. M. Butler & S. Bryson, 1992): There is no common standard or agreement on file formats for scientific data, but there could be. The inductive approach common in software development usually leads to “special solutions for special problems” which are extended on a case by case basis. Consequently, each application develops their own approach of describing data, ultimately leading to incompatibilities in the file formats associated with these applications and groups of applications. In contrast, the deductive approach to describe data based on the theory of fiber bundles leads to a generalized way of describing scientific data independent of a particular scientific domain or application as it exhibits the mathematical abstractions that are common across disciplines.

This presentation will review the “F5” approach of modeling data for scientific visualization using fiber bundles on top of HDF5 as underlying I/O layer. The F5 model has originally been developed about 12 years ago in the context of numerical relativity simulation data as produced by Cactus, for handling both primary simulation (uniform grids, adaptive meshes, curvilinear grids, multipatch data…) and secondary post-processing data (surfaces, lines, particle sets, …). The formalism and implementation itself is independent of Cactus and has since been applied to other domains such as medical imaging, computational fluid dynamics (CFD), smoothed particle hydrodynamics (SPH) and geoscience data, easing scientific visualization through this common denominator. The presentation will provide a short review of the basic concepts and provide an in-depth discussion on how the F5 model describes various common and less common data types.

Overview

- The Problem
- Mathematics for Visualization
- The Fiber Bundle data model
- HDF5
- Mapping the Fiber Bundle model to HDF5
  - Basics
  - Extensions
  - F5 API



Thanks to Mark Miller, LLNL

Data Types

- Point Sets
- Unstructured Cell Data
- Tetrahedral Grids
- Regular Grids
- Uniform Grids
- Uniform Cartesian Grids
- Uniform Polar Grids
- …

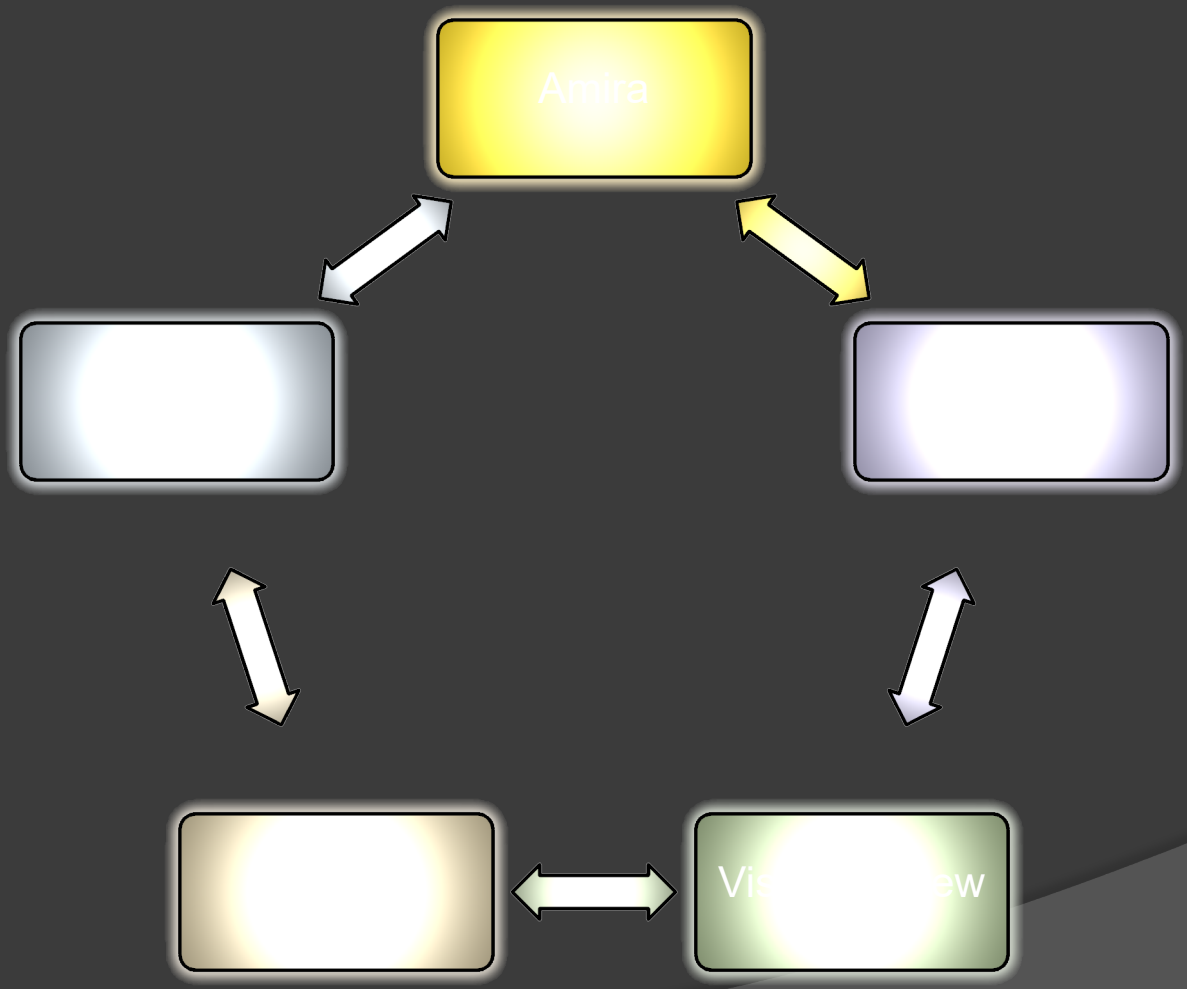
- Triangular Surfaces
- Quad-based Surfaces
- Irregular Surfaces
- Hierarchical Grids (AMR)
- Streamlines, Particle Trajectories, Geodesics
- Apparent Horizons
- Embedding Surfaces
- …

“Special problems need special solutions”

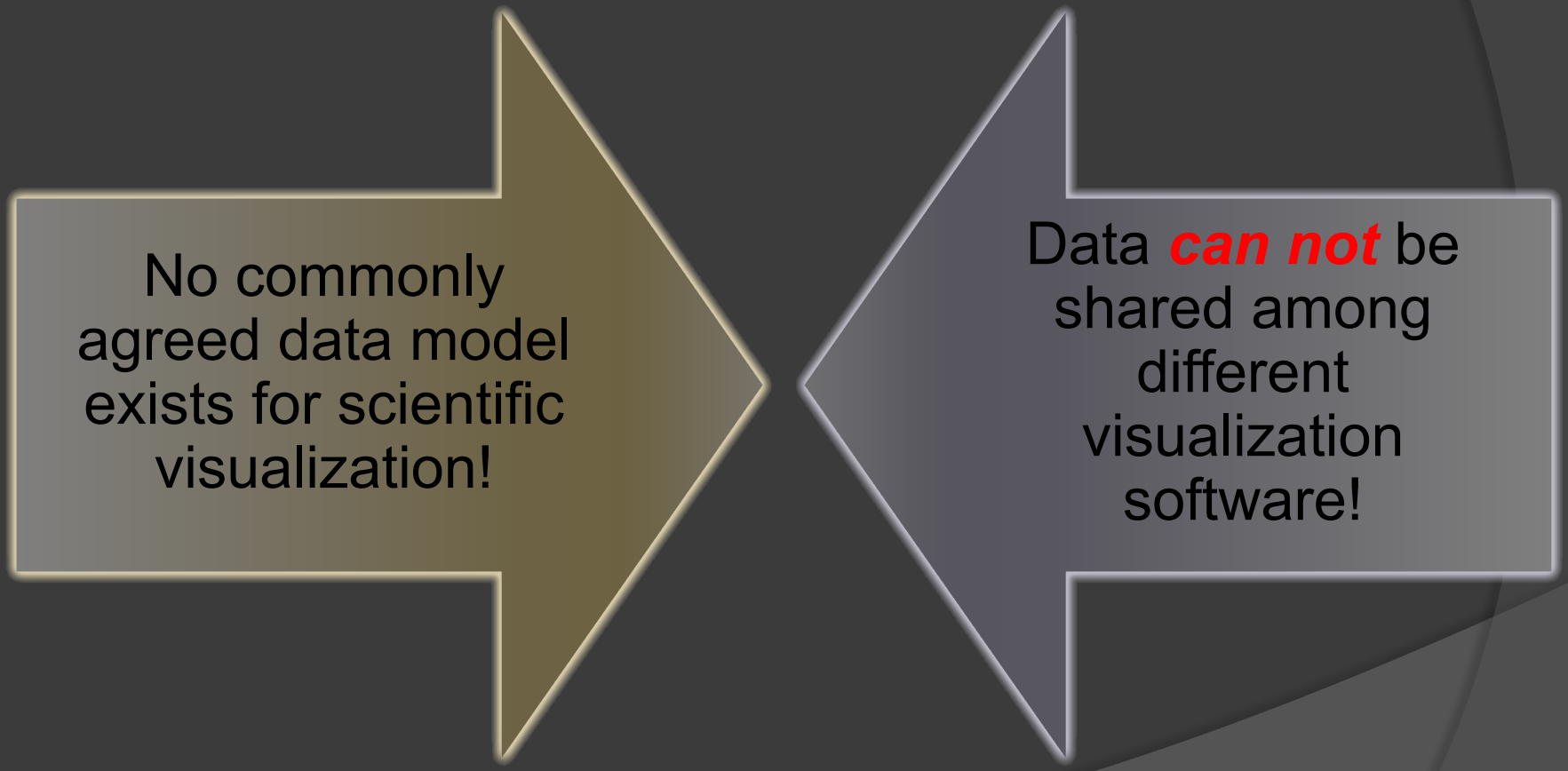
- Inductive approach in software engineering leads to incremental advances starting from a specific problem
- Solutions often implemented on an enumerated case-by-cases basis

→ Results in many independent solutions

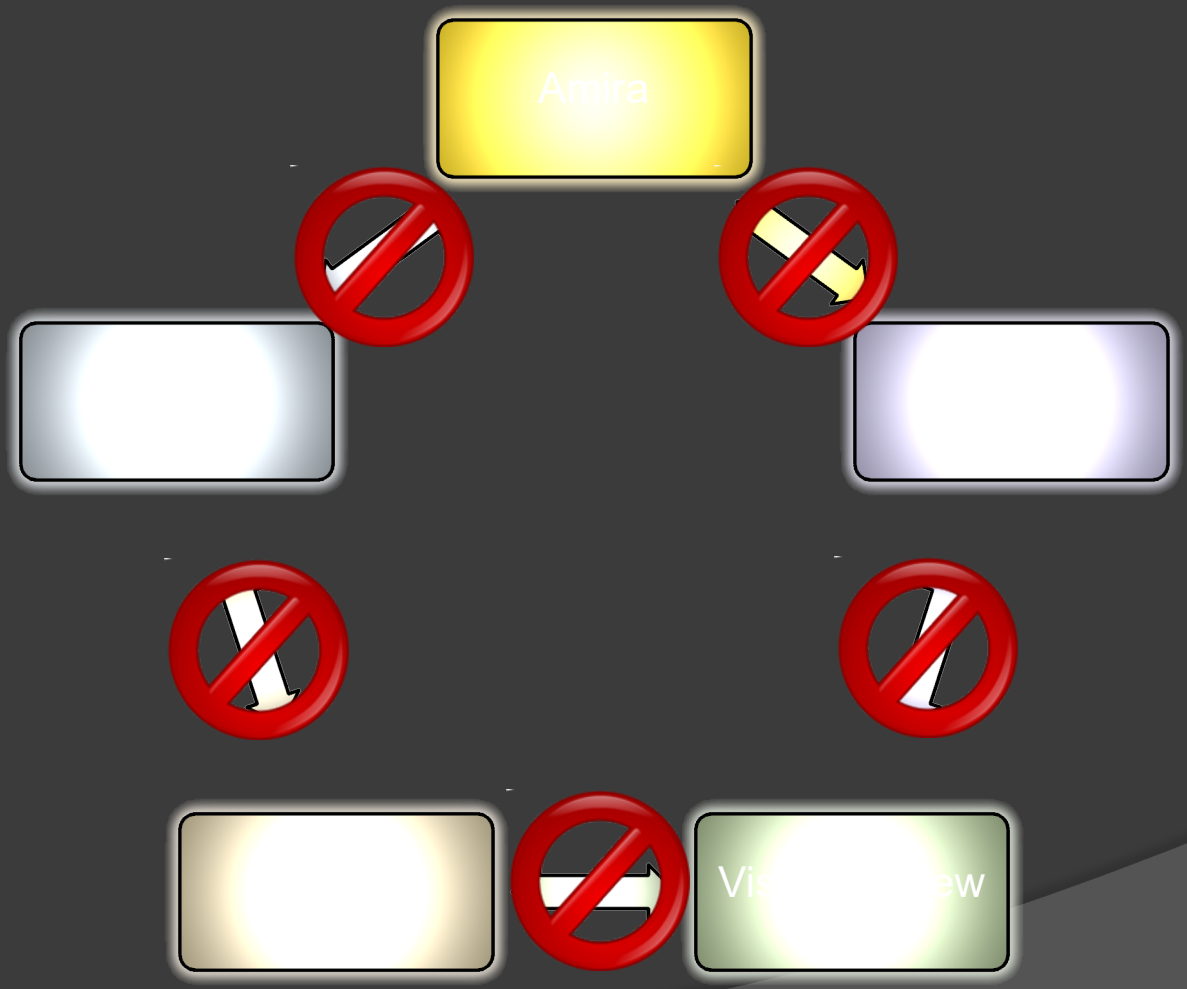
Interoperability?



Data Model in SciViz



Interoperability… wishful thinking



Interoperability

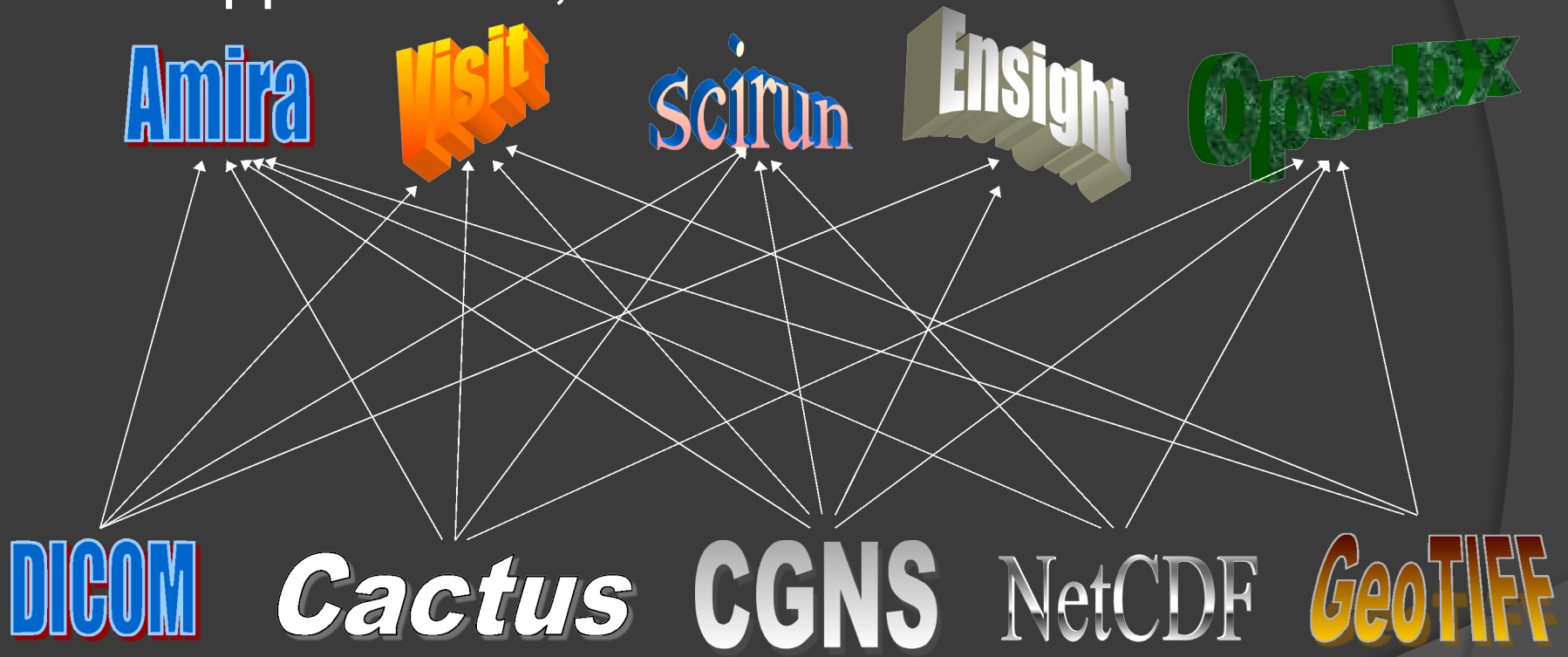
- N Applications

Diverse Visualization Software ↔ Data sets in diverse file formats

- M File Formats

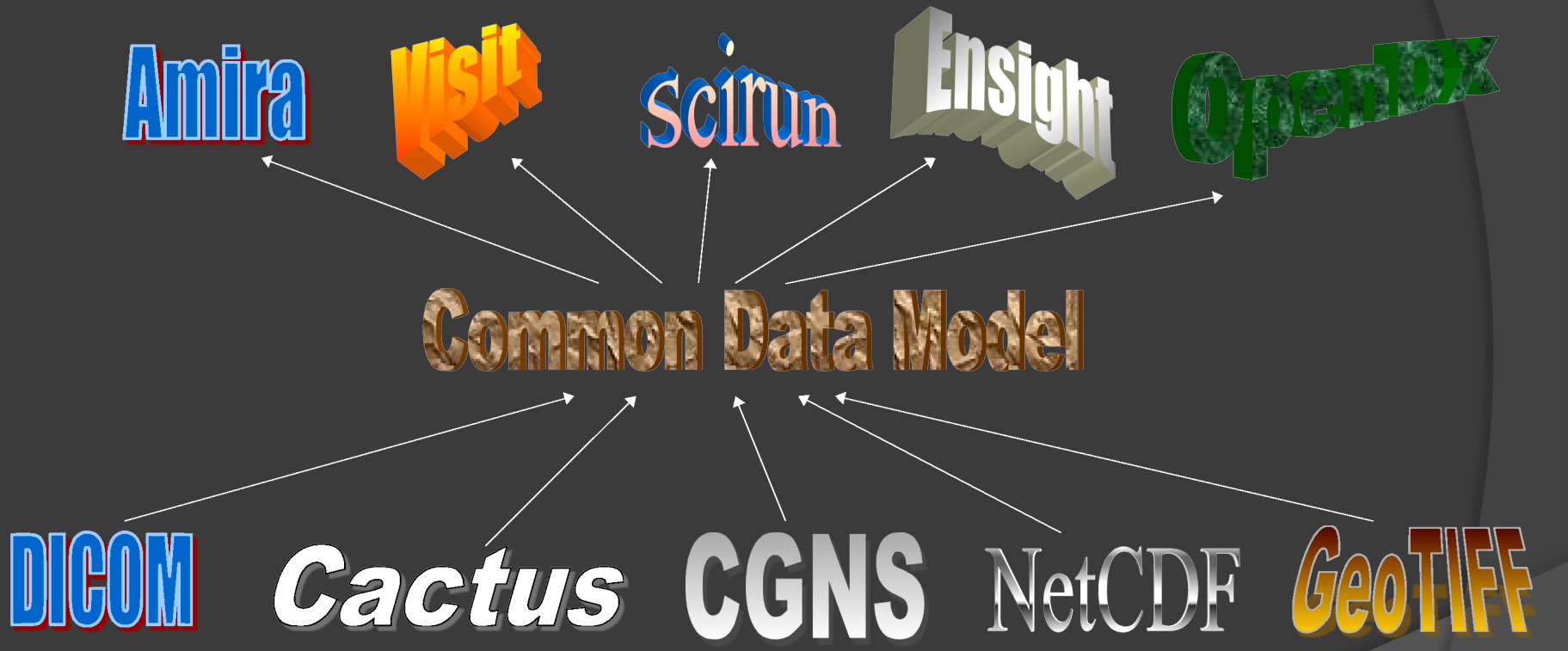
N x M Implementation Effort

- n applications, m file formats

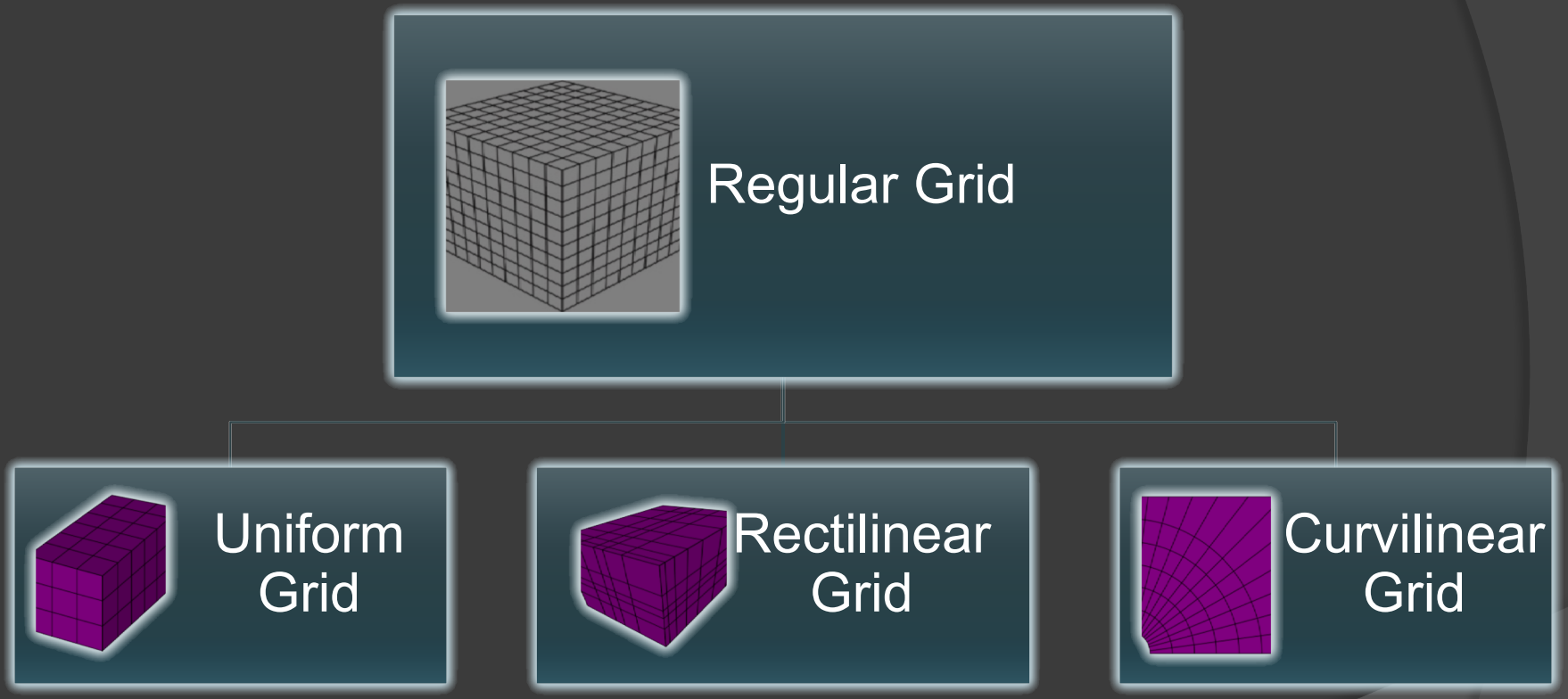


N + M Common Data Model

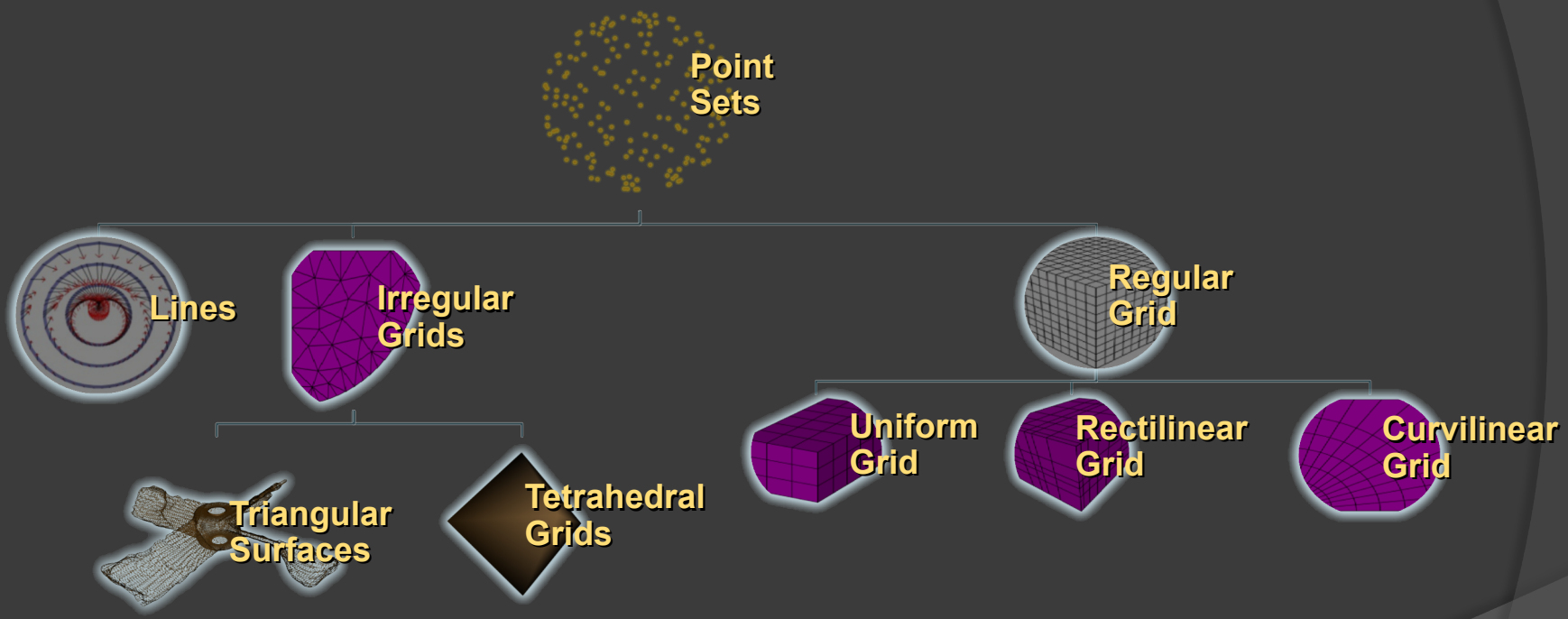
- n sources, m sinks, 1 route



Commonalities of Grid types



Hierarchy of Grid types



Challenge of a Common Data Model

"The proper abstractions for scientific data are known. We just have to use them."

D. M. Butler & S. Bryson

Vector-Bundle Classes form Powerful Tool for Scientific Visualization

Computers in Physics, Vol. 6, No 6., Nov/Dec 1992

Motivation

- Elegant mathematical formulations inspire visualization methods to become
  - More general
  - Wider applicable
  - Better structured, more modular
- *Deductive* Approach
  - Methods (I/O, numerical algorithms, visualization) are implemented with the broader context in mind

Mathematics for Visualization

Topology

- Discretization schemes, combinatorial structure, relational information

Differential Geometry

- Coordinate representations, transformations, differential operators, tensor algebra

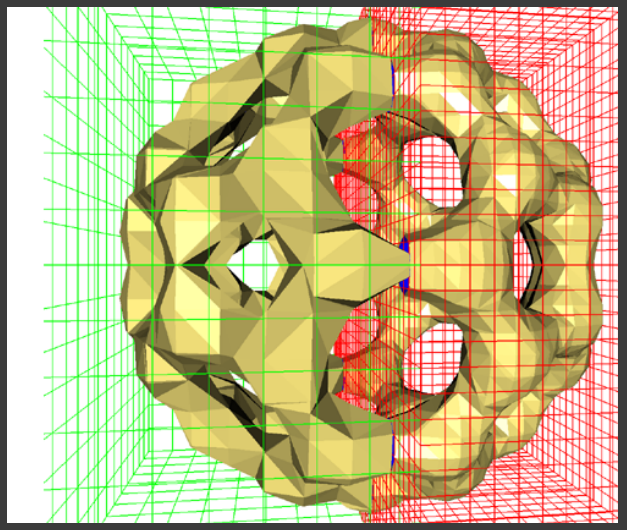
Geometric Algebra

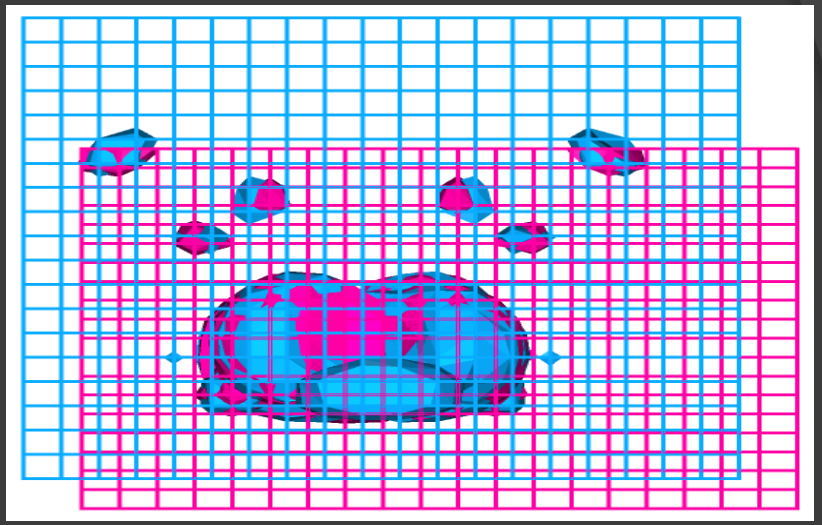
- N-dimensional Rotations, navigation, data analysis, feature extraction, spinor formalism, generalized quaternions, mimetic operators

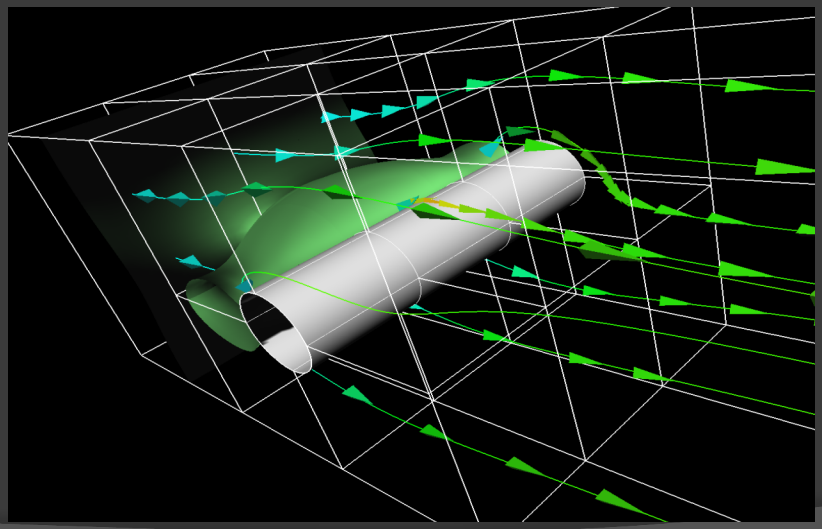
Topology

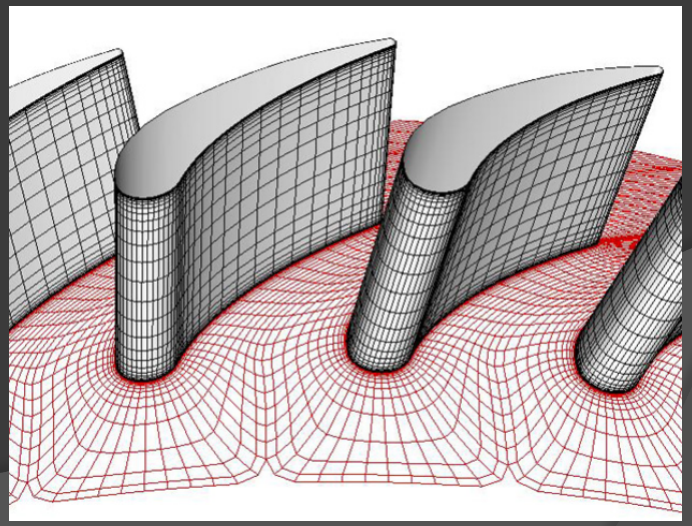
- Concept of fiber bundles
- Concept of cw-complexes and k-Skeletons
- Relational Information

Fiber Bundles: Unification of data types









Mathematical Definition

Let *E, B* be topological spaces and $f : E \to B$ a continuous map.

(*E,B,f*) is called a *(fiber) bundle* if there exists a space *F* such that the union of fibers of a neighborhood $U_b \subset B$ of each point $b \in B$ are homeomorphic to $U_b \times F$ such that the projection pr_1 of $U_b \times F$ is U_b again.

Naming

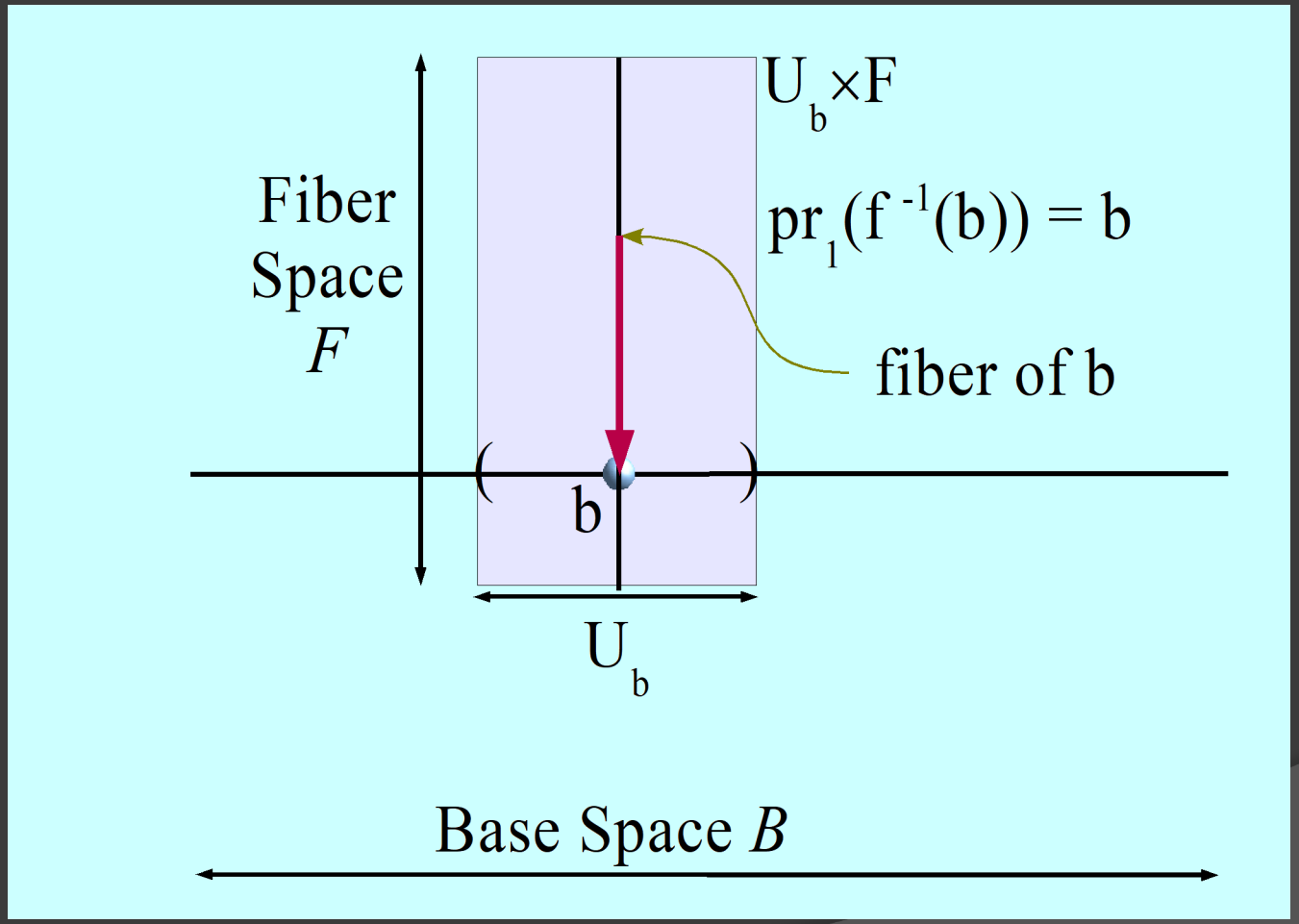
- E is called the *total space* E,
- B is called the *base space*,
- f : E $\rightarrow$ B the *projection map*.

(E,B,f: E$\rightarrow$B) ***bundle*** $\Leftrightarrow \exists$**F :** $\forall$**b**$\in$**B :** $\exists$**U$_b$**$\subset$**B : f^{-1} (U$_b$)** **U$_b$**$\times$**F**

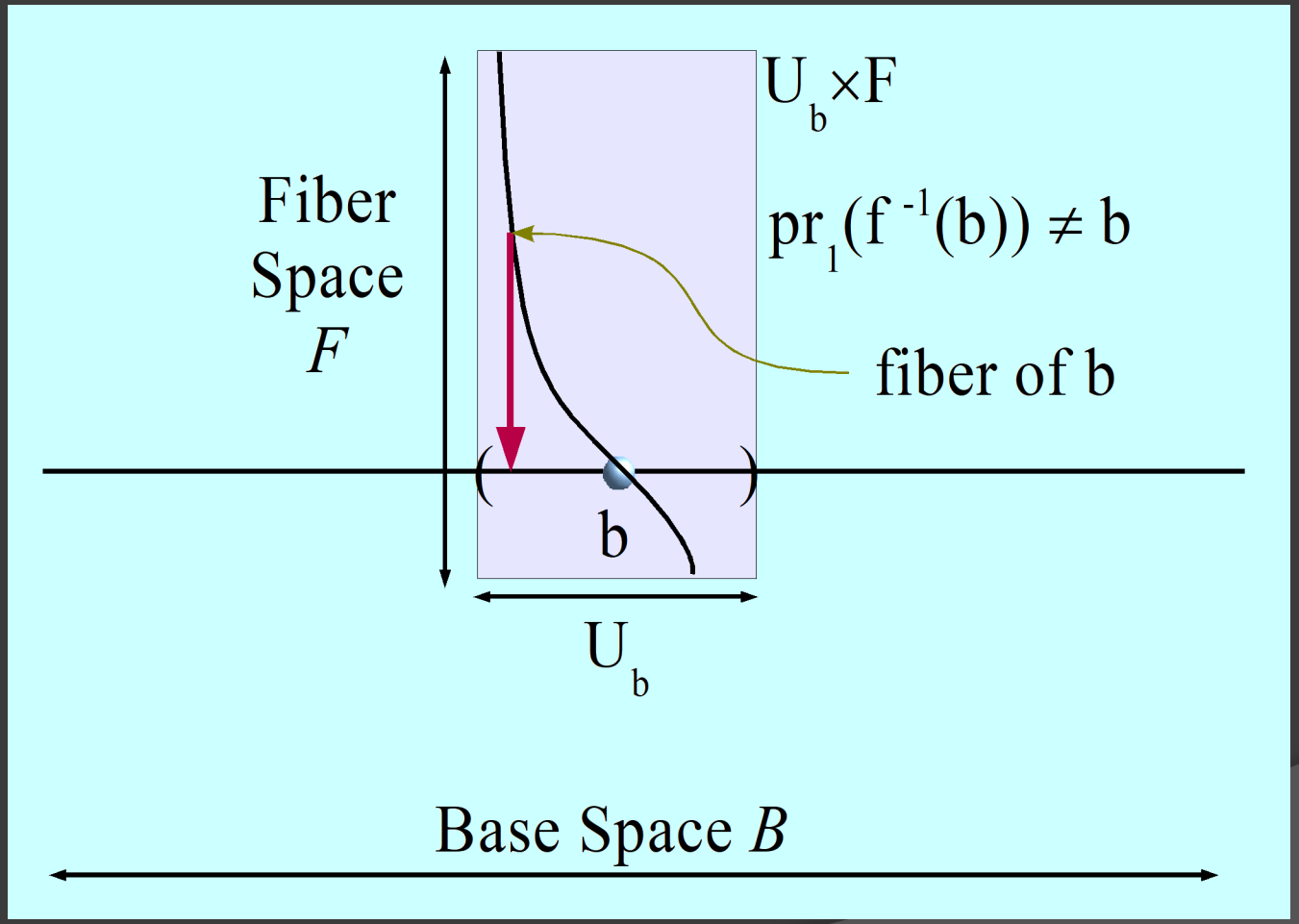
and **pr$_1$(U$_b$** $\times$ **F) = U$_b$**

- The space F is called the *fiber* of the bundle.

A Fiber Bundle Homeomorphism

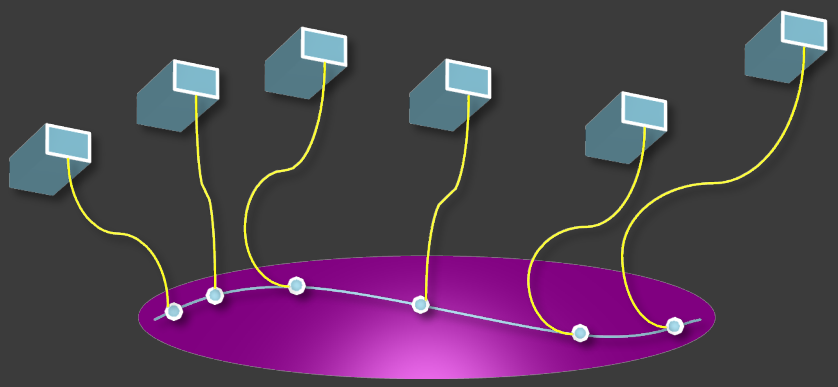


A Non-Fiber Bundle Homeomorphism



Fiber Bundle Data Model

- Is a generic approach to handle a wide range of data types used for scientific visualization
- Basic concept: Base space maps to fibers



Fiber Space at each point of Base Space

Base Space

Total Space for Data in SciViz

- Discretized
  - Neighborhood, Connectivity, k-Skeletons
  - Described via integer-valued arrays referring to indices
  - Numerical values are intrinsic, no dependence on coordinate system
- Continuous
  - Scalar, vector, tensor fields
  - Described via float point numbers
  - Numerical value depends on representation (coordinate system)
    - Exception: true scalar fields

BASE SPACE

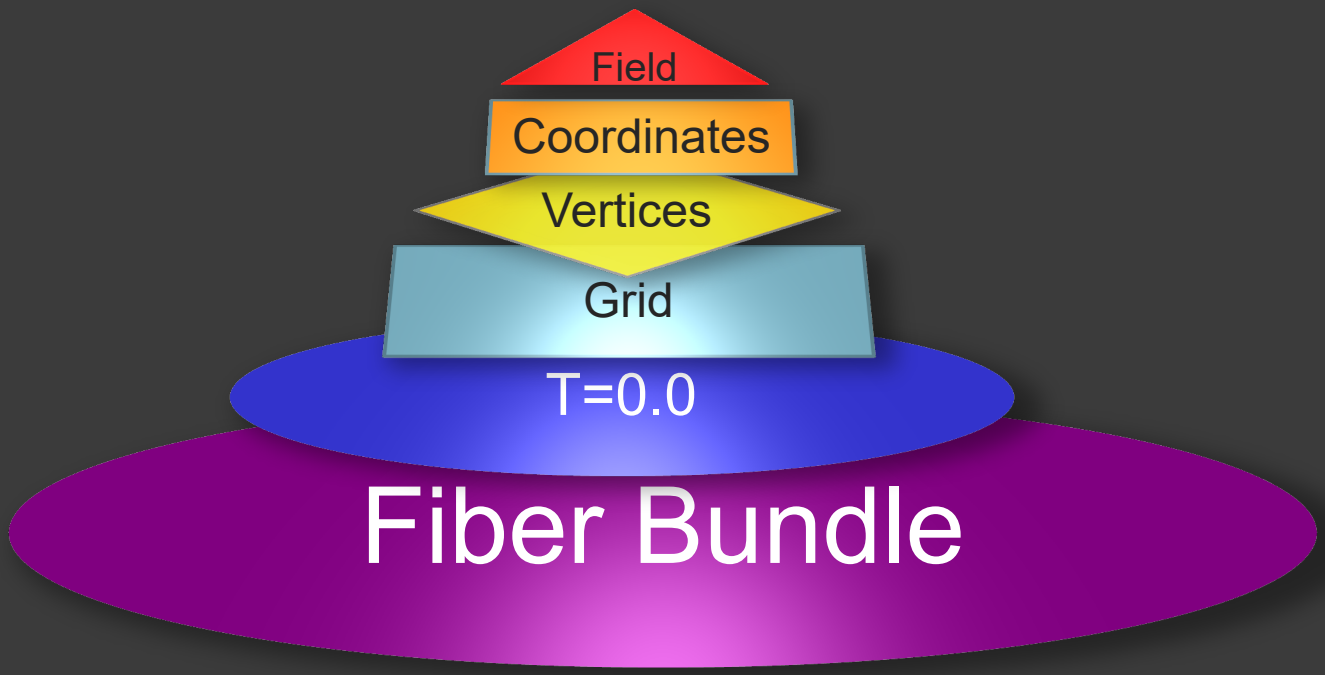
Fiber Space

“F5 Fiber Bundle Data Model”

- A property-based description of scientific data
- A specific data type is built from “construction blocks” (not enumerated case lists)
- Does not answer the question *“what is it?”* but answers the question(s) *“how is it?”*
- A specific data set can have properties of *many* data types
  - → non-exclusive properties
  - → eases/enhances interoperability
- Enables abstract operations independent of data type

The F5 Fiber Bundle Data Model

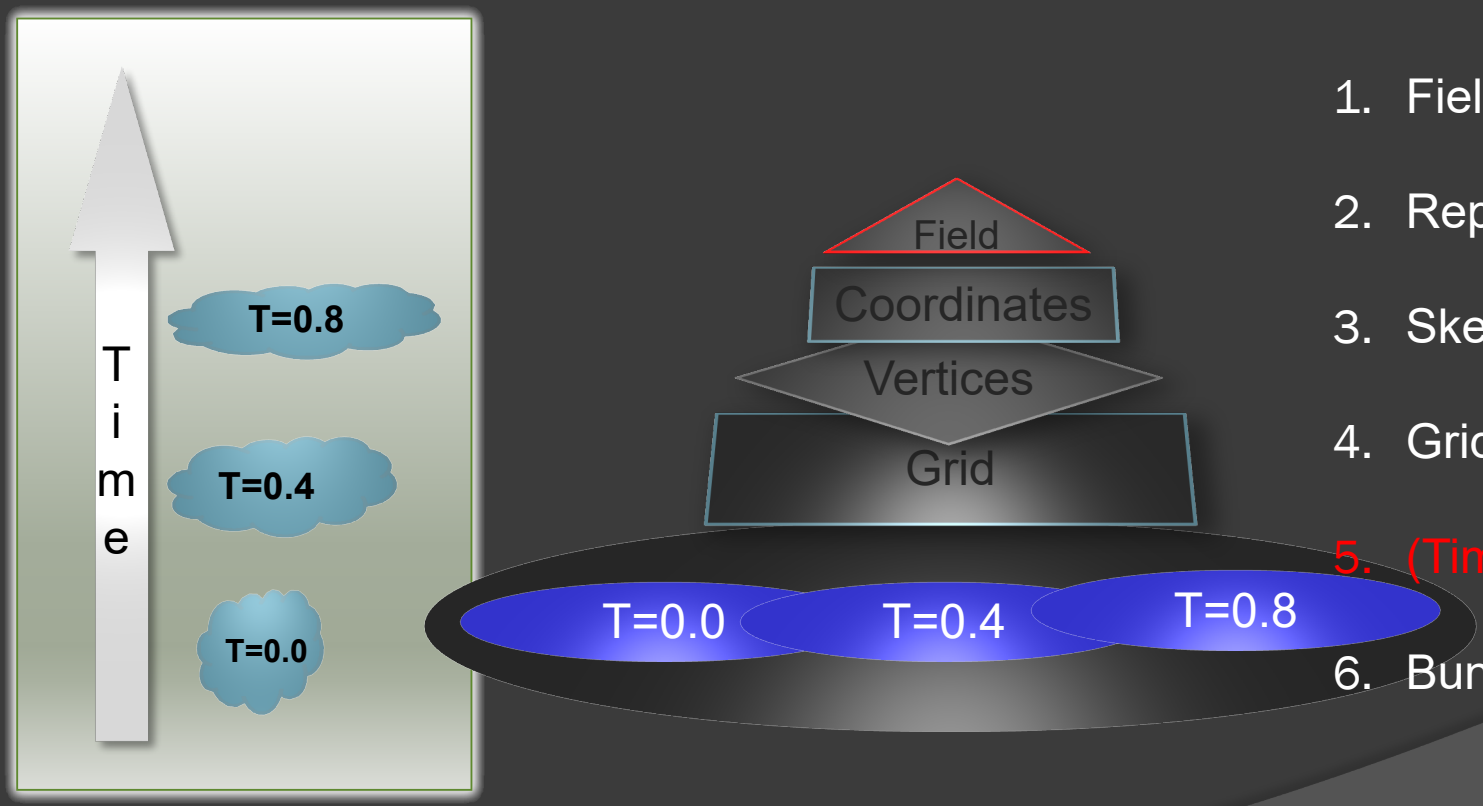
- Casts data into hierarchy:



1. Field
2. Representation
3. Skeleton
4. Grid object
5. (Time) Slice
6. Bundle

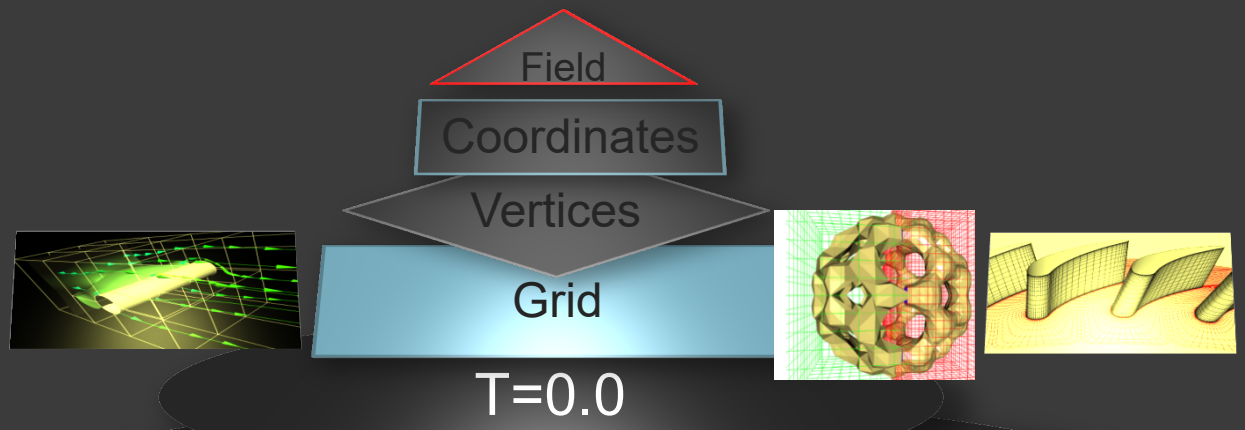
Lowest Level: Fiber Bundle over a Parameter Space

Similar to 3+1 Decomposition in Numerical Relativity



1. Field
2. Representation
3. Skeleton
4. Grid object
5. (Time) Slice
6. Bundle

Grid: A geometric entity

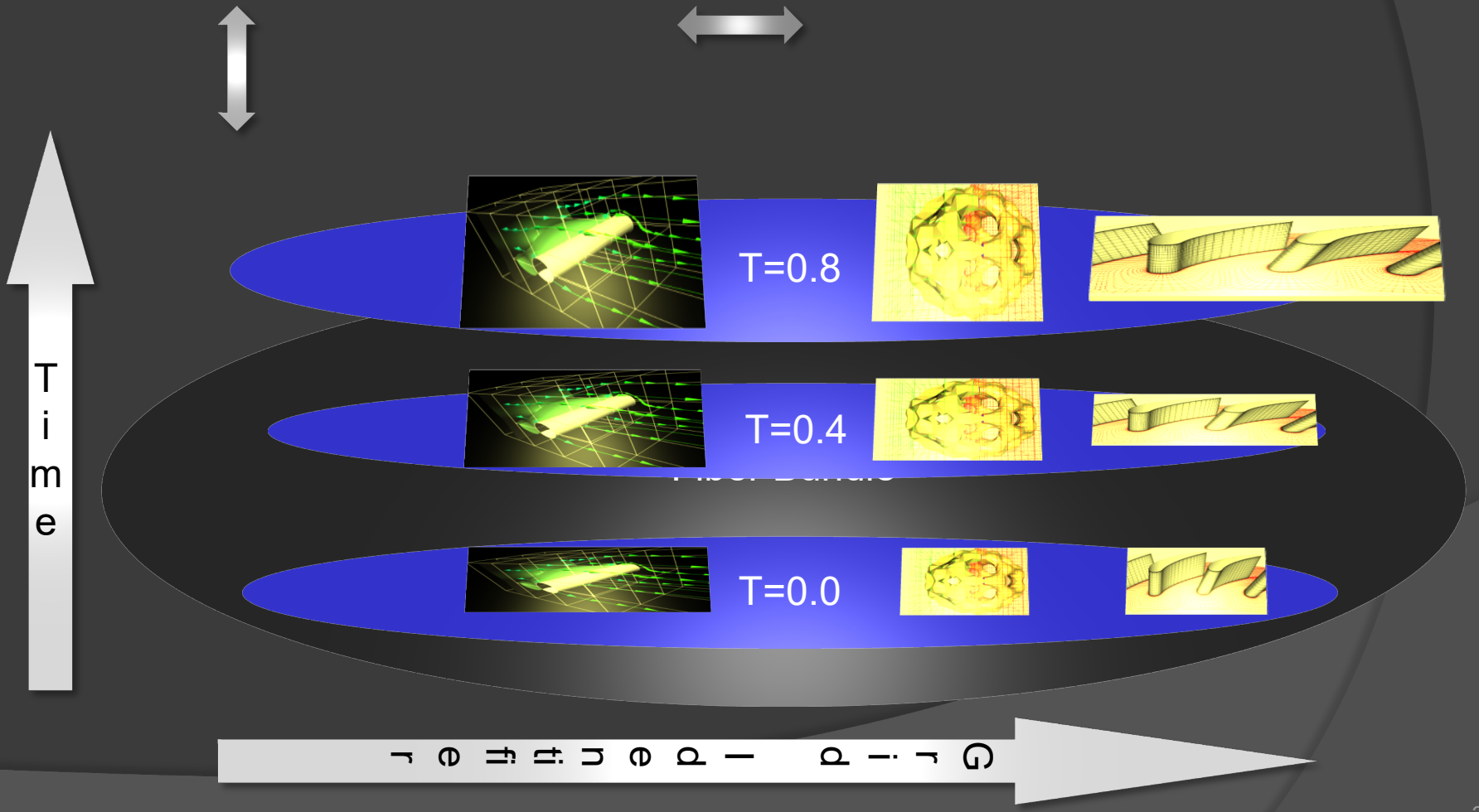


Fiber Bundle

1. Field
2. Representation
3. Skeleton
4. Grid object
5. (Time) Slice
6. Bundle

Grid Objects as Fibers of a two-dimensional Parameter Space

- Evolutions and Identifiers



Grid Iterations

Iterate over 1st dimension "time":

- Find all Grid objects that exist for a specific instance of (physical) time or time interval
- Iteration by physical time, not "time-step"

➓→ Easy synchronization of distinct data sources

Iterate over 2nd dimension "Grid ID"

- Find all times for which a Grid exists
- Temporal evolution of a Grid and its data
- Allows to trace features in Grid objects

Generalization of one-dimensional "time" to higher-dimensional parameter spaces possible

- Compare an ensemble of simulations

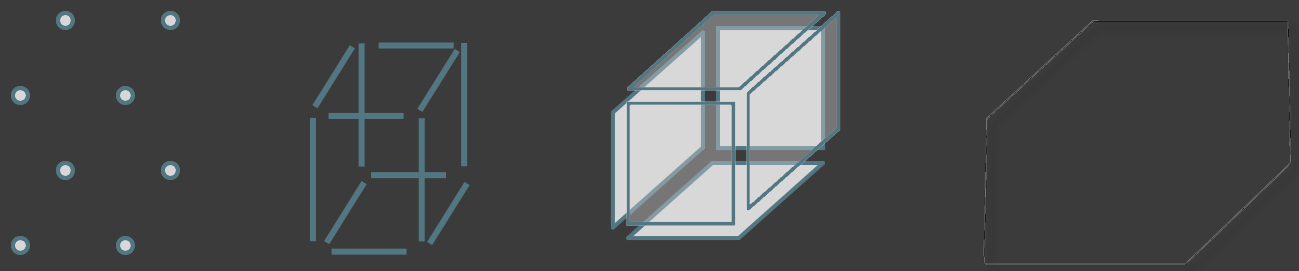
Structure of a Grid Object

Consider *discrete* base space:

1. Definition “k-cell”
2. Definition “incident”
3. Definition “adjacent”
4. Definition “n-Skeleton”
5. Definition “CW-Complex”

Topology: cw-Complex

- Discretized n-dimensional Manifold
- Concept of k-cells with adjacency
- Hierarchy of k-Skeletons, k=0…N



0-cell: Vertices 1-cell: Edges 2-cell: Faces 3-cell: simplexes

Definition “incident”

- Two cells c_1, c_2 are *incident* if $c_1 \subseteq \partial c_2$, where ∂c_2 denotes the border of the cell c_2.
- Two cells of the same dimension can never be incident because $\dim(c_1) \neq \dim(c_2)$ for two incident cells c_1, c_2.
- c_1 is a side of c_2 if $\dim(c_1) < \dim(c_2)$, which may be written as $c_1 < c_2$.

Definition: “adjacent”

- Two k−cells c_1, c_2 with $k > 0$ are called *adjacent* if they have a common side, i.e.

  cell c_1, c_2 adjacent $\Leftrightarrow \exists$ cell f : $f<c_1$, $f<c_2$.

- For k = 0 two 0-cells (i.e. vertices) v_1, v_2 are said to be adjacent if there exists a 1-cell (edge) e which contains both, i.e. $v_1 < e$ and $v_2 < e$.

Formal Definition: Skeleton

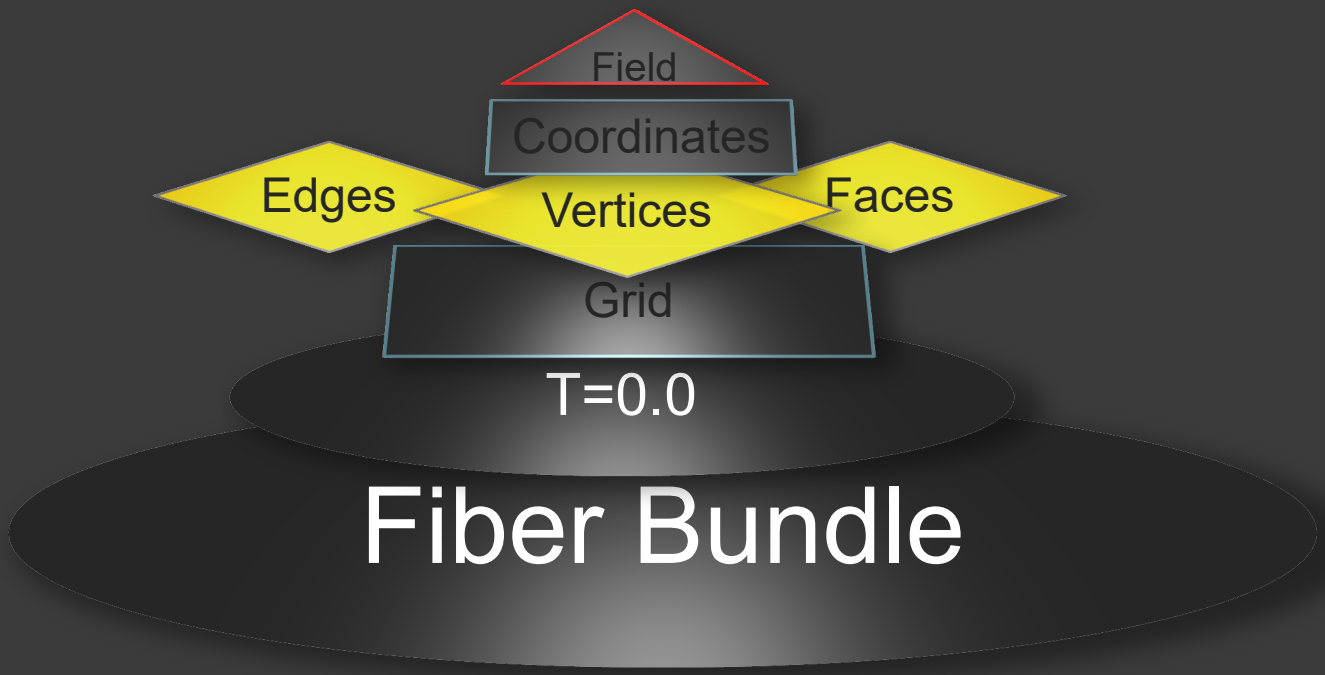
- A ***decomposition*** of a Hausdorff space X, is a hierarchical system of spaces $X^{(-1)} \subseteq X^{(0)} \subseteq X^{(1)} \subseteq \ldots \subseteq X^{(n)}$, constructed by pairwise disjoint open cells $c \subset X$ with the Hausdorff topology, such that $X^{(n)}$ is obtained from $X^{(n-1)}$ by attaching adjacent n-cells to each (n − 1)-cell and $X^{(-1)} =$.

- The respective subspaces $X^{(n)}$ are called the *n-skeletons* of X.

CW-Complex

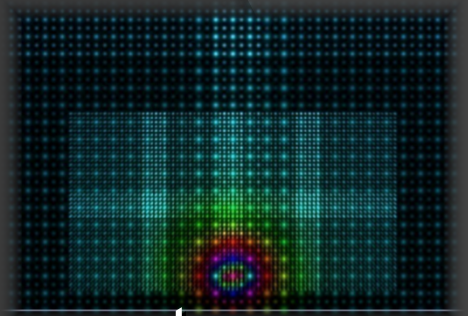
- A decomposition is also called a CW-complex.
- It can be understood as a set of cells which is glued together at their subcells.
- It is a generalization of the concept of a graph by adding cells of dimension greater than 1.

Topology: k-Skeletons

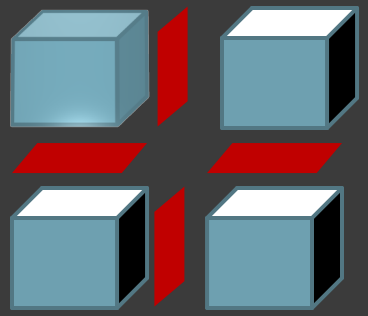


1. Field
2. Representation
3. Skeleton
4. Grid object
5. (Time) Slice
6. Bundle

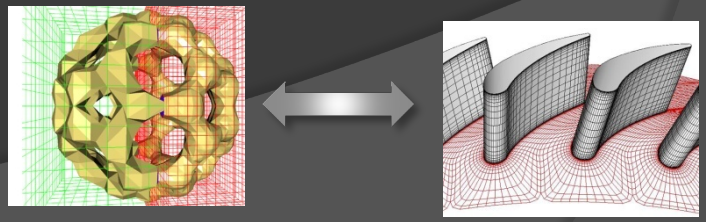
Extended Skeletons



- Refinement Levels
  - Multiresolution, Adaptive Mesh Refinement
- Cell Complex
  - Blocks, fragments

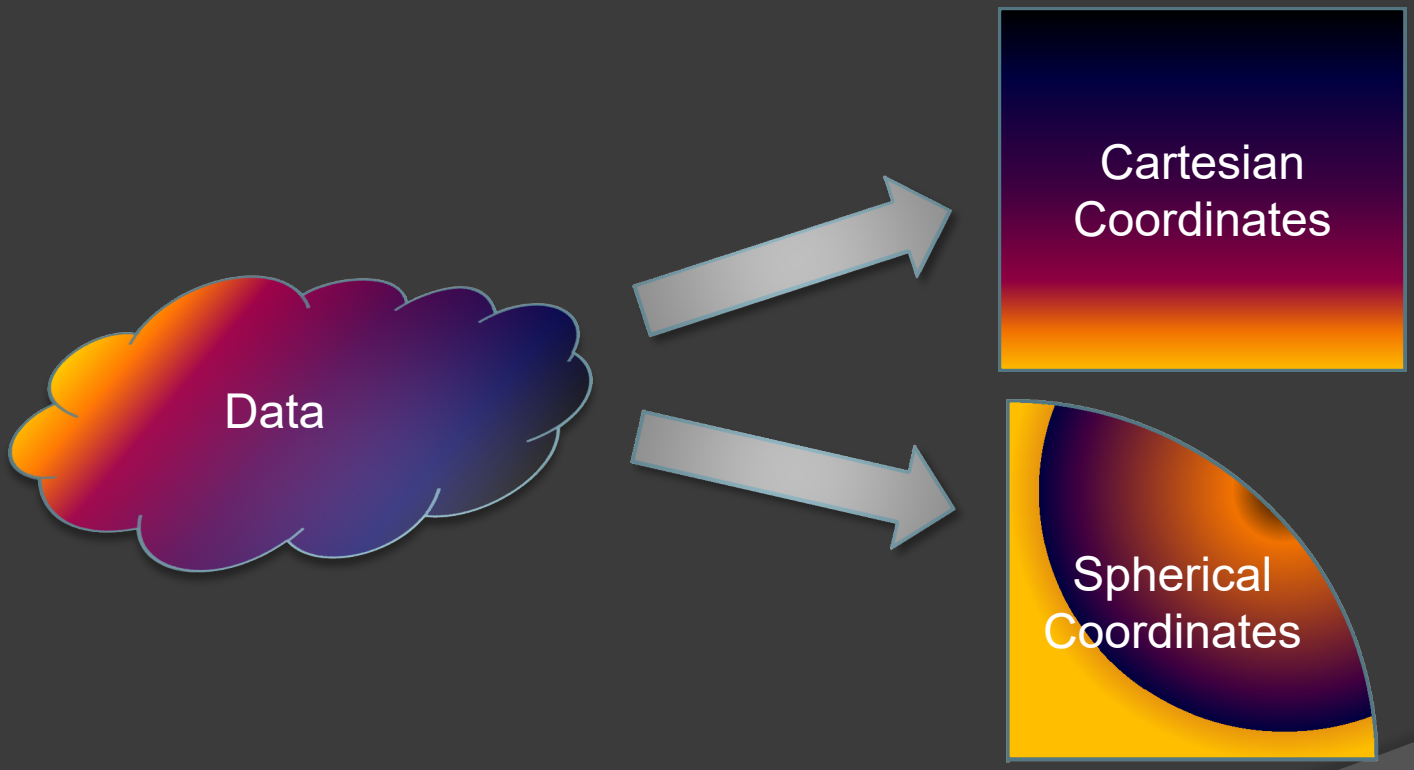


- Topological Relationships
  - Grid future/past
  - Intergrid Relationships



Differential Geometry

- Concept of a manifold and charts



Structure of the Fiber Space

- May carry
  - Scalar fields
  - Vector fields
  - Covector fields
  - Tensor fields
  - Pseudo-scalars
  - Multi-Vectors
  - Non-tensorial geometrical objects (e.g., Christoffel symbols)

Special cases of fiber space

- Has vector space property:
  - E.g. contains vectors, tensors:

    → Vector Bundle

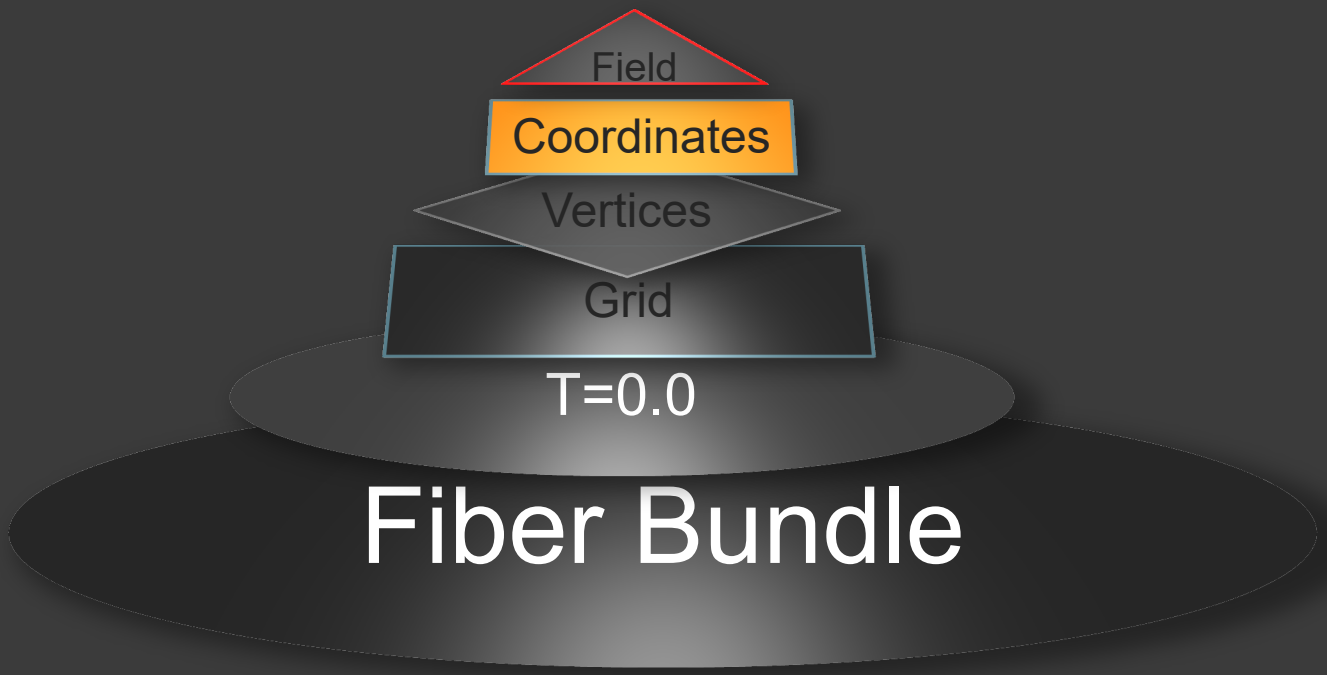
- Is the tangent space of a manifold M:

  → Tangent Bundle $M \times T(M)$

Representation

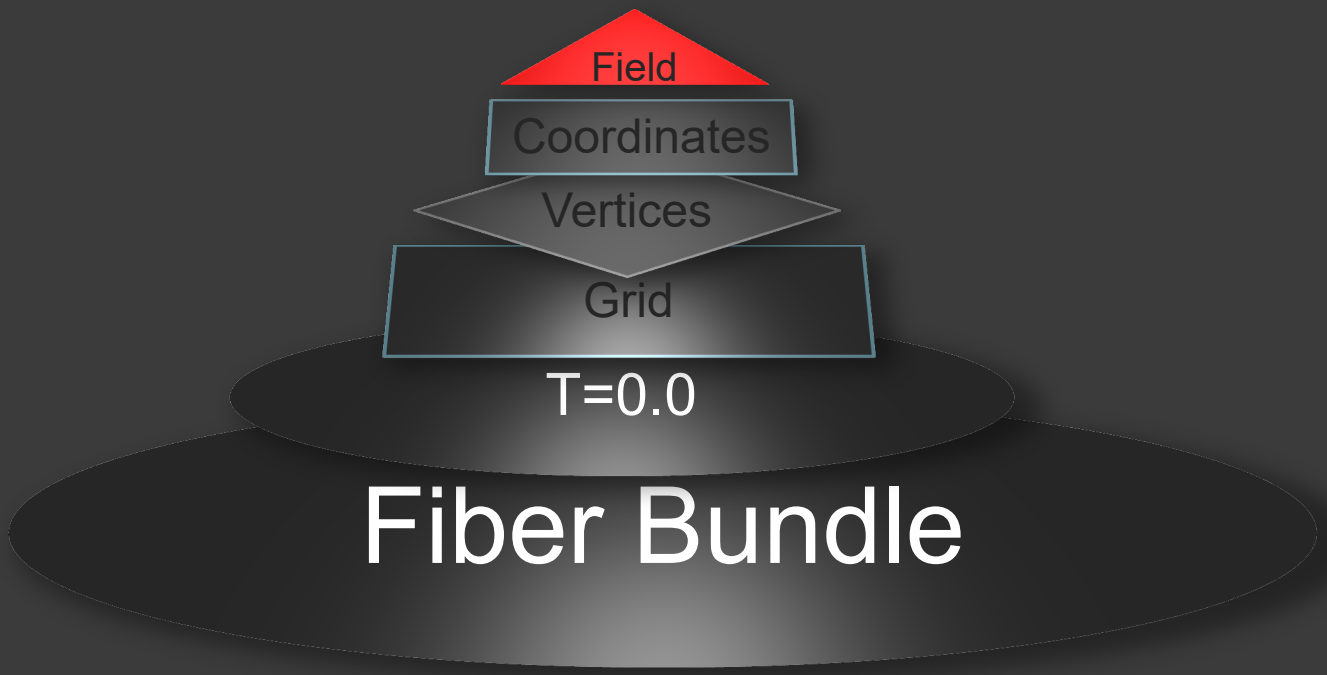
- Numerical representation via coordinates:
  - A manifold M is an Hausdorff space with (local!) maps $x:M \rightarrow \mathbb{R}^n$.
  - $x=\{x^0,x^1,x^2 \ldots x^{n-1}\}$ is called a *chart*
  - $x^m:M \rightarrow \mathbb{R}^n$ the m^{th} coordinate function
- A chart in the base manifold M induces a coordinate basis in the tangential space.

Multiple Representations



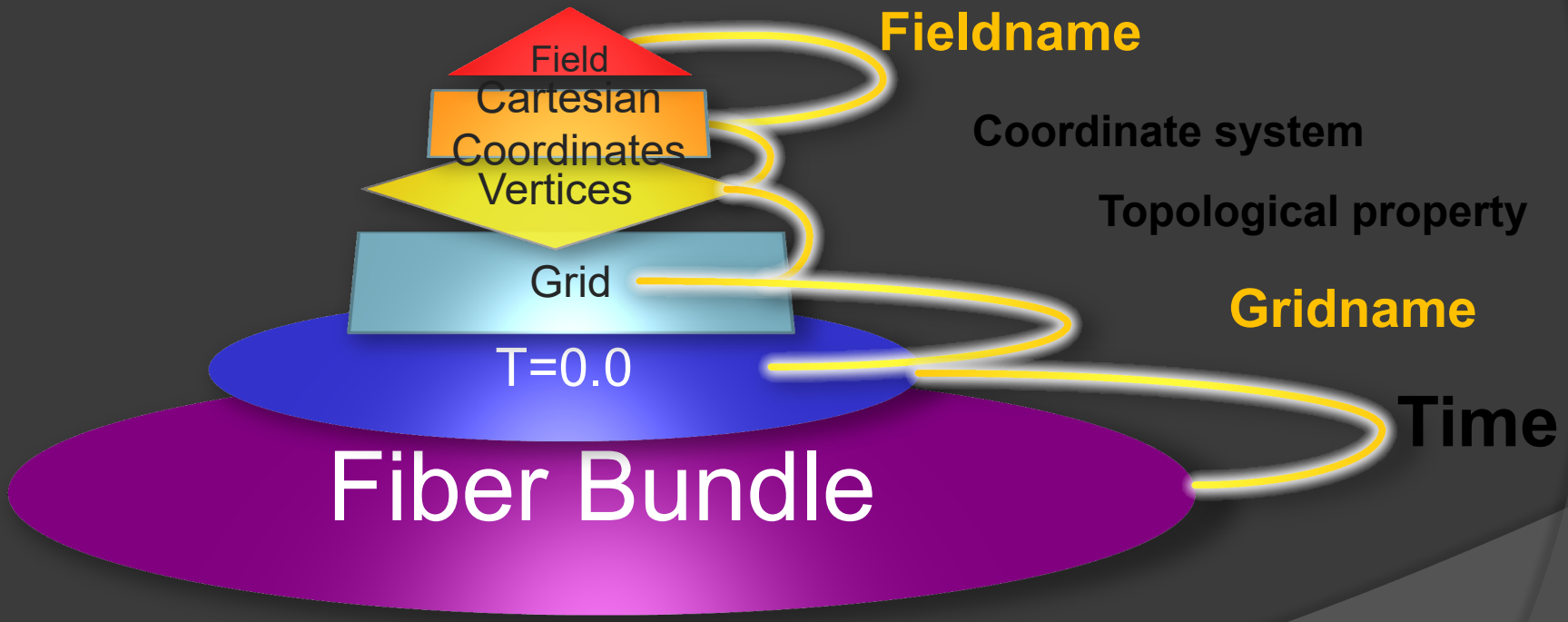
1. Field
2. Representation
3. Skeleton
4. Grid object
5. (Time) Slice
6. Bundle

Geometric Algebra: Vectors, Tensors, Spinors, ...



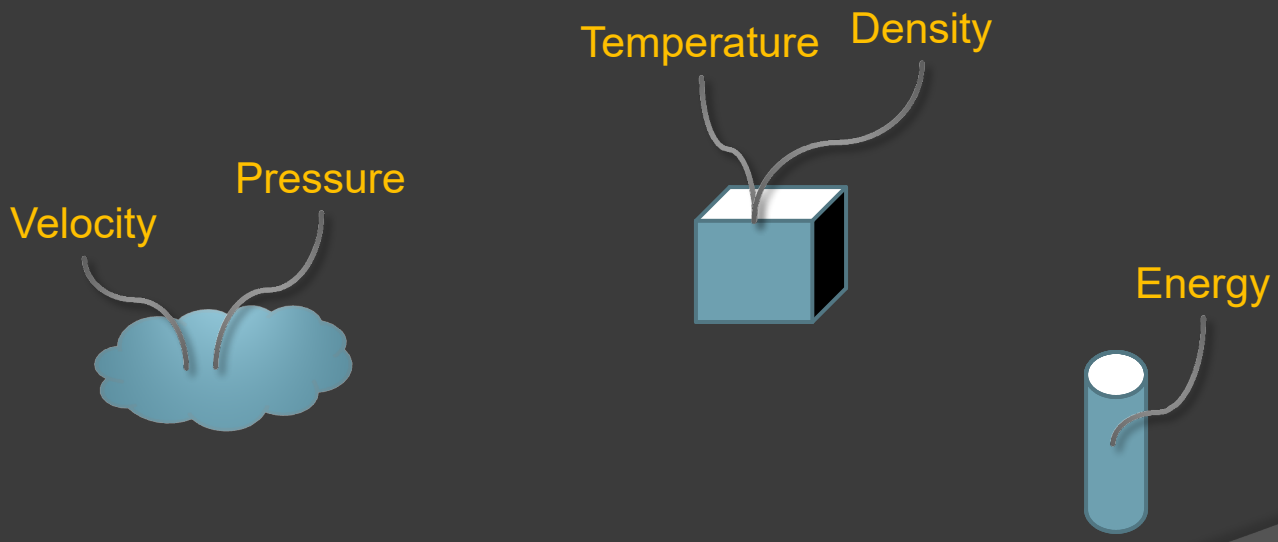
1. Field
2. Representation
3. Skeleton
4. Grid object
5. (Time) Slice
6. Bundle

Access via Parameters



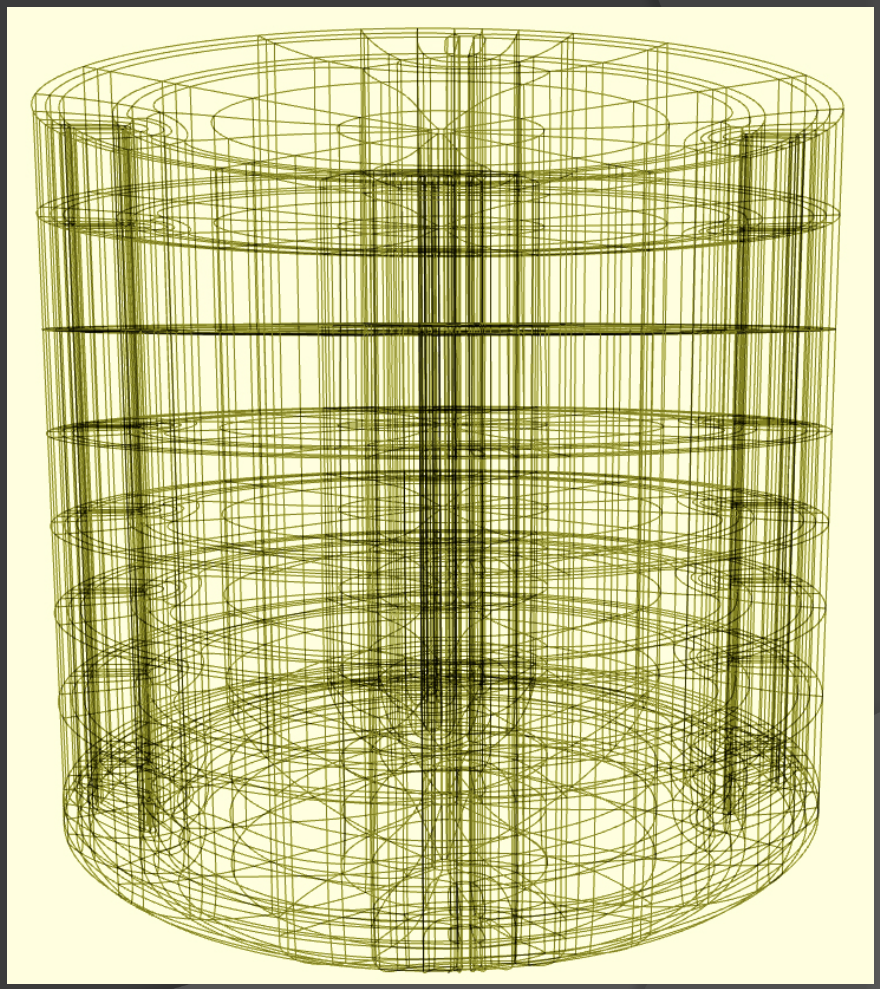
Grids and Fields

- End-user operates only on Grid objects and Fields

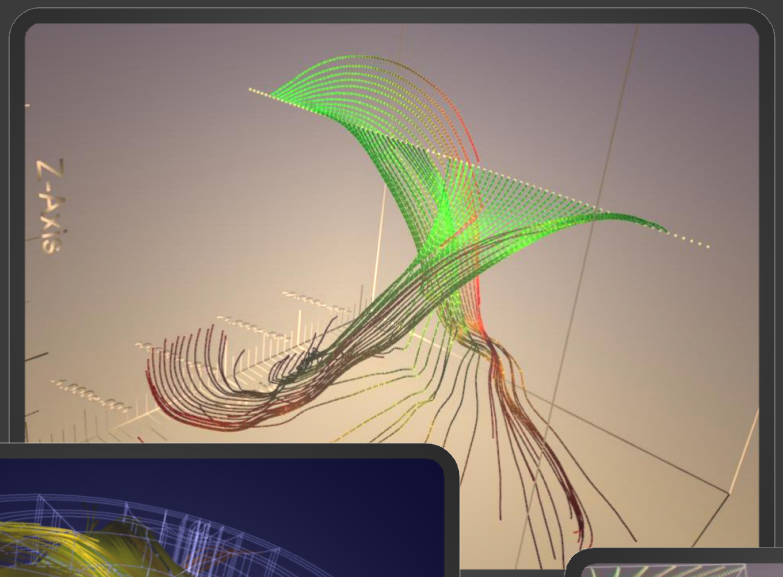


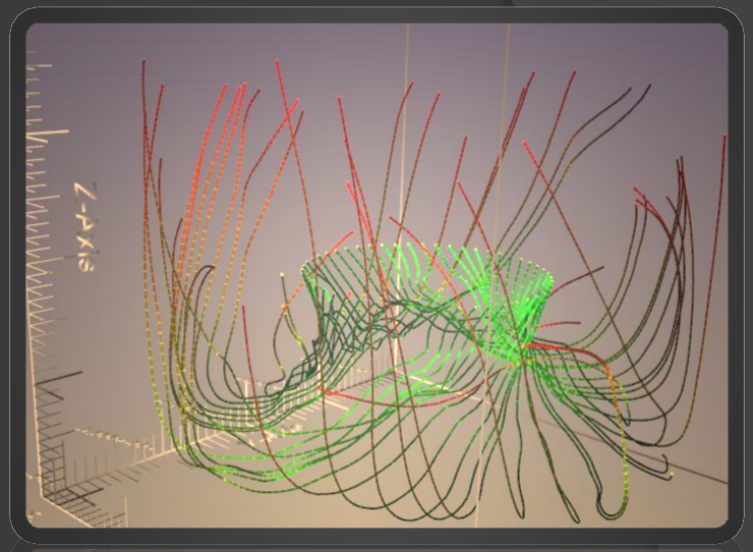
Application Example: Multiblock Curvilinear Data

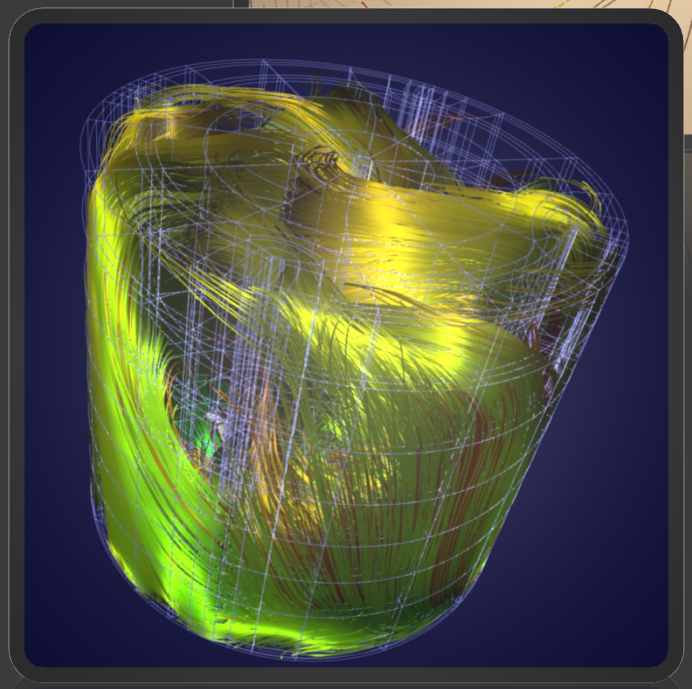
- The Stirtank Dataset
  - 2088 curvilinear blocks
  - 5000 time steps
  - Vectorfield describing velocity
  - Scalarfield describing pressure
  - 500GB Binary Data

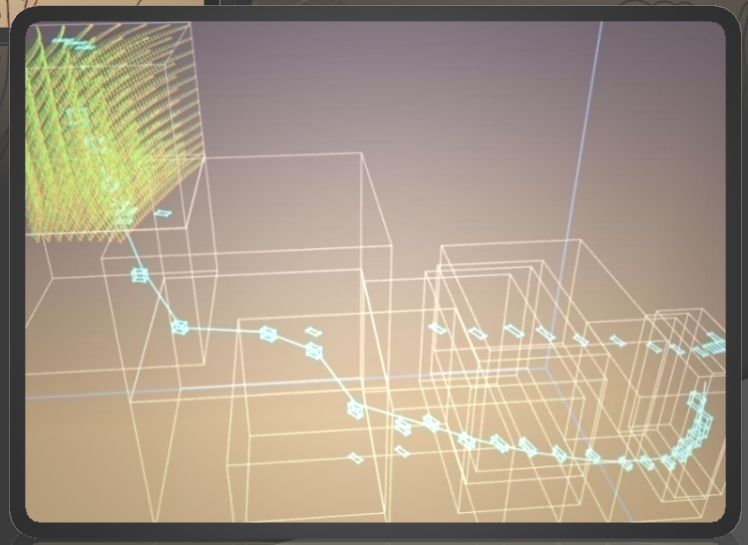


Streamlines



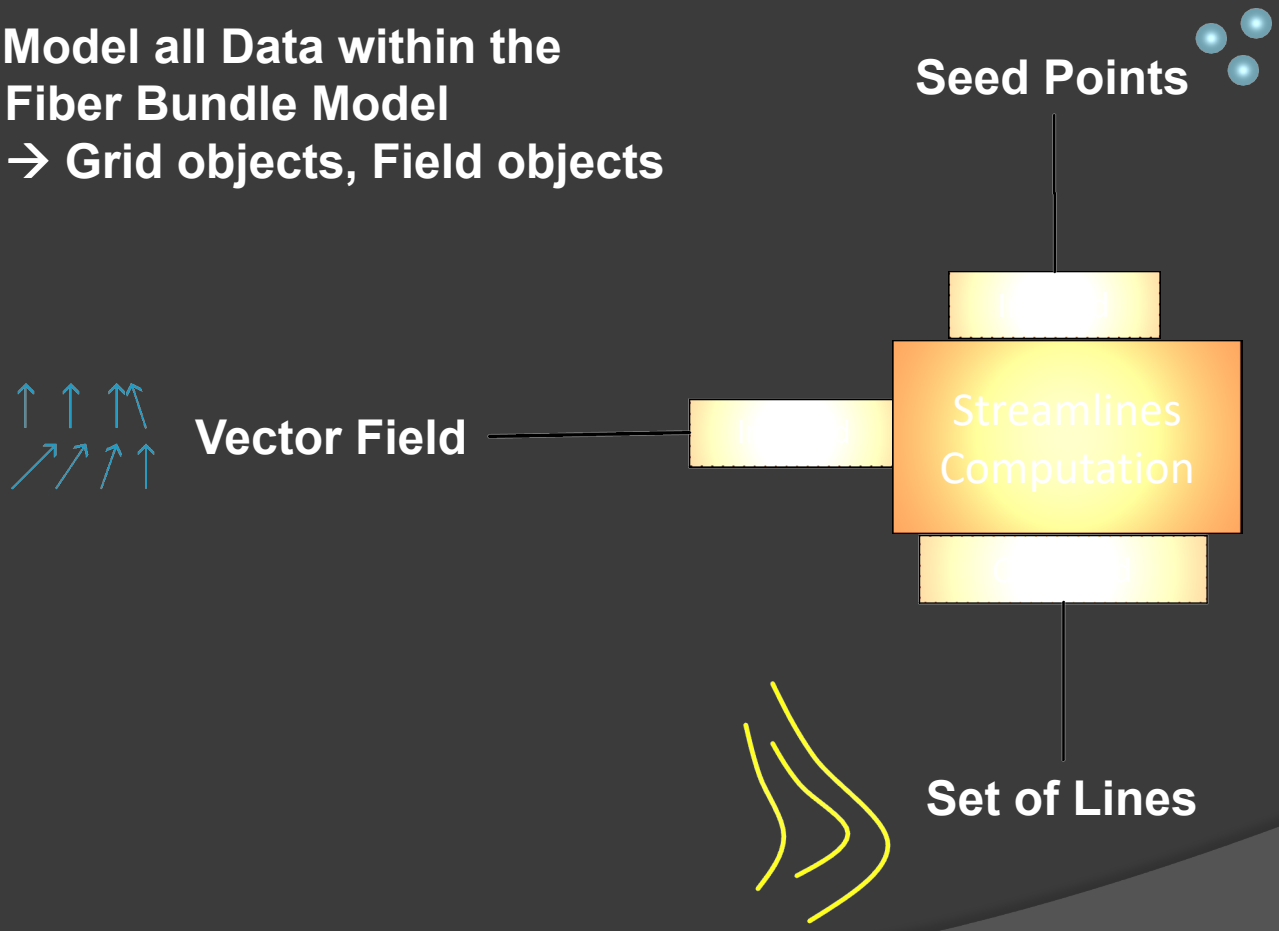






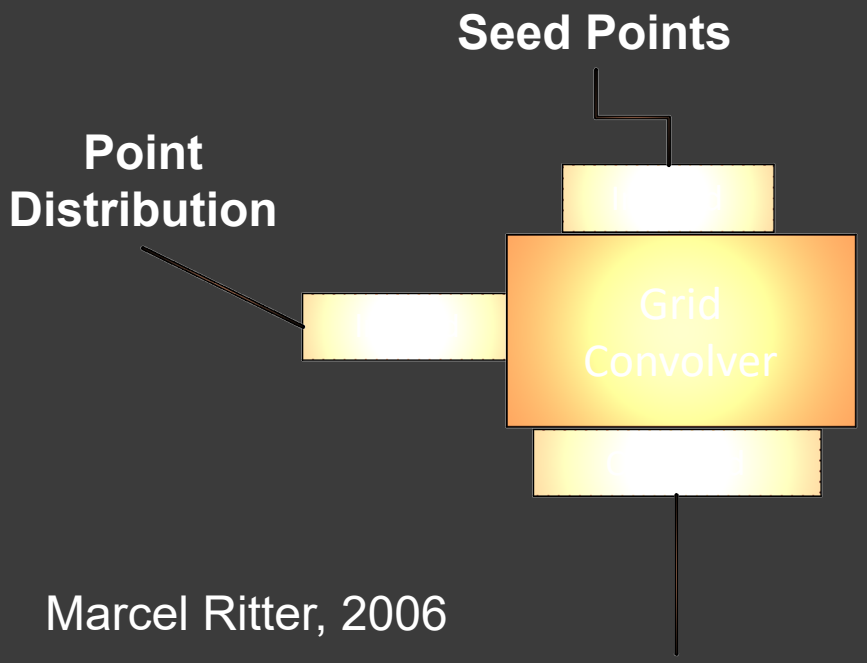
Streamlines in the Fiber Bundle

Model all Data within the Fiber Bundle Model
→ Grid objects, Field objects

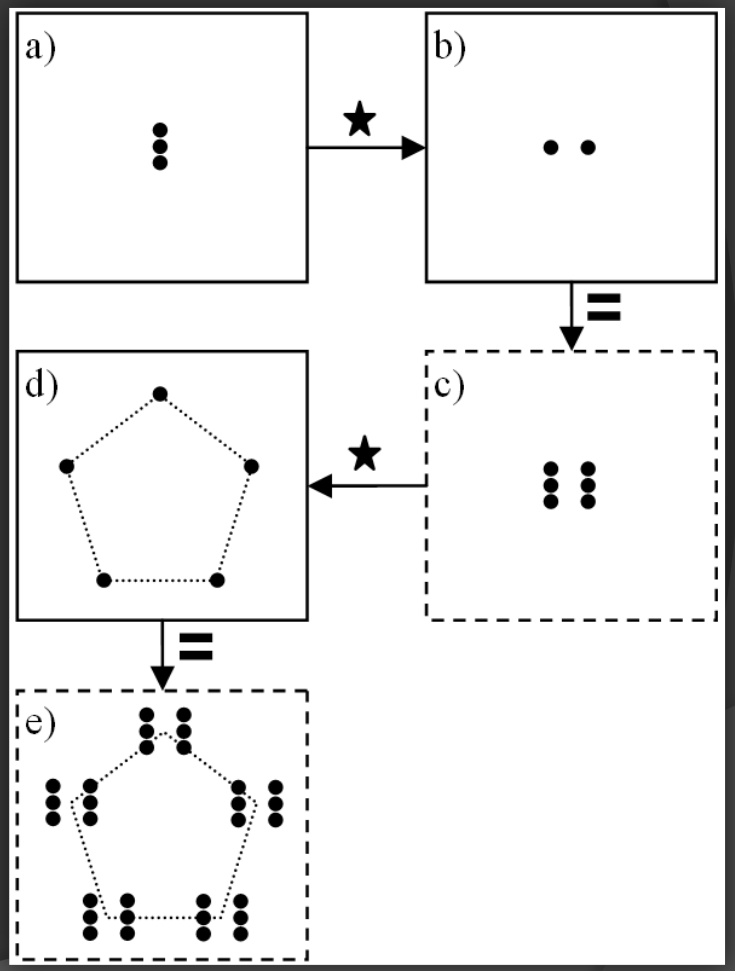


“Convolution” of Grids

Replicate Grid by points on another Grid

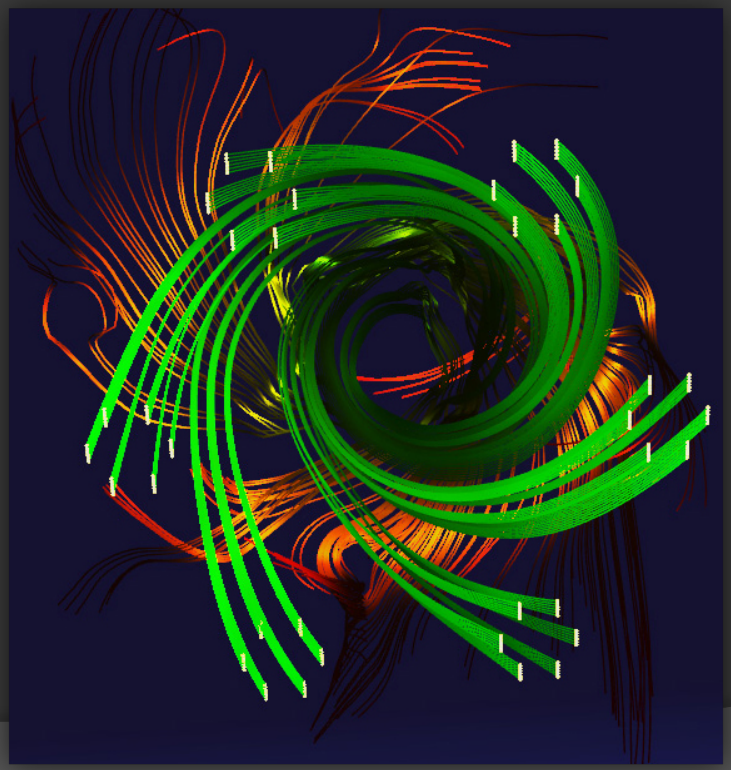


Marcel Ritter, 2006



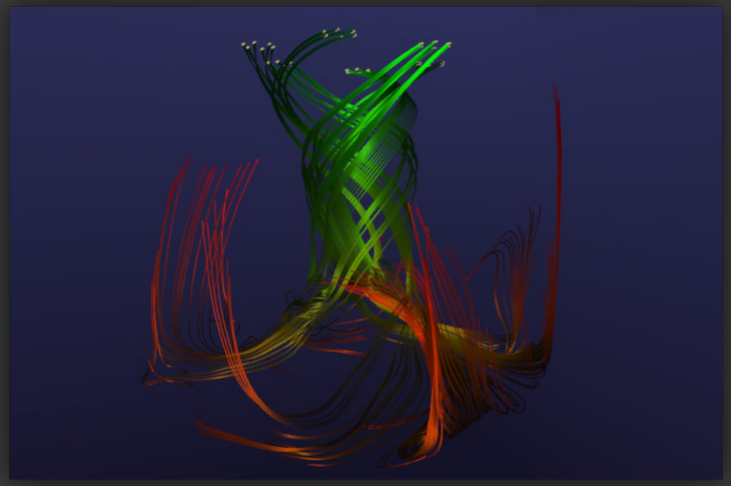
Systematic Approach

Grid objects and Fields as entities within a visualization environment



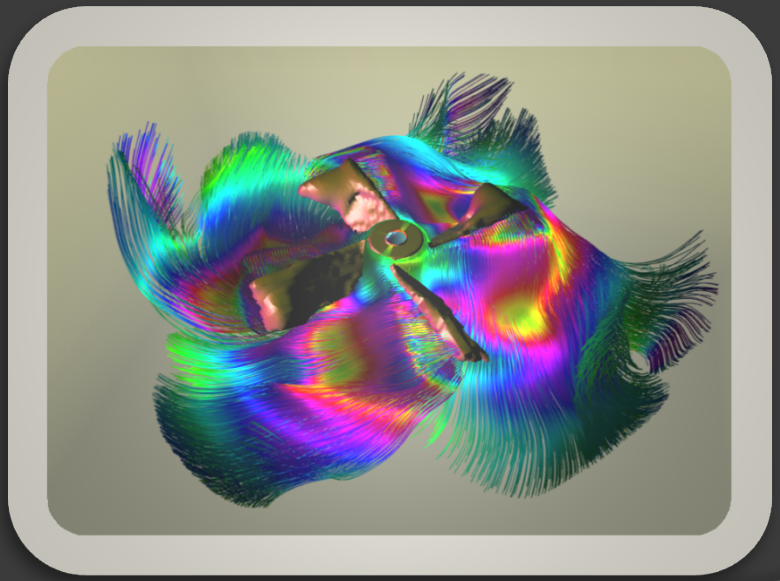
Systematic Approach

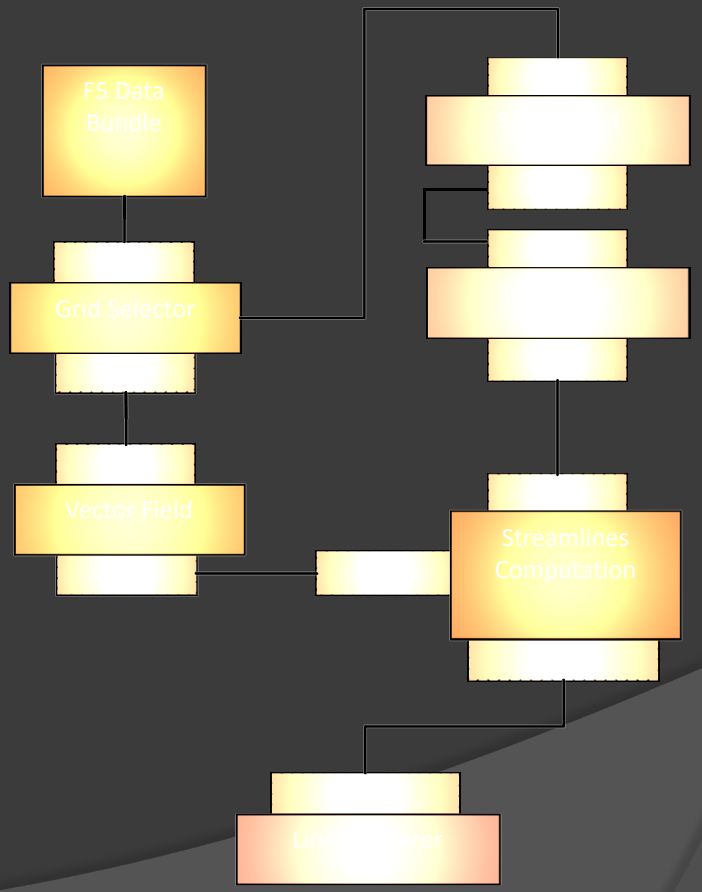
Grid objects and Fields as entities within a visualization environment



Systematic Approach

Seeding Streamlines from an Isosurface





F5 as file format and I/O library

F5 – Fiber Bundle HDF5

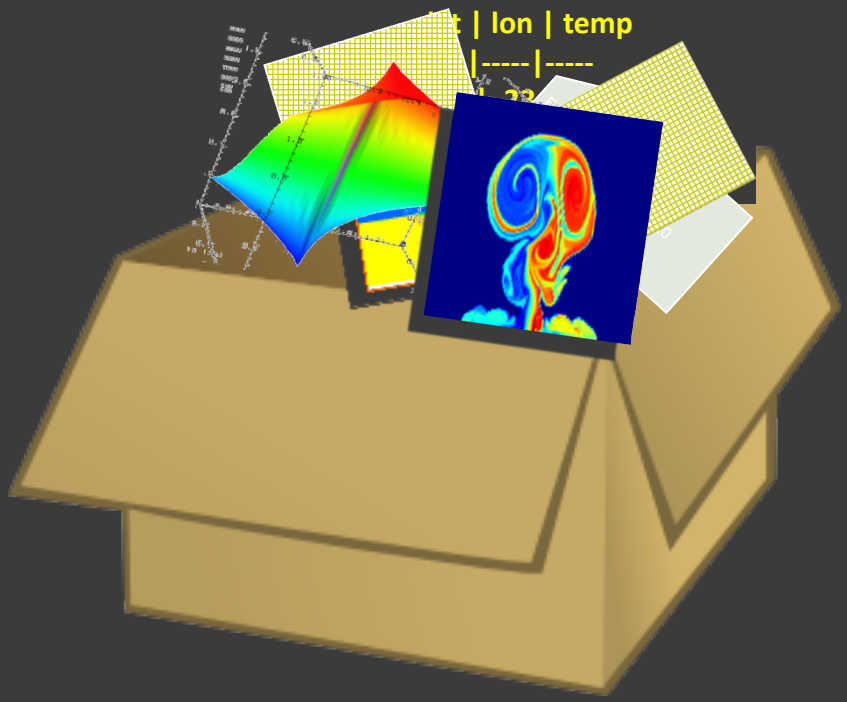
HDF = Hierarchical Data Format

- HDF5 is the second HDF format
  - Development started in 1996
  - First release was in 1998
  - http://www.hdfgroup.org
- Designed for HPC simulations
- High-performance, large data, long-term data preservation (archival), portability

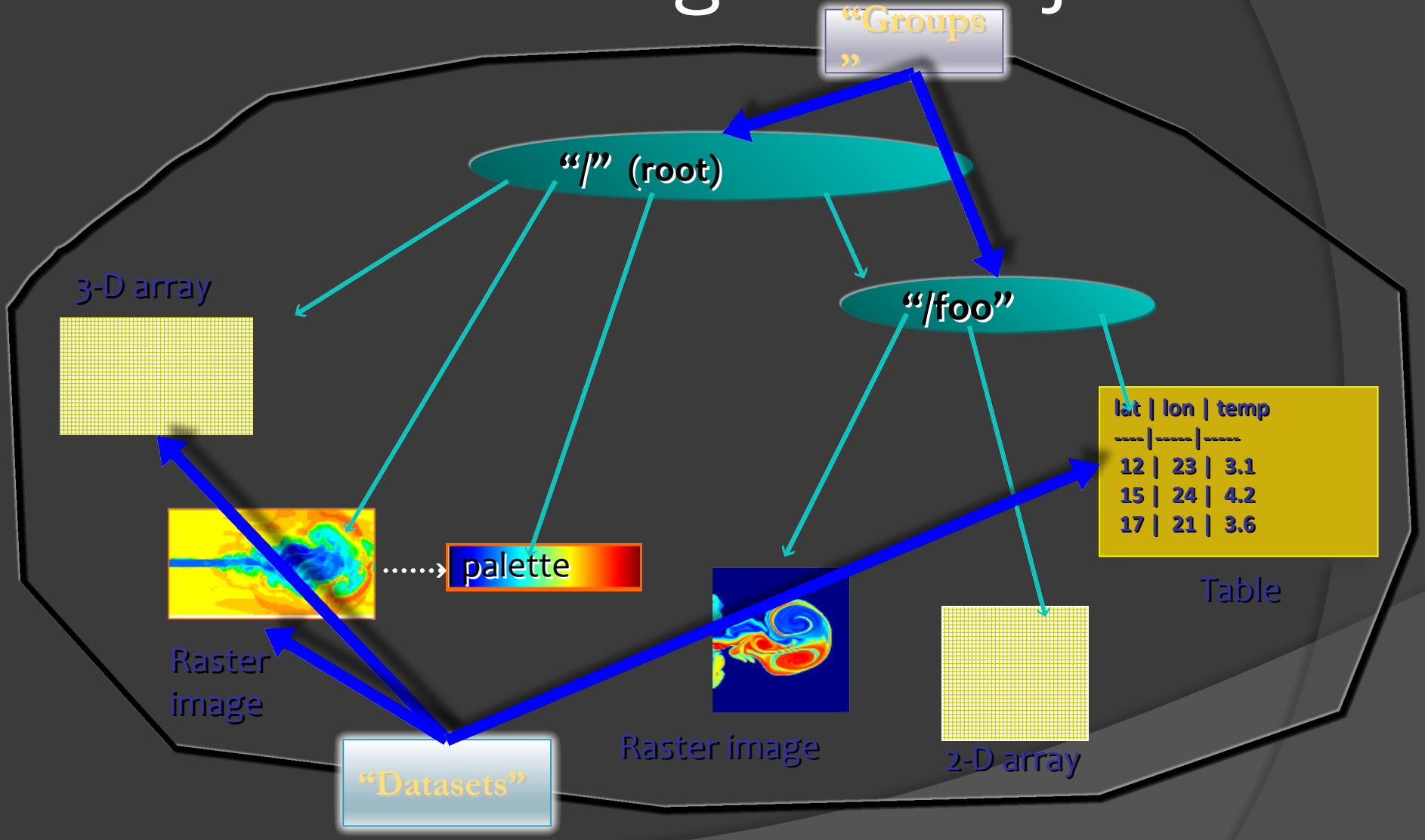
- F5:Casting Fiber Bundle Data Model into HDF5

HDF5 File

An HDF5 file is a **container** that holds data objects.

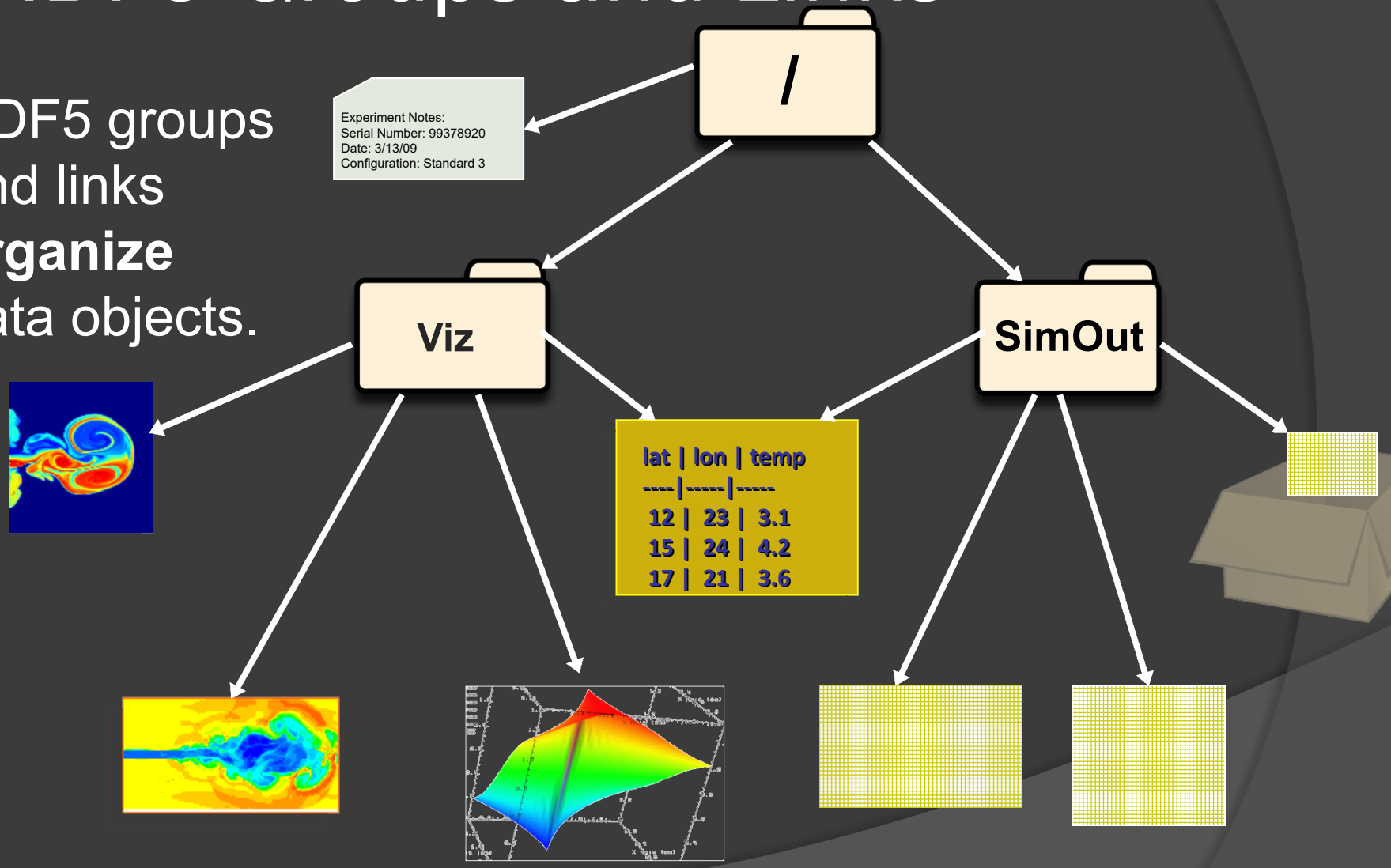


Structures to organize objects



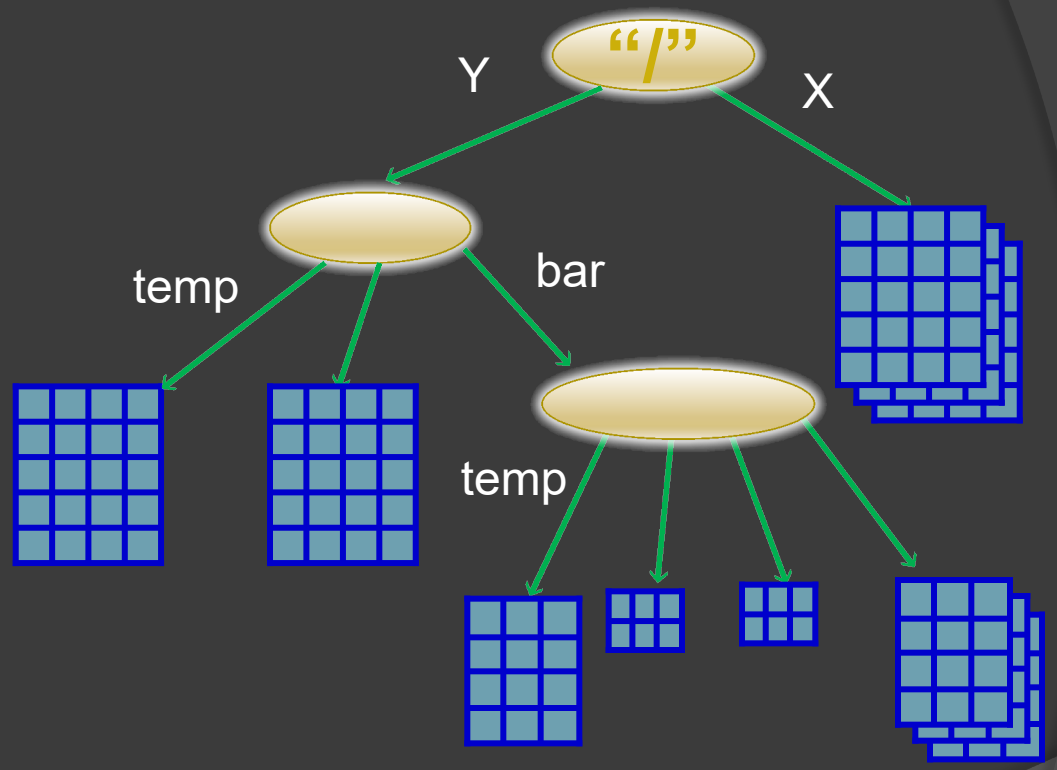
HDF5 Groups and Links

HDF5 groups and links **organize** data objects.



Path to HDF5 object in a file

/ (root)
/X
/Y
/Y/temp
/Y/bar/temp



Example Dataset

The internal HDF5 file organisation made visible using the **h5ls** command: h5ls –rv file.h5

```
/Block00001 Dataset {5/5, 13/13, 9/9}
    Location: 1:15768
    Links:     1
    Storage:   7020 allocated bytes
    Type:      struct {
                   "x"      +0    native float
                   "y"      +4    native float
                   "z"      +8    native float
               } 12 bytes
    Data:
      (0,0,0) {0.210951, -0.0406732, 0.0611351},
              {0.210204, -0.0443333, 0.0611199},
              {0.209324, -0.0483009, 0.0611070},
              {0.208286, -0.0525892, 0.0610958}
      (0,0,4) {0.207065, -0.0571980, 0.0610863},
              {0.205640, -0.0621138, 0.0610815},
```

HDF5 allows easily to attach names to the numerical values.

HDF5 takes care of endianess, double to float conversion and different layouts, e.g. {x,y,z} vs {z,x,y}

The naming scheme can also serve to identify the coordinate system rel. to which the numbers are stored e.g. {x,y,z} vs {r, phi, theta}

HDF5 as a file system in a file

- Group
- Dataset
  - Size
  - Creation date
  - Dimensionality
  - Data type (int, float, …)
- Symbolic Link
- Attributes
- Named data types
- Mounting external data
- Virtual File Driver (stdio, mpi-io, streaming VFD,…)

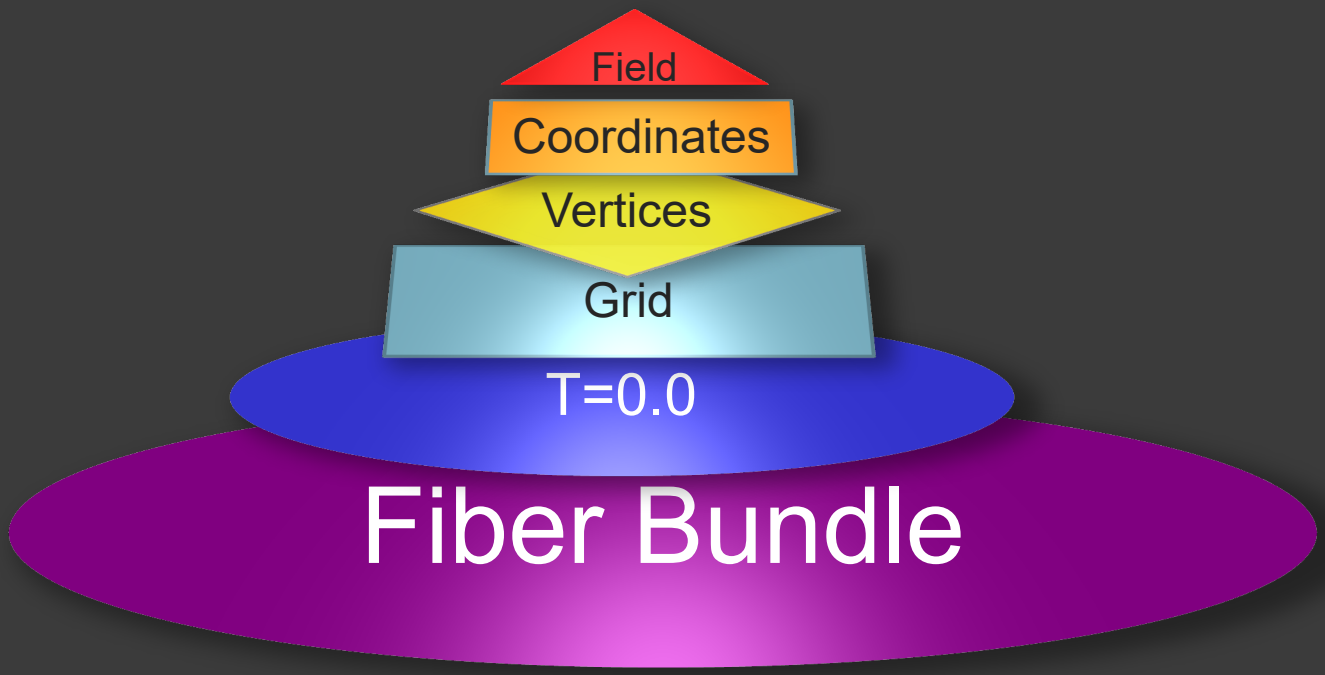
HDF5

- Directory, Folder
- File
  - Size
  - Creation data
  - -
  - -
- Symbolic Link, Shortcut
- -
- -
- Mounting file systems
- File system type (ext3, ntfs, nfs,…)

File System

F5 File Format

- Casting 6 hierarchy levels into HDF5



1. Field
2. Representation
3. Skeleton
4. Grid object
5. (Time) Slice
6. Bundle

/Time/Grid/Top/Rep/Field/

```
h5ls -r minkowski-0000.f5
/T=0/minkowski/Points/StandardCartesianChart3D Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::LAPSE Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/Positions Group
/T=0.11904761904761904101057723437 4 Group
/T=0.11904761904761904101057723437 4/minkowski Group
```

F5 Hierarchy in HDF5

1. Time Slices (parameter space)
2. Grid (geometrical entity)
3. Topology → n-Skeletons
4. *Representation* (coordinate systems,
5. ***Field*** (denotes data sets)
6. *Field fragments* (denotes data layout)
7. Compound component (alternative layout for compound data)

/Time/Grid/Top/Rep/Field/Fragment

- Slicing of a spacetime
- Bundles all entities that belong to a certain physical time
- Encompasses all information that might relates to a certain time
- May contain multiple geometries
- Constraints on HDF5 file:
  - No datasets contained on this level
  - Group must have floating-point attribute “time”

Example: h5ls -r

```
h5ls -r minkowski-0000.f5
/T=0/minkowski/Points/StandardCartesianChart3D Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::LAPSE Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/Positions Group
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS Group
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dz Dataset {43, 37, 19}
/T=0.11904761904761904101057723437 4 Group
/T=0.11904761904761904101057723437 4/minkowski Group
```

/Time/Grid/Top/Rep/Field/Fragment

- Description of a geometrical entity
- Bundles all properties that belong to a certain object
- Encompasses all information that is or might be required to perform operations on the certain object
- … is a spatial manifold (or topological space) plus fiber space
- Constraints on HDF5 file:
  - No datasets contained on this level
  - Group must have at least one subgroup fulfilling the constraints of a “Vertex Skeleton”

Example: h5ls -r

```
h5ls -r minkowski-0000.f5
/T=0/minkowski/Points/StandardCartesianChart3D Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::LAPSE Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/Positions Group
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS Group
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dz Dataset {43, 37, 19}
/T=0.11904761904761904101057723437 Group
/T=0.11904761904761904101057723437/minkowski Group
```

/Time/Grid/Top/Rep/Field/Fragment

- Topological properties of a Grid
- One entry for each k-Skeleton:
  - Vertices, Edges, Faces, Cells
- Contains *Index spaces*:
  - Discrete sets with properties
    - Color per vertex, color per face, color per cell
  - Cell Complexes: sets of cells
- Constraints on HDF5 file:
  - Attribute “index depth”
  - All datasets in this group or below must have same number of elements and dimensionality
  - Optional attribute “refinement”

Example: h5ls -r

```
h5ls -r minkowski-0000.f5
/T=0/minkowski/Points/StandardCartesianChart3D Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::LAPSE Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/Positions Group
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS Group
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dz Dataset {43, 37, 19}
/T=0.11904761904761904101057723437 Group
/T=0.11904761904761904101057723437/minkowski Group
```

/Time/Grid/Top/Rep/Field/Fragment

- Representation of Skeleton relative to "something", e.g.:
  - Charts (coordinate systems)
  - N-Skeletons (cells as vertices, faces as vertices, vertices as cells)
  - Cell complexes (cells per complex, complexes per cell)
  - Another Grid (interpolation weight)
- Constraints on HDF5 file:
  - The name of the group must refer to an existing entity, such as a coordinate system identifier or another Skeleton group

Example: h5ls -r

```
h5ls -r minkowski-0000.f5
/T=0/minkowski/Points/StandardCartesianChart3D Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::LAPSE Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/Positions Group
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS Group
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dz Dataset {43, 37, 19}
/T=0.119047619047619041010577234374 Group
/T=0.119047619047619041010577234374/minkowski Group
```

/Time/Grid/Top/Rep/Field/Fragment

- A set of numerical values
- Positional information in the base space
  - Vertex coordinates or vertex indices per cell
- Field data in the fiber space
  - Scalar, vector, tensor, …
  - Color per cell, color per vertex
- Constraints on HDF5 file:
  - An entries may be *n*-dimensional dataset or a group; if it is a group, it needs to be identifiable as either
    1) Fragmented field (contains sequence of *n*-dimensional datasets)
    2) Compound data type with *k* separated components (contains *k* *n*-dimensional scalar-valued datasets)
    3) Compound data type constructed as direct product of components (contains n one-dimensional scalar-valued datasets)
    4) Linear field or polynomial expression

Procedural fields

- Storage of field via group with attributes instead of dataset
- Value per point computed procedurally on demand
- Examples:
  - Equidistant coordinates (linear map of index to coordinate values)
  - Storage of compound data as separated components (XYZXYZ ←→ XXYYZZ)

Special Field Names

- Reserved field name “Positions”
- Stores coordinate locations
  - Coordinates in physical space
    - Cartesian,
    - Spherical
    - Cylindrical
  - Coordinates in index space
    - Vertex indices per cell
    - Edges per cell

Example: h5ls -r

```
h5ls -r minkowski-0000.f5
/T=0/minkowski/Points/StandardCartesianChart3D Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::CURV/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::LAPSE Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC Group
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gzz Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/Positions Group
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS Group
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dx Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dy Dataset {43, 37, 19}
/T=0/minkowski/Points/StandardCartesianChart3D/STATICCONFORMAL::CONFAC_1DERIVS/Dz Dataset {43, 37, 19}
/T=0.119047619047619041010577234374 Group
/T=0.119047619047619041010577234374/minkowski Group
```

Multiple Coordinate Systems

/T=0/minkowski/Points/StandardCartesianChart3D Group

/T=0/minkowski/Points/StandardCartesianChart3D/Positions Group

/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC Group

/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxx Dataset {43, 37, 19}

/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxy Dataset {43, 37, 19}

/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gxz Dataset {43, 37, 19}

/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyy Dataset {43, 37, 19}

/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gyz Dataset {43, 37, 19}

/T=0/minkowski/Points/StandardCartesianChart3D/ADMBASE::METRIC/gzz Dataset {43, 37, 19}

/T=0/minkowski/Points/SphericalChart3D Group
/T=0/minkowski/Points/SphericalChart3D/Positions Dataset {43, 37, 19}
/T=0/minkowski/Points/SphericalChart3D/ADMBASE::METRIC Group
/T=0/minkowski/Points/SphericalChart3D/ADMBASE::METRIC/grr Dataset {43, 37, 19}
/T=0/minkowski/Points/SphericalChart3D/ADMBASE::METRIC/grp Dataset {43, 37, 19}
/T=0/minkowski/Points/SphericalChart3D/ADMBASE::METRIC/grh Dataset {43, 37, 19}
/T=0/minkowski/Points/SphericalChart3D/ADMBASE::METRIC/gpp Dataset {43, 37, 19}
/T=0/minkowski/Points/SphericalChart3D/ADMBASE::METRIC/gph Dataset {43, 37, 19}
/T=0/minkowski/Points/SphericalChart3D/ADMBASE::METRIC/ghh Dataset {43, 37, 19}

/Time/Grid/Top/Rep/Field/Fragment

- Internal memory layout of a field
- Field may be stored
  - as contiguous block
  - as multiple sub-blocks (e.g. from domain decomposition)
- Layout information may be shared among fragments
  - Coordinate location/boundaries of each fragment

Howto...

- Specify relative information
  - Vertices per triangle, triangles per vertex, vertices per edge, edges per vertex, ..
- Specify hierarchical meshes
- Specify interpolation weights between grids
- Define clusters of cells, edges, …
- Trace time-dependent changes
- Specify partially time-independent data

Concept of “Index Depth”

- An integer property of an index space (Skeleton object), similar to dimensionality
  - Dimensionality: intrinsic property
  - Index depth: extrinsic property
- Describes “how far” the index space is from the primary (mandatory) Vertex information of a Grid:
  - 0 → Vertices
  - 1 → Edges, Faces, Tetrahedrons, …
  - 2 → Groups of edges, faces, tetrahedrons
  - 3 → Groups of elements of index depth 2
  - 4 → Groups of 3-elements
  - …

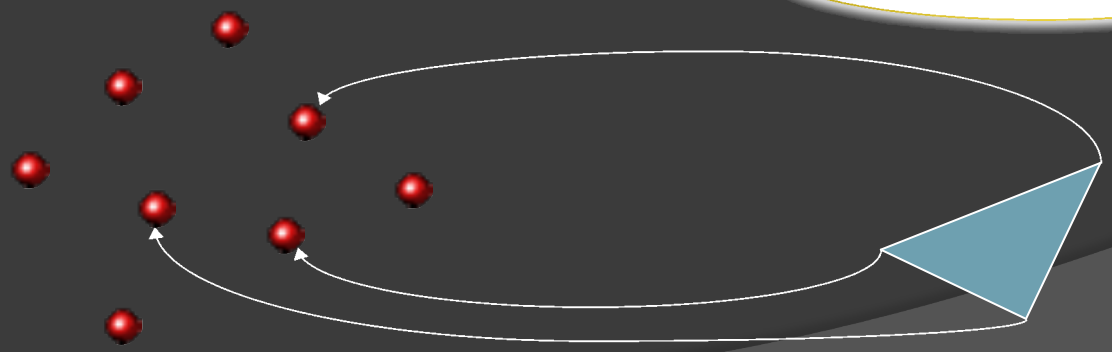
Relative Representations

Triangular Surface: 600331 Triangles, 314442 Points

```
/T=1                                  Group
/T=1/adcirc                           Group
/T=1/adcirc/Points                    Group
/T=1/adcirc/Points/StandardCartesianChart3D Group
/T=1/adcirc/Points/StandardCartesianChart3D/Positions Dataset {314442}
/T=1/adcirc/Connectivity Group
/T=1/adcirc/Connectivity/Points Group
/T=1/adcirc/Connectivity/Points/Positions Dataset {600331}
```

Index Depth = 0

Index Depth = 1

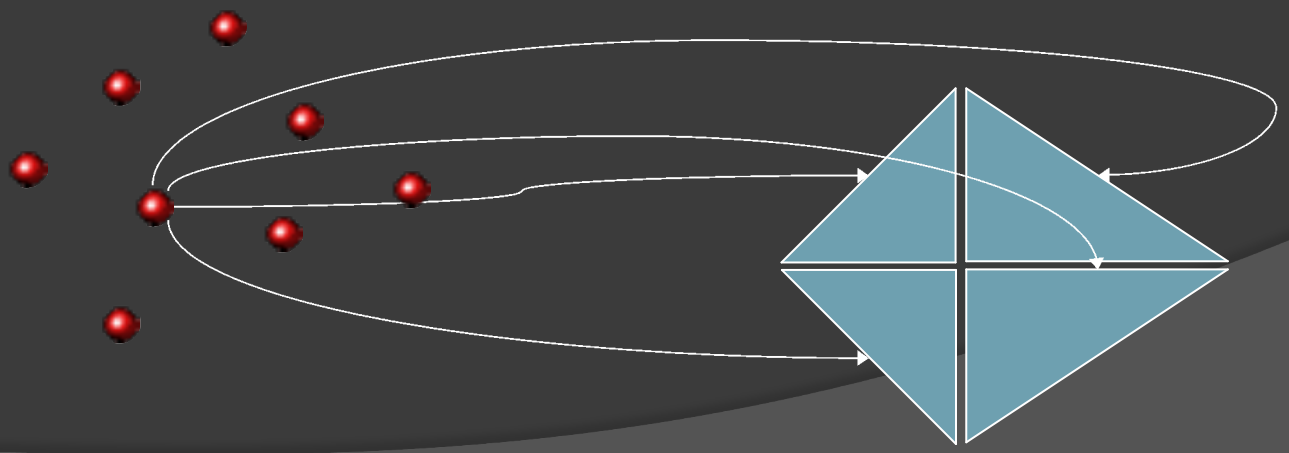


Relative Representations

Inverse Information: Triangles per Vertex

```
/T=1                                Group
/T=1/adcirc                         Group
/T=1/adcirc/Points                  Group
/T=1/adcirc/Points/StandardCartesianChart3D Group
/T=1/adcirc/Points/StandardCartesianChart3D/Positions Dataset {314442}
```
/T=1/adcirc/Points/Connectivity/Positions Dataset {314442}

```
/T=1/adcirc/Connectivity Group
/T=1/adcirc/Connectivity/Points Group
/T=1/adcirc/Connectivity/Points/Positions Dataset {600331}
```



Fields on Relative Representations

Triangular Surface: 600331 Triangles, 414442 Points

plus data on the vertices

```
/T=1                    Group
/T=1/adcirc/Connectivity Group
/T=1/adcirc/Connectivity/Points Group
/T=1/adcirc/Connectivity/Points/Positions Dataset {600331}
/T=1/adcirc/Points      Group
/T=1/adcirc/Points/StandardCartesianChart3D Group
/T=1/adcirc/Points/StandardCartesianChart3D/Positions Dataset {314442}
/T=1/adcirc/Points/StandardCartesianChart3D/elevation Dataset {314442}
/T=1/adcirc/Points/StandardCartesianChart3D/vector Dataset {314442}
```

Relative Representations

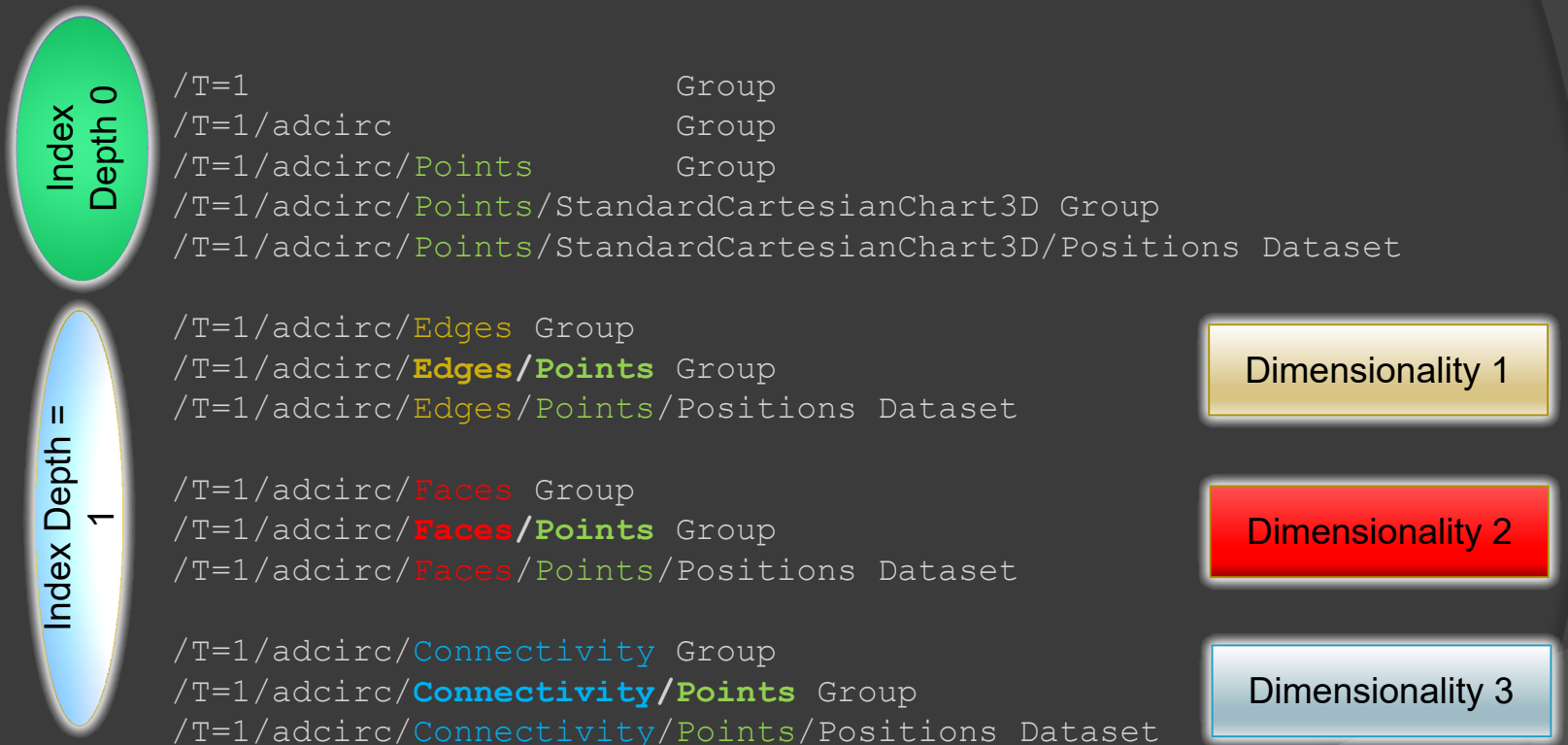
Triangular Surface: 600331 Triangles, 414442 Points

plus data on the triangles

```
/T=1                          Group
/T=1/adcirc/Connectivity      Group
/T=1/adcirc/Connectivity/Points Group
/T=1/adcirc/Connectivity/Points/Positions Dataset {600331}
/T=1/adcirc/Connectivity/StandardCartesianChart3D/elevation Dataset {600331}
/T=1/adcirc/Connectivity/StandardCartesianChart3D/vector Dataset {600331}
/T=1/adcirc/Points            Group
/T=1/adcirc/Points/StandardCartesianChart3D Group
/T=1/adcirc/Points/StandardCartesianChart3D/Positions Dataset {314442}
```

Unstructured Meshes

- Skeletons and Primary Representations:



Unstructured Meshes

Skeletons and Secondary Representations:

```
/T=1                                  Group
/T=1/adcirc                           Group
/T=1/adcirc/Points                    Group
/T=1/adcirc/Points/StandardCartesianChart3D Group
/T=1/adcirc/Points/StandardCartesianChart3D/Positions Dataset
/T=1/adcirc/Points/Faces Group
/T=1/adcirc/Points/Faces/Positions Dataset

/T=1/adcirc/Edges Group
/T=1/adcirc/Edges/Faces Group
/T=1/adcirc/Edges/Faces/Positions Dataset

/T=1/adcirc/Faces Group
/T=1/adcirc/Faces/Edges Group
/T=1/adcirc/Faces/Edges/Positions Dataset

/T=1/adcirc/Connectivity Group
/T=1/adcirc/Connectivity/Edges Group
/T=1/adcirc/Connectivity/Edges/Positions Dataset
```

Faces per Vertex

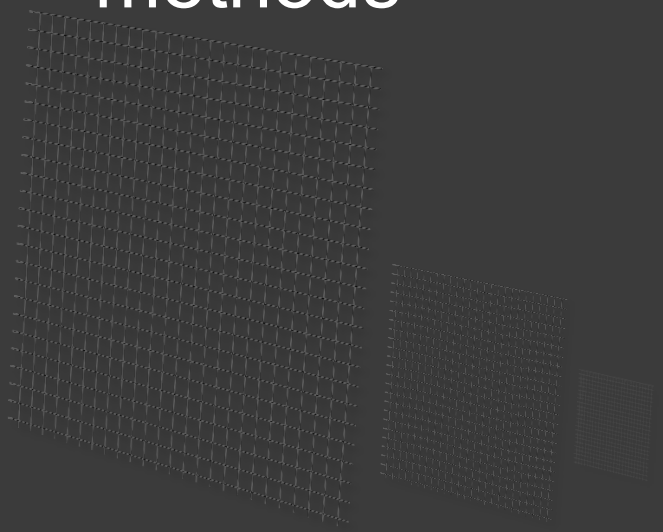
Faces per Edge

Edges per Face

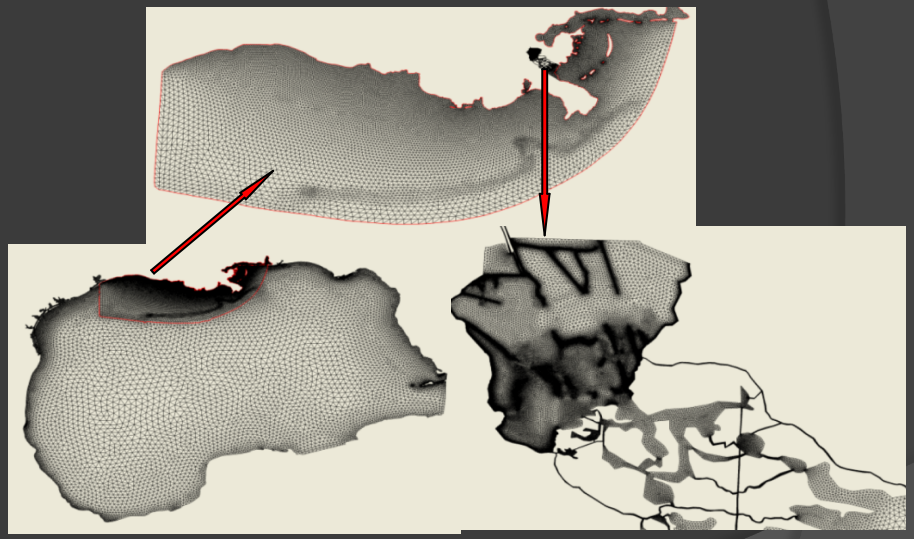
Edges per Cell

Hierarchical Meshes

- One vertex may be covered by multiple refinement levels – multiresolution methods



Block-structured adaptive mesh refinement grid (AMR)



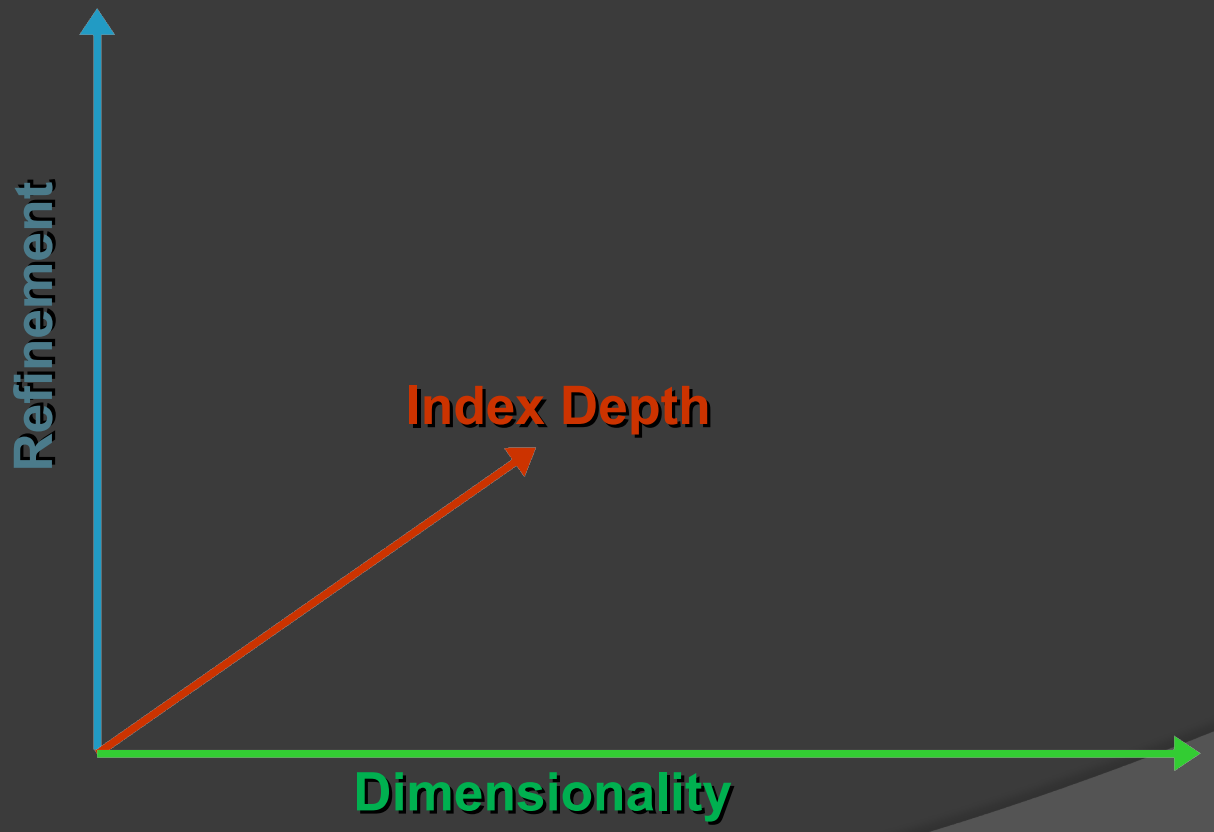
Nested triangular grids for the La-Tex continental shelf (upper panel, Wang and Justic, 2009), the Breton Sound estuary (lower right panel, Huang et al., 2011) and (lower left) the Gulf of Mexico model.

Hierarchical Skeletons

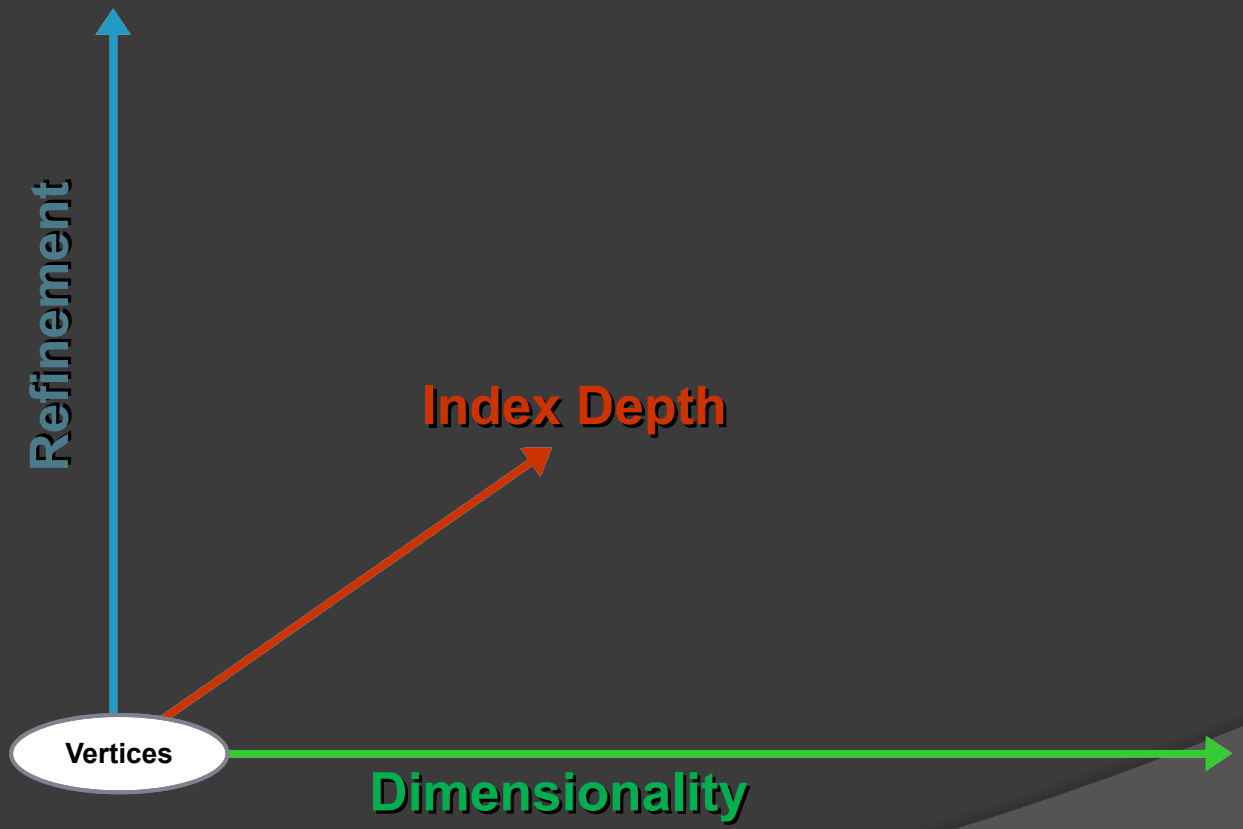
- Third identification parameter on Skeletons:

  "refinement" (integer valued or n-dimensional set of integer valued group attribute)
- Allows to formulate "a grid within a grid"
- Replicates topological structure on different refinement levels

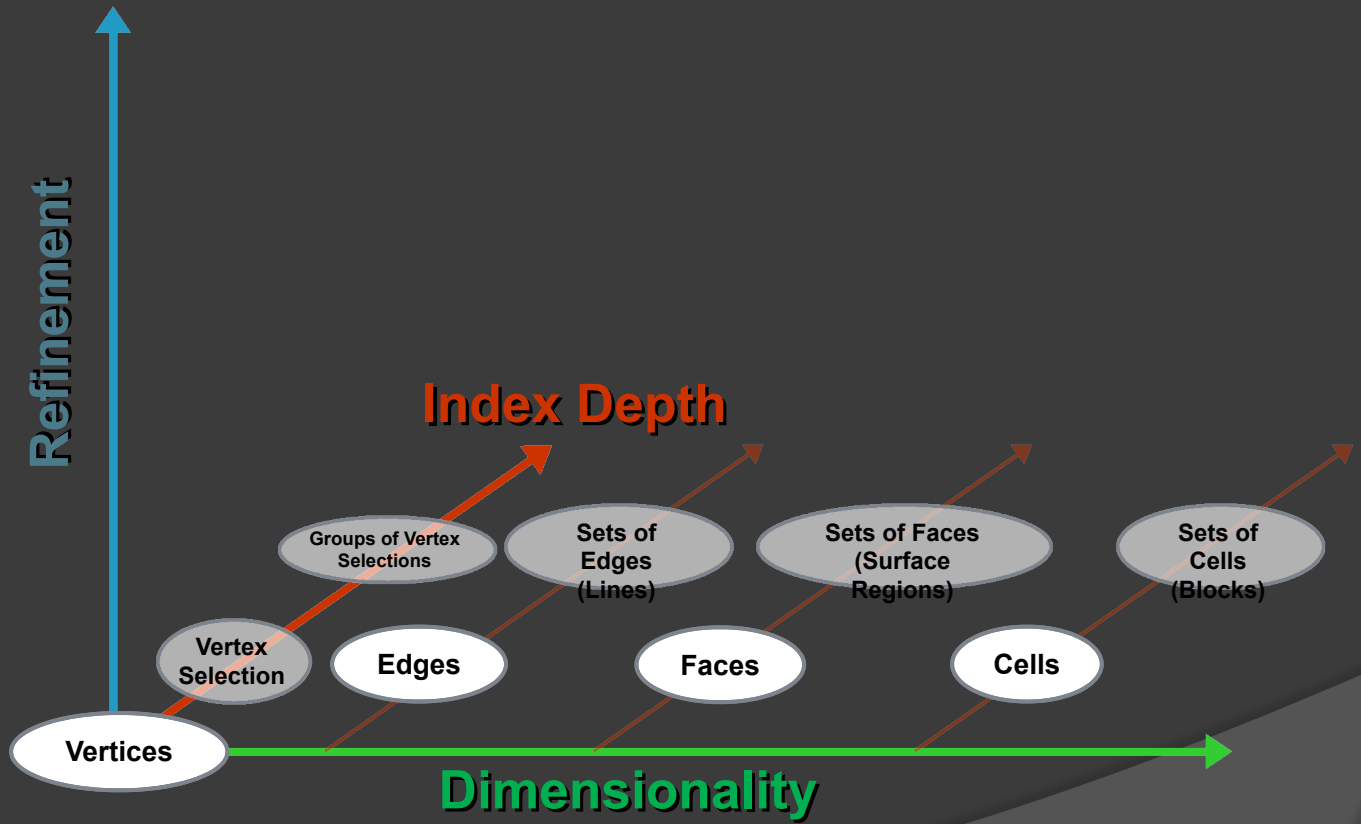
Skeletons as Fibers of a 3-dimensional parameter space



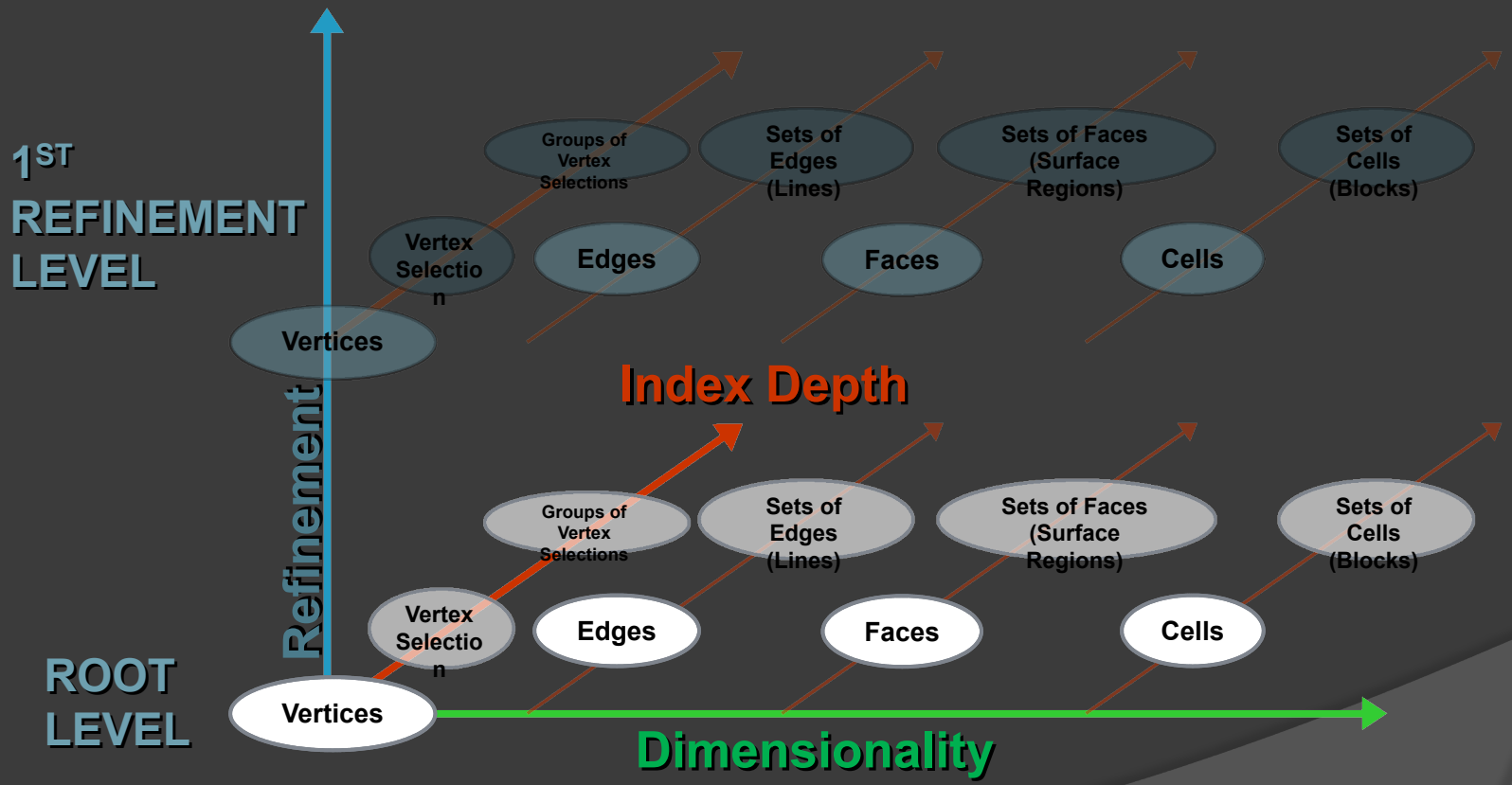
Skeletons as Fibers of a 3-dimensional parameter space



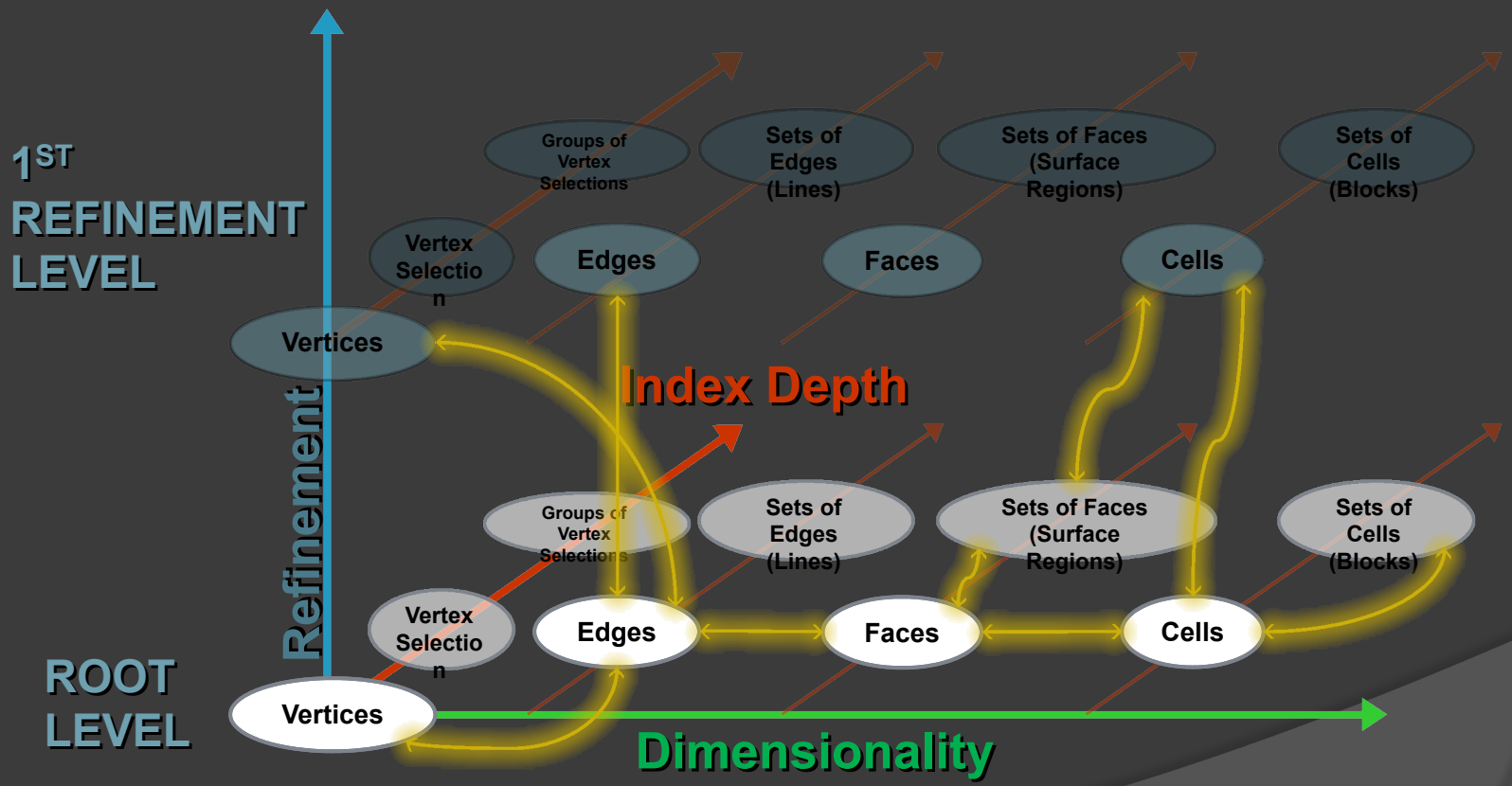
Skeletons as Fibers of a 3-dimensional parameter space



Skeletons as Fibers of a 3-dimensional parameter space



Relative Representations as Connectors between Skeletons

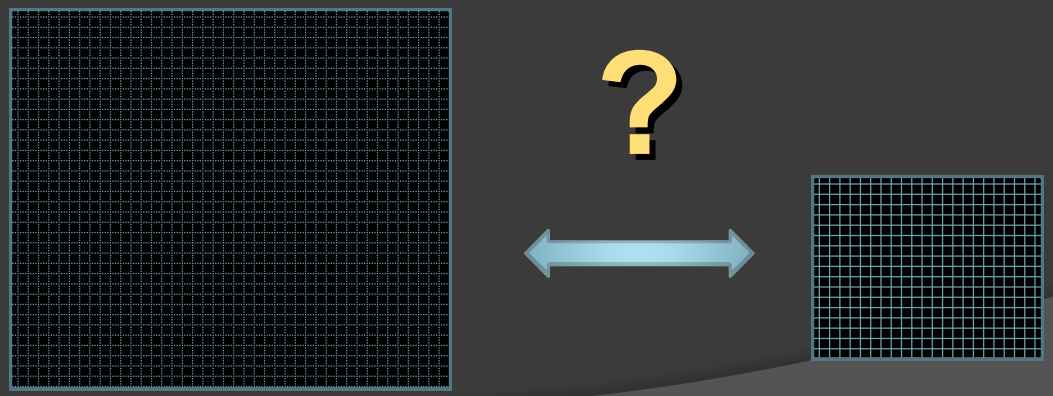


Inter-Grid Relationships

- Representations may refer to Skeletons within another Grid:

/T=0/FirstGrid/Points/CartesianChart3D/Positions {float x,y,z} Dataset {1000}

/T=0/SecondGrid/Points/CartesianChart3D/Positions {float x,y,z} Dataset {100}



Inter-Grid Relationships

- Use *index fields/bitmaps* when points match exactly:

```
/T=0/FirstGrid/Points/CartesianChart3D/Positions      {float x,y,z} Dataset {1000}
/T=0/FirstGrid/Points/SecondPoints/Positions          {bitmap}      Dataset {1000}
                                    Representer → /T=0/SecondGrid/Points

/T=0/SecondGrid/Points/CartesianChart3D/Positions {float x,y,z}  Dataset {100}
/T=0/SecondGrid/Points/FirstPoints/Positions          {int    index} Dataset {100}
                                    Representer → /T=0/FirstGrid/Points
```

- Utilizes HDF5 symbolic links

Inter-Grid Relationships

- Use *interpolation weights and indices* when points do ***not*** match exactly:

```
/T=0/FirstGrid/Points/CartesianChart3D/Positions      {float x,y,z} Dataset {1000}
/T=0/FirstGrid/Points/SecondPoints/Positions          {int [];}     Dataset {1000}
                                  Weights             {float [];}   Dataset {1000}
                                  Representer → /T=0/SecondGrid/Points

/T=0/SecondGrid/Points/CartesianChart3D/Positions {float x,y,z}  Dataset {100}
/T=0/SecondGrid/Points/FirstPoints/Positions          {int   [];}   Dataset {100}
                                  Weights             {float [];}   Dataset {100}
                                  Representer → /T=0/FirstGrid/Points
```

- Allows to evaluate any field on FirstGrid on SecondGrid (and vice versa)

Temporal Relationships

- Representations may refer to Skeletons of the same Grid on different times:

```
/T=0/FirstGrid/Points/CartesianChart3D/Positions      {float x,y,z} Dataset {1000}
/T=0/FirstGrid/Points/SecondPoints/Positions          {int [];}     Dataset {1000}
                                  Weights             {float [];}   Dataset {1000}
                                  Representer → /T=1/FirstGrid/Points

/T=1/FirstGrid/Points/CartesianChart3D/Positions {float x,y,z}  Dataset {100}
/T=1/FirstGrid/Points/FirstPoints/Positions          {int    [];}   Dataset {100}
                                  Weights            {float [];}   Dataset {100}
                                  Representer → /T=1/FirstGrid/Points
```

- Allows to trace the evolution of features
- Timelike derivatives, …

Partial Time Dependence

- Application Scenario:
  - Curvilinear Grid with vector field
  - Coordinates do not vary over time
  - Dynamic Vector field, changes each time step

```
/T=0/MyGrid/Points/CartesianChart3D/Positions      {float x,y,z} Dataset {10,10,10}
/T=0/MyGrid/Points/CartesianChart3D/Vectorfield    {float dx,dy,dz} Dataset {10,10,10}

/T=1/MyGrid/Points/CartesianChart3D/Positions → /T=0/MyGrid/Points/CartesianChart3D/Positions
/T=1/MyGrid/Points/CartesianChart3D/Vectorfield    {float dx,dy,dz} Dataset {10,10,10}
```

- Utilizes HDF5 symbolic links
  - Any application operation on dynamic data can read partially time-dependent data as well
  - Any field can be defined time-independent, for all time steps, or just some time interval

Summary

- The F5 model avoids “keyword” conventions as much as possible
- Semantic information is cast into the *structure* of the file *instead* of relying on naming conventions (enumeration approach, registries, naming standards…)
- Interpretation of data fields is based on *placement* of data sets in the hierarchy
- Internal layout is not 100% keyword-free, but as widely as possible
  - Some keywords will become obsolete with future advancements of HDF5 (e.g., shared dataspaces)

Conclusion

- Commonalities across Grid types are not just enabled, they are unavoidable
- Simple data types have simple representations
- Complex data types are constructible from similar, reusable structures
- The F5 model does not cover *any* type of data, but a very wide range
- Certain aspects of the F5 model leave space for further refinement and definition how to interpret things

Implementation Design & Concepts

F5 API

Design of the F5 Library

- API similar to HDF5
- Expose HDF5, not hiding it
  - Use as much of HDF5 as possible
  - Upgrade to future HDF5 functionality once available
  - Emulate future HDF5 functionality in the meanwhile

Design of the F5 Library

- 3 categories of API functions:
  - Internal API
    - Common functions to share logistics
  - Low-Level API
    - Access to all constructions elements constituting the F5 formulation
  - High-Level API
    - Simple, “one-call” functions covering frequently asked cases

F5Path object

- Central object for user-accessible operations
- This structure contains all HDF5 identifiers that describe a path in an HDF5 file towards a field structure.
- /Time/Grid/Top/Rep/Field/Fragment

  hid_t hid_t hid_t hid_t hid_t hid_t

Example: API levels

- F5Lwrite(hid_t R_id, const char*fieldname, int dimension, const hsize_t*dims, hid_t fieldtype, hid_t memtype, const void * dataPtr, hid_t enum_type, hid_t property_id);
- F5Fwrite(F5Path*fpath, const char*fieldname, int dimension, hsize_t*dims, hid_t fieldtype, hid_t memtype, const void * dataPtr, hid_t property_id)
- F5Fwrite_uniform_cartesian3D(hid_t file_id, double time, const char*gridname, const F5_vec3_point_t*origin, const F5_vec3_float_t*spacing, hsize_t dims[3], const char*fieldname, hid_t fieldtype, const void * dataPtr, const Char*coordinate_system,hid_t property_id);

F5F.h

- User Field Access Functions
  - e.g.:

```
F5Fwrite(F5Path*fpath, const char*fieldname, int dimension, hsize_t*dims,
    hid_t fieldtype, hid_t memtype, const void * dataPtr,  hid_t property_id);
```

  - Uses F5Path object to write HDF5 data to the right location
  - Passes HDF5 information to the right location
  - Performs additional semantic checks beyond HDF5 – dataspace consistent

F5L.h

- Internal low-level field functions
  - E.g.:

F5Lwrite(hid_t R_id, const char*fieldname, int dimension, const hsize_t*dims, hid_t fieldtype, hid_t memtype, const void * dataPtr, hid_t enum_type, hid_t property_id);

  - Merely uses HDF5 identifiers
  - Creates HDF5 dataset with attributes as interpreted by HDF5
  - Caller must know exact location

Groups of Functions

- F5Bchart.h
  - coordinate systems, charts
- F5AMR.h
  - adaptive mesh refinement
- F5uniform.h
  - Uniform meshes
- …

The End

http://www.hdfgroup.org

http://www.fiberbundle.net

http://sciviz.cct.lsu.edu/projects/vish/

http://f5.origo.ethz.ch