ISSN: 2834-7706

Research Article

Journal of Mathematical Techniques and Computational Mathematics

Polyadic Decomposition of Voxel Hypercubes in ISR

Greg Passmore*

PassmoreLab, Austin, Texas, USA

***Corresponding Author**
Greg Passmore, PassmoreLab, Austin, Texas, USA.

Submitted: 2025, Oct 30; **Accepted:** 2025, Dec 01; **Published:** 2025, Dec 08

Citation: Passmore, G. (2025). Polyadic Decomposition of Voxel Hypercubes in ISR. *J Math Techniques Comput Math, 4*(12), 01-08.

Abstract
Volumetric datasets in intelligence, surveillance, and reconnaissance (ISR) typically extend beyond three-dimensional spatial fields and often include additional measurements such as spectral, polarization, or temporal components. A voxel hypercube is treated as a high-rank tensor in which fixed spatial indices are combined with one or more attribute dimensions. This method provides a mathematical structure for multi-dimensional voxel data and permits the direct use of tensor operations, including compression and polyadic decomposition, on the resulting arrays. The approach begins with a spatial rank-3 tensor representation and incorporates additional axes for parameters such as spectral frequency, polarization state, and time. Although all axes can be expressed uniformly within the tensor, time differs conceptually because it indexes repeated realizations of the complete spatial–attribute block rather than adding new attributes within a single voxel. The tensor model supports CP, Tucker, and related decompositions, allowing spatial factors to be separated from spectral and polarization components. The treatment of the temporal axis is examined both as an additional tensor rank and as a sequence of discrete hypercubes, with consequences for analysis and computation. Representing voxel hypercubes in this tensor form provides a consistent and general framework in which time encodes evolution of the volume, while the remaining axes encode the set of measurements associated with each voxel.

Keywords: Voxel Hypercubes, Tensor decomposition, Polyadic modeling, Hyperspectral imaging, Spatio-temporal data, Multilinear algebra, High-dimensional data analysis

1. Introduction

Three-dimensional volumetric pixels (voxels) are fundamental in medical imaging, geospatial modeling, and scientific simulations. A voxel represents a value on a regular grid in 3D space (for example, a CT scan's intensity at a given (x, y, z) location). Unlike a simple surface model that only defines external geometry, a voxel model divides space into a rich grid where each small cube can hold multiple data layers. Each voxel is not just an empty box in space; it can encapsulate various attributes describing the material or signal at that location. For example, a geophysical voxel for use in ground penetrating radar, could contain dielectric factors, clay composition, moisture content, etc., all co-located in the same spatial cell. With advanced sensors and simulations, it is increasingly common to encounter such multiattribute voxel data – e.g., a medical voxel with multiple imaging contrasts, or a remote sensing voxel with spectral and polarization information. These rich datasets present a challenge for storage and analysis, but also an opportunity: since they naturally form multi-dimensional arrays where we can leverage the mathematics of tensors to handle them in a unified way.

Treating a voxel hypercube as an N-dimensional tensor means representing the data in a structured array with one axis per coordinate or attribute type. This approach is in line with modern data formats in scientific computing that store measurements over space, time, and other variables as n-dimensional arrays. By casting the multi-modal voxel data as a tensor, we unlock powerful linear algebraic operations (such as multi-way array factorizations, spectral transforms, and compression techniques) that operate across all dimensions simultaneously. In the following sections, I will formalize this tensor representation of voxel hypercubes. We can begin by defining the baseline 3D spatial tensor and then extend it to incorporate additional dimensions for frequency (spectral bands), polarization (multiple signal channels), and time. We distinguish the conceptual roles of spatial versus attribute indices, and show how a polyadic decomposition (Canonical Polyadic or CP) can separate a voxel hypercube into interpretable factors.

The Canonical Polyadic (CP) decomposition, also known as CANDECOMP/PARAFAC[1], expresses an N-way tensor as a sum of rank-one outer products. For an N-mode tensor V, the CP model of rank R is

$$V(i_1, i_2, \ldots, i_N) \approx \sum_{r=1}^{R} u^{(1)} i_1, r, u^{(2)} i_2, r, \cdots, u^{(N)}_{i_N, r}$$

Each term in the sum is a separable component formed by the outer product of N vectors, one drawn from each mode. The decomposition therefore represents the tensor as the sum of R multilinear rank-1 components. The CP rank is the smallest integer R for which such a representation exists.

When R is small relative to the dimensions of the tensor, the CP model provides a compressed representation that isolates dominant patterns along each mode.

Finally, we devote special attention to the temporal dimension: while time can be mathematically treated as just another tensor axis, it has a unique semantic role as it indexes repeated measurements of the entire 3D volume. We discuss two ways to interpret time in the dataset (as a tensor dimension or as a sequence of volumes) and consider the implications of each approach for data processing and analysis. Through this comprehensive formulation, we aim to provide a clear framework for thinking about multi-dimensional voxel data and to illustrate the benefits of a tensor-based description for complex imaging datasets.

2. Spatial Representation

It should also be noted that voxels are not restricted to Cartesian space, but can reside in alternative coordinate domains.

A voxel index may correspond to a position in Fourier space, wavelet coefficient space, spherical-harmonic space, Laplacian eigenbases, or any other orthonormal transform domain. In these cases, the array retains the same tensor structure, but the interpretation of each axis changes. Instead of sampling the value of a physical field at spatial coordinates (x, y, z), the tensor samples the field's representation in a transformed basis. For example, a tensor defined in Fourier space stores coefficients of the form $\hat{V}$ (k_x, k_y, k_z), where (k_x, k_y, k_z) are frequency indices rather than spatial locations. A wavelet-domain tensor stores multiscale and multi-orientation coefficients, and a spherical-harmonic tensor stores coefficient indexed by (ℓ, m) for angular frequencies.

The essential point is that the tensorial representation is independent of the coordinate system used to express the data. The tensor formalism applies equally to physical space and to transformed domains because each domain provides a complete basis in which the volumetric field can be expanded. When CP, Tucker, or tensor-train decompositions are applied to such tensors, the resulting mode factors describe separable structure in the coefficient domain rather than in physical space. This makes polyadic analysis applicable to both spatial voxel grids and to their representations in frequency, multiscale, or spectral-angular bases.

For our immediate purposes of ISR, this paper will focus on Cartesian space, but always keeping an eye on efficiencies that may result in alternative spaces. An example of this for ISR tasking could be risk-similarity space using risk-feature vectors in a local neighborhood, risk-weighted similarity kernels, or spectral risk-similarity embeddings.

3. Tensor Formulation of Voxel Hypercubes

Let's jump into the Cartesian version and the core issue at hand. A three-dimensional voxel grid is represented as a rank-3 tensor. Let N_x, N_y, and N_z denote the number of voxel samples along the x, y, and z axes. The spatial volume is written as

$$V \in \mathbb{R}^{N_x \times N_y \times N_z}$$

where each entry $V(i, j, k)$ is a scalar measurement associated with the spatial coordinate (i, j, k). This representation corresponds to the standard volumetric form used in medical imaging, geospatial modeling, and simulation work. A CT dataset, for example, stores an intensity value at each (x, y, z) location, and a 3D occupancy grid stores a boolean or probability value at the same type of index. In all cases, $V(i, j, k)$ provides one measured quantity per voxel, and this serves as the foundational structure for higher-rank extensions.

The rank-3 form is conceptually a regular three-dimensional array indexed by spatial coordinates. For a concrete example, V (10,25,30) denotes the value stored at spatial location (i = 10, j = 25, k = 30). The structure generalizes across domains: in a CT scan it may represent attenuation, while in a geophysical volume it may indicate density or porosity. Regardless of application, the essential property is that the tensor encodes a fixed scalar per spatial cell on a regular grid. This three-axis representation is the base upon which additional attribute axes are appended to form a voxel hypercube. We frequently see visual examples which imply there is actually more information here than a single scalar, through the use of transfer functions across color and transparency. Nevertheless, this is the most simple (and common) of forms.

Adding an Attribute Dimension: Spectral Data

In some imaging systems, each voxel does not contain a single value, but instead stores a frequency-dependent measurement. Hyperspectral imaging provides a clear example: every spatial point contains a spectrum across a set of discrete frequency or wavelength bins. Extending this to volumetric data produces a rank-4 tensor. Let N_f denote the number of spectral bins. The tensor becomes

$$V \in \mathbb{R}^{N_x \times N_y \times N_z \times N_f}$$

and the entries satisfy

$$V(i, j, k, f)$$

where (i, j, k) identifies the voxel and f indexes the spectral band. Each voxel therefore contains a spectrum of length N_f, giving a local spectral signature at a fixed spatial location. This generalizes hyperspectral cubes, which in the twodimensional case are arrays of shape (x, y, λ), to the volumetric case (x, y, z, λ).

This formulation captures the fact that each spatial voxel now holds a vector rather than a scalar. If N_f is large, the pervoxel data is high dimensional, and the overall dataset becomes significantly richer. One may treat $V(i, j, k, f)$ as either a stack of N_f individual scalar 3D volumes, each corresponding to one frequency band, or as a collection of $N_x N_y N_z$ spectral vectors indexed by spatial location. Both views are equivalent. The tensor representation makes this explicit and provides a consistent mathematical framework for applying multi-way operations to spectral–spatial volumetric data.

The extension from $\mathbb{R}^{N_x \times N_y \times N_z}$ to $\mathbb{R}^{N_x \times N_y \times N_z \times N_f}$ is the first step in constructing a voxel hypercube. Additional attribute axes, such as polarization or time, may be appended in the same manner. Each new attribute dimension multiplies the number of measurements attached to each spatial coordinate and increases the tensor rank by one. The spectral extension described here therefore illustrates the general strategy for incorporating multi-attribute voxel data into a unified tensor representation.

4. Adding Multiple Attribute Dimensions: Frequency, Polarization, and Time

The tensor framework extends naturally when multiple per-voxel attributes are present. Modern imaging systems frequently record not only spectral information but also polarization state, temporal evolution, or additional modalityspecific measurements. Each attribute dimension introduces a new axis in the tensor, increasing its rank while maintaining the same spatial indexing.

Let N_f denote the number of frequency bins, N_p the number of polarization channels, and N_t the number of time samples. A general multi-attribute voxel hypercube is represented as

$$V \in \mathbb{R}^{N_x \times N_y \times N_z \times N_f \times N_p \times N_t}.$$

An individual entry satisfies

$$V(i, j, k, f, p, t),$$

where (i, j, k) identifies the spatial voxel and (f, p, t) specifies its attribute state. The spatial indices remain fixed for that voxel, and the remaining axes encode the measured signal across the spectral, polarization, and temporal dimensions.

This representation captures a broad set of practical systems. A spectro-polarimetric volume records frequency and polarization simultaneously at each spatial cell. A dynamic hyperspectral volume records a time series of full spectral cubes. More complex sensors may provide frequency, polarization, angular response, and temporal evolution, each represented as an additional axis. Regardless of modality, the tensor representation isolates spatial structure from the multi-modal signal content attached to each voxel.

The per-voxel signal can be extracted by fixing the spatial indices. For a particular voxel (i, j, k), the attribute subtensor is

$$\mathbf{v}_{i,j,k} = V(i, j, k, :, :, :) \in \mathbb{R}^{N_f \times N_p \times N_t}.$$

This object captures the full attribute state at a single spatial position. The spatial structure of the volume is preserved separately through the (i, j, k) indices, which define the location of that sub-tensor in the 3D grid. The tensor model therefore decouples spatial indexing from attribute indexing, while representing all measurements within a unified array.

5. Spatial and Attribute Indices: Functional Separation

Although spatial and attribute axes coexist within the same tensor, they serve distinct roles. The spatial axes (i, j, k) define the sampling of the physical domain. They correspond to fixed positions in Euclidean space and describe the geometry of the volume. The attribute axes (f, p, t) define the measurements recorded at each voxel. These axes span the feature space associated with each location rather than new spatial coordinates.

This separation is essential when applying operations selectively along specific dimensions. A spatial interpolation acts on (i, j, k), while a spectral transform acts on (f), and a temporal filter acts on (t). In the tensor $V(i, j, k, f, p, t)$, each axis can be isolated through indexing or multilinear operations, enabling precise control over how algorithms interact with spatial and attribute information.

Scientific data formats that support multi-dimensional arrays, such as TIFF derivatives, NetCDF, HDF5, and Zarr, commonly store data using explicit dimension ordering to preserve this separation. A typical ordering in spatio-temporal imaging systems is (t, c, z, y, x), where c denotes channels such as frequency or polarization. The tensor framework generalizes this approach by treating each axis explicitly, making the structure and interpretation of the dataset unambiguous.

6. Polyadic Decomposition of Voxel Hypercubes

The tensor formulation allows the application of polyadic decompositions that generalize matrix factorizations to higher-order arrays. A primary example is the Canonical Polyadic (CP) decomposition. For the multi-attribute tensor

$$V(i, j, k, f, p, t),$$

a CP model of rank R is expressed as

$$V(i,j,k,f,p,t) \approx \sum_{r=1}^{R} a_i^{(r)}, b_j^{(r)}, c_k^{(r)}, d_f^{(r)}, e_p^{(r)}, g_t^{(r)}.$$

Each term in the sum is an outer product of six vectors, one along each mode of the tensor. The spatial factors (*a, b, c*) capture structure aligned with the *x*, *y*, and *z* directions. The attribute factors (*d, e, g*) capture spectral, polarization, and temporal patterns. A low-rank CP representation therefore decouples geometric structure from multi-modal signal content and provides a compressed representation of the voxel hypercube.

This decomposition generalizes the familiar low-rank matrix model. It also supports other factorizations, such as Tucker decompositions and tensor-train factorizations. In all cases, the high-rank hypercube is mapped to a set of lowerdimensional components that describe couplings across spatial and attribute axes. This enables dimensionality reduction, noise suppression, and feature extraction using mathematically principled methods.

7. Time as a Repetition Axis

Although time can be included in the tensor as another index, it differs conceptually from other attributes. Frequency and polarization modify the signal state at a fixed instant, whereas time indexes distinct realizations of the entire spatial–attribute cube.

For a fixed time *t*, the volume slice is

$$V_t(i,j,k,f,p) = V(i,j,k,f,p,t).$$

This is a complete hypercube describing the spatial distribution of spectral and polarization measurements at time *t*. Across N_t samples, the dataset becomes a sequence

$$,V_1,;V_2,;\ldots,;V_{N_t},,$$

each of which contains a full spatial–attribute description. The temporal axis therefore represents an evolution across frames, rather than additional channels at a single point in time.

When time is treated as a tensor dimension, operations such as temporal filtering, frame differencing, or joint spatiotemporal decomposition are implemented directly as tensor operations. When treated as a sequence, one may process each hypercube independently and then analyze temporal behavior across frames. Both are valid interpretations, and the tensor structure accommodates either approach.

8. Tensor Decompositions for Voxel Data

Tensor representations of voxel hypercubes allow multilinear decomposition processes that generalize matrix factorization to higher-order arrays. These decompositions separate spatial structure from spectral, polarization, and temporal variation. For a tensor

$$V(i,j,k,f,p,t) \in \mathbb{R}^{N_x \times N_y \times N_z \times N_f \times N_p \times N_t},$$

each index corresponds to one coordinate axis of the hypercube. The decomposition expresses the data as a multilinear combination of lower-dimensional factors.

9. Canonical Polyadic (CP) Model

The CP model represents the voxel tensor as a sum of rank-one terms. The approximation is

$$V(i,j,k,f,p,t) \approx \sum_{r=1}^{R} a_i^{(r)} b_j^{(r)} c_k^{(r)} d_f^{(r)} e_p^{(r)} g_t^{(r)}.$$

Each vector group corresponds to one tensor axis. The vectors $a_i^{(r)}, b_j^{(r)}$, and $c_k^{(r)}$ represent spatial factors, and $d_f^{(r)}, e_p^{(r)}$, and $g_t^{(r)}$ represent spectral, polarization, and temporal factors. A low CP rank indicates redundancy across axes and provides a compact representation.

9.1. Tucker Decomposition

The Tucker model, which comes from psychometrics. He developed the model to generalize classical factor analysis beyond matrices. Psychological testing data often arranged responses across subjects, items, and conditions, forming three-way arrays (a subset of polydiadics). Tucker proposed a multilinear factor model in which each mode of variation receives its own factor matrix and the interactions across modes are encoded in a compact core tensor. This structure allows multiway covariation to be captured in a way not achievable with matrix-based methods. The approach later became a foundation for tensor analysis in chemometrics, signal processing, and computer vision. The form is

$$V(i,j,k,f,p,t) \approx \sum_{\alpha,\beta,\gamma,\delta,\epsilon,\eta} G_{\alpha\beta\gamma\delta\epsilon\eta} A_{i\alpha} B_{j\beta} C_{k\gamma} D_{f\delta} E_{p\epsilon} T_{t\eta}.$$

Here G is a reduced core tensor, and (A, B, C, D, E, T) are factor matrices.

This form supports independent dimensionality reduction along each axis.

9.2. Tensor-Train (TT) Representation

The tensor-train representation expresses the six-dimensional voxel hypercube using a sequence of third-order cores. In TT form,

$$V(i,j,k,f,p,t) = G_1(i), G_2(j), G_3(k), G_4(f), G_5(p), G_6(t),$$

where each G_m is a core tensor with TT-ranks controlling compression.

The TT format scales linearly with the number of dimensions and is well suited to extremely large voxel data.

9.3. Interpretation

These decomposition processes expose the structure of multi-attribute voxel datasets. Spatial indices (i, j, k) capture geometry, while (f, p, t) describe spectral, polarization, or temporal characteristics, as an example. Decomposition separates these contributions into modes that can be analyzed independently or jointly. The resulting factors provide a basis for compression, denoising, feature extraction, and modeling of variation across both spatial and attribute dimensions.

10. Time as a Special Dimension in Voxel Hypercubes

The tensor representation introduces time t in the same formal manner as any other attribute axis. A dataset may therefore be written as $V(i, j, k, f, p, t)$, and common array formats store such data directly as an n-dimensional array.

While this is mathematically consistent, time differs conceptually from attributes such as frequency or polarization. Frequency and polarization enrich the state of a single voxel at a given instant, whereas time indexes distinct realizations of the entire spatial and attribute structure. This section describes this distinction and presents two equivalent interpretations for incorporating time into voxel hypercubes.

10.1. Time as a Repetition of the Full Spatial-Attribute Structure

For a fixed time t_0, the tensor slice

$$V_{t_0}(i,j,k,f,p) = V(i,j,k,f,p,t_0)$$

contains all spatial and attribute measurements at that instant. It is a complete hypercube describing the state of the system at time t_0. When time advances to t_1, the slice $Vt_1(i, j, k, f, p)$ represents another full hypercube with potentially different values due to motion, illumination changes, or physical evolution. The entire dataset can therefore be written conceptually as a sequence

$$V_1, V_2, \ldots, V_{N_t},$$

Where Each $V_t \in \mathbb{R}^{N_x \times N_y \times N_z \times N_f \times N_p}$ is a complete spatial-attribute volume. This view emphasizes that time indexes discrete frames, analogous to individual frames in a movie. Each page contains a full scene, and the temporal ordering of pages corresponds to the emerging evolution of the volume.

Including time as an explicit tensor axis produces the single structure

$$V(i,j,k,f,p,t) \in \mathbb{R}^{N_x \times N_y \times N_z \times N_f \times N_p \times N_t},$$

which is mathematically equivalent to the ordered list of hypercubes. The difference lies in how the dataset is conceptualized: the tensor form treats time as one axis among several, while the list view treats time as an index selecting distinct hypercubes.

10.2. Why Time Differs From Other Attributes

Attributes such as frequency or polarization describe the internal state of a voxel at a single instant. They coexist within the same voxel and expand the per-voxel measurement vector. Time, in contrast, generates multiple instances of this vector across a sequence. For a fixed spatial coordinate (i, j, k), the set

$$V(i,j,k,:,:,t), \quad t = 1, \ldots, N_t,$$

defines a time series of spectral-polarimetric states. This collection represents an evolutionary trajectory of measurements at a single voxel, not an expansion of its instantaneous state. Thus, attribute axes increase the dimensionality of the voxel measurement, whereas the temporal axis increases the number of voxel instances.

Frequency, polarization, and related channels enrich voxel content at one moment; time produces a sequence of such enriched voxel states. The spatial indices (i, j, k) maintain their geometric meaning across all t, while the voxel values evolve along the temporal axis.

10.3. Two Equivalent Interpretations of Temporal Data

As you can probably guess, then, temporal data may be treated in two equivalent ways:

1.Tensor View (6D Tensor)

Time is included directly as a sixth axis, yielding the tensor

$$\boldsymbol{V(i, j, k, f, p, t)}.$$

This approach is favored by formats such as OME-TIFF, NetCDF, HDF5, and Zarr, which store time-lapse data as contiguous multidimensional arrays. Treating time as an axis allows uniform application of tensor operations, including multidimensional FFTs and multilinear decompositions. This view simplifies programmatic access, indexing, and I/O because the entire dataset resides in a single structured array.

2. Product View (Sequence of Hypercubes)

The dataset is represented as a collection of full volumes,

$$V_1, V_2, \ldots, V_{N_t},$$

where each V_t contains all spatial and attribute information at time t. This view aligns with how dynamic datasets are often conceptualized or processed. Algorithms may iterate over t, treating each hypercube independently or as part of a temporal sequence. This interpretation is convenient when the temporal sampling is irregular or when frame-by-frame analysis is preferred.

Both views are mathematically identical. The tensor view emphasizes uniformity of representation, while the product view emphasizes the sequential nature of time. The choice depends on algorithmic and storage considerations rather than any fundamental difference in the underlying data.

11. Algorithmic Implications for High-Rank Voxel Tensors

Representing voxel data as a high-rank tensor has direct implications for storage layout, numerical processing, and algorithmic efficiency. Each tensor axis corresponds to a physical or measurement dimension, and the order of these axes determines how data are accessed in memory. For a tensor

$$V(i,j,k,f,p,t) \in \mathbb{R}^{N_x \times N_y \times N_z \times N_f \times N_p \times N_t},$$

the indexing structure influences cache behavior, I/O patterns, and the cost of applying transforms along specific dimensions. When tensor sizes grow into the billions of entries, careful ordering of indices becomes necessary to avoid excessive data movement.

11.1. Memory Layout and Access Patterns

High-rank tensor operations often require repeated traversal along one or more axes. Spatial operations use contiguous slices, such as

$$V(:,:,k,:,:,:)$$

while spectral or temporal transformations require reading data along the last dimensions, such as

$$V(i,j,k,:,p,t).$$

Performance therefore depends on how the tensor is stored. Row-major formats (common in C/C++) place the last index fastest in memory, while column-major formats (common in Fortran, MATLAB, NumPy) place the first index fastest. For multi-attribute voxel datasets, storing the spatial axes (i, j, k) as contiguous dimensions minimizes overhead in neighborhood operations, while storing spectral or polarization channels as slow dimensions reduces the frequency of cache-misses during spatial filtering.

11.2. Computational Scaling

Let $N = N_x\, N_y\, N_z\, N_f\, N_p\, N_t$ denote the total number of entries in the voxel hypercube. Direct operations on V scale as $O(N)$. Decomposition methods have different costs:

CP decomposition via alternating least squares typically scales as

$O(RN)$

per iteration, where R is the CP rank.
Tucker decomposition with mode-unfolding scales approximately as

$$O\,(N, \max\,(N_x, N_y, N_z, N_f, N_p, N_t)),$$

depending on factor dimensions.
Tensor-train (TT) decomposition scales linearly in the number of modes and is often

$$O(dr^2n),$$

where d is the number of dimensions, r the TT rank, and n a typical mode size.

$$V_t = V(:,:,:,:,:,t),$$

The tensor representation therefore enables efficient processing when low-rank structure exists, but the naive cost grows with the full hypercube size.

11.3. Temporal Evolution

Time plays a special role in numerical algorithms. Temporal evolution corresponds to applying operations independently to each time slice

or applying joint spatio-temporal algorithms that treat time as an additional axis. For example, a 3D FFT followed by a 1D temporal FFT can be applied across the last dimension, yielding a full 4D transform. Alternatively, filtering can be done per time step followed by temporal differencing. The tensor structure makes both approaches expressible within the same indexing scheme. The whole concept of two methods for time is a little odd, but hopefully this clarifies the approaches.

High-rank voxel tensors support a unified representation for spatial, spectral, polarization, and temporal measurements. Their algorithmic implications depend on the choice of axis ordering, the existence of low-rank structure, and the computational model used for decomposition or filtering. By structuring voxel hypercubes as tensors, large datasets can be processed systematically using established multilinear algebra tools.

11.4. Implementation Notes: Time in Practice

In practical code and data-processing pipelines, time is usually handled through iteration because it represents repeated measurement rather than an expanded per-voxel attribute. Even when the full dataset can be expressed formally as a single tensor $V\,(i, j, k, f, p, t)$, it is often impractical to load or operate on all N_t temporal slices at once, especially when N_t is large. A common pattern is to stream or load one volume at a time, process it, and then advance to the next time index. This approach does not change the underlying tensor model; it is an implementation decision driven by memory constraints and I/O behavior.

In contrast, attributes such as color channels or polarization states are typically stored directly as array dimensions. The number of such channels is usually small, and all channels are required simultaneously to describe the voxel state at a given time. For example, a video sequence of color images can be represented as a five-dimensional tensor of shape $(N_t, N_z, N_y, N_x, N_c)$, where $N_z = 1$ for a 2D frame or greater for volumetric data, and $N_c = 3$ for RGB channels. In this representation, an element $V\,(t, z, y, x, c)$ gives the intensity of channel c at the spatial position (x, y, z) in frame t. Values $c = 0,1,2$ typically correspond to red, green, and blue. This matches the standard tensor layout (frames,height,width,channels) used in deep-learning systems.

In practice, temporal processing may loop over t explicitly, applying algorithms frame by frame. Alternatively, an operation may be vectorized across the entire tensor if memory permits. Both approaches conform to the same mathematical model. Time simply appends another index to the tensor. The dataset may be stored as a single array or as a sequence of arrays; the representation is equivalent within the tensor framework. Splitting or batching across time is a matter of computational convenience rather than a change in the formal structure. The tensor $V(i, j, k, f, p, t)$ remains the complete spatio-temporal hypercube.

12. Related Hierarchy Considerations

As we discussed with time, there is a linkage, an order to many types of related variables. A slice of the RF spectrum signal will contain a polarization which is meaningless without the parent signal. This is in contrast to linkages between optical intensity of a Lidar and RF signal amplitude, since no relationship exists. Keep these relations straight needs to be accounted for.

Matrix hierarchy is defined by the number of indices used to label an element. A scalar uses no indices and is written as s. A vector uses one index, written as v_i. A matrix uses two indices, written as M_{ij}. Higher-order objects extend this pattern: a third-order tensor is written as T_{ijk}, and an order-d tensor is written as $X_{i_1 i_2 \dots i_d}$. The hierarchy is therefore encoded directly in the index count. The order of the object is the number of indices required to reference a single entry.

13. Mathematical Notation of Structure

The order determines admissible linear transformations. A matrix M_{ij} supports left and right multiplication. A tensor $X_{i_1 i_2 i_3}$ supports mode-wise multiplication by matrices $A^{(1)}i_1\alpha$, $A^{(2)}i_2\beta$, and $A^{(3)}_{i_3\gamma}$ and , with the product written as

$$Y_{\alpha\beta\gamma} = \sum_{i_1,i_2,i_3} X_{i_1 i_2 i_3} A^{(1)} i_1 \alpha A^{(2)} i_2 \beta A^{(3)}_{i_3\gamma}.$$

The notation preserves the index structure, and the number of free indices on the left-hand side defines the resulting order. This establishes the hierarchy algebraically because contractions remove indices and mode-product operations transform them.

13.1. Memory Representation

Large multidimensional arrays are stored in memory as linear sequences. The hierarchy is retained by a deterministic mapping from a multi-index $(i_1, i_2, \dots, i_d)$ to a single linear offset. In row-major layout, the offset is

$$\text{offset} = ((((i_1 n_2 + i_2)n_3 + i_3)\dots)n_d + i_d),$$

while in column-major layout, the first index changes slowest. This mapping ensures that the d-index structure is preserved even though physical memory is one-dimensional. The hierarchy is therefore not a memory construct, but a logical indexing rule layered on top of a flat byte array.

13.2. Retaining Structure

Large scientific systems store order-d arrays together with metadata describing their dimension sizes $(n_1, n_2, \dots, n_d)$ and their storage layout (row-major or column-major). The combination of dimension metadata and index-mapping rules fully preserves the hierarchy. The implementation never infers order from the raw memory contents; it is always supplied explicitly by the array descriptor.

When the order increases, only the dimension array changes. A three-way tensor is declared with three lengths, a sixway tensor with six lengths. Operations such as mode-k multiplication compute linear offsets using the same mapping, ensuring that the multi-index semantics remain intact across the entire hierarchy. The structure persists for arbitrarily large objects because the logical index tuple is never lost.

14. Conclusion

This work presents a compact overview of voxel hypercubes as tensors that merge spatial indices with several attribute dimensions. A voxel carries a vector of measurements rather than a single value, and the tensor $V(i, j, k, f, p, t)$ encodes this directly. The indices (i, j, k) specify spatial position, while (f, p, t) specify spectral, polarization, and temporal coordinates. This produces a single algebraic object suitable for multilinear analysis and compression.

A distinction arises between temporal and non-temporal attributes. Frequency and polarization expand the instantaneous state of a voxel, whereas time generates successive realizations of the full spatial-attribute field. The tensor form is unchanged by this distinction, but interpretation differs. Temporal evolution can be treated as a sequence of frames or embedded as an additional mode without loss of generality.

The representation supports CP, Tucker, and tensor-train decompositions, which separate spatial and attribute factors, expose structure across modes, and permit reduction and denoising. Scientific data already follow multi-indexed layouts, so the tensor view is consistent with standard data organization.

This unified treatment shows that 3D images, hyperspectral volumes, and time-varying fields are all instances of the same class of multi-indexed arrays. The approach supplies a shared mathematical language for high-dimensional voxel data and enables direct transfer of algorithms across sensing domains [1-10].

References

1. Carroll, J. D., & Chang, J. J. (1970). Analysis of individual differences in multidimensional scaling via an N-way generalization of “Eckart-Young”

decomposition. *Psychometrika, 35*(3), 283-319.

2. Coulson, Annie. (2025). “New Software Tool Visualizes the Inside of 3D Images.” BioTechniques News, January 17, 2025. Accessed November 28, 2025.
3. Harshman, R. A. (1970). Foundations of the PARAFAC procedure: Models and conditions for an “explanatory” multi-modal factor analysis. *UCLA working papers in phonetics, 16*(1), 84.
4. Earthmover. (2025). “What Is Zarr? A Cloud-Native Format for Tensor Data.” Earthmover Blog, May 20, 2025. Accessed November 28, 2025.
5. Guo, F., Zhu, J., Huang, L., Li, F., Zhang, N., Deng, J., ... & Hou, X. (2024). Multi-dimensional fusion of spectral and polarimetric images followed by pseudo-color algorithm integration and mapping in HSI space. *Remote Sensing, 16*(7), 1119.
6. Kolda, T. G., & Bader, B. W. (2009). Tensor decompositions and applications. *SIAM review, 51*(3), 455-500.
7. Pixxel. (2025). “Understanding Hyperspectral Data Cubes.” Pixxel Hyperspectral Academy. Accessed November 28, 2025.
8. Tang, Dave. (2022). “Data Representations – Deep Learning (Tensor Basics).” DeepLearning.DaveTang.org Blog, May 30, 2022. Accessed November 28, 2025.
9. Voxel Space. (2025). “Inside the Cube: What Makes Each Voxel a Data Powerhouse.” Voxelspace Blog, September 10, 2025. Accessed November 28, 2025.
10. Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., ... & Mons, B. (2016). The FAIR Guiding Principles for scientific data management and stewardship. *Scientific data, 3*(1), 1-9.