Tensor Omics: Core Preprocessing, Analysis Pipeline, and Geometric Tools for Semantic Interpretation

Asis Hallab, "el Bobito"

October 1, 2025

1 Overview

This document outlines the sequential pipeline used in Tensor Omics for preprocessing, vector space construction, outlier detection, and geometric analysis of gene expression data.

2 From Raw Data to Expression Vectors

The starting point for all downstream analyses is a table of gene-wise expression measurements across a set of biological samples, such as tissues, developmental stages, or experimental conditions. These measurements may originate from transcriptomic, proteomic, or metabolomic experiments, typically normalized to account for sequencing depth and technical variation.

What is Gene Expression Data? For each gene, a real-valued quantity is recorded that reflects its biological activity in a given sample. In the case of transcriptomics, this is often the abundance of messenger RNA (mRNA) molecules transcribed from that gene, estimated via sequencing and mapped back to the gene's known sequence. The result is a non-negative real number proportional to the gene's transcriptional activity in that sample. After normalization and transformation, these values can be treated as coordinates in a real vector space.

Thus, each gene is represented as a vector $\vec{v} \in \mathbb{R}^n$, where n is the number of biological samples (e.g., tissues or time points), and each component $v_i \geq 0$ corresponds to the expression level in sample i. These vectors encode the distribution of gene activity across conditions and form the geometric basis for all downstream analyses.

Note on Stress Responses. In certain studies, a subset of the samples may correspond to stress conditions. Here, "stress" refers to any external or internal perturbation—such as mechanical damage, heat shock, pathogen infection, or nutrient deprivation—that triggers measurable changes in gene expression. These are often analyzed relative to baseline conditions using log fold-change transformations, such that the resulting vectors reflect differential expression patterns, including up- and downregulation across dimensions.

3 Normalization of Expression Values

Raw gene expression data is normalized in the following steps:

1. Division by gene-wise standard deviation for scale normalization. For estimation of the correct standard deviation in low dimensional data-sets see below (3.1).
2. Quantile normalization across samples to reduce global expression biases.
3. Log-transformation: $x \mapsto \log(1 + x)$ to stabilize variance and suppress outliers.
4. For stress response studies, log fold-changes are computed:

$$\Delta_{\text{stress}} = \log(1 + x_{\text{stress}}) - \log(1 + x_{\text{control}})$$

This effectively makes each stress related axis show the fold change in gene expression under stress, e.g. "drought in root divided by control root".

3.1 Variance (Standard Deviation) Stabilization for Scaling

In Tensor Omics, normalization of expression vectors requires division by gene-wise standard deviation (SD) across samples. However, empirical SD estimates are unstable for small sample sizes. To address this, we implement a dimension-dependent stabilization strategy that improves robustness while maintaining geometric interpretability.

The strategy is defined as follows:

- For small sample sizes $n < 10$:
  - Empirical SD estimates are highly unreliable.
  - Therefore, for each gene, the standard deviation is replaced by a LOESS-fitted value (see 3.1.1).
  - The LOESS curve is fitted to model the relationship between gene-wise mean expression and gene-wise empirical SD across all genes.
- For intermediate sample sizes $10 \leq n < 21$:
  - A weighted combination of empirical SD and LOESS-fitted SD is used.
  - The effective standard deviation is computed as

$$\mathrm{SD}_{\mathrm{effective}} = \lambda \times \mathrm{SD}_{\mathrm{empirical}} + (1 - \lambda) \times \mathrm{SD}_{\mathrm{LOESS}},$$

    where $\lambda = (n - 10)/11$ increases linearly from 0 at $n = 10$ to 1 at $n = 21$.
- For large sample sizes $n \geq 21$:
  - Empirical SD estimates are considered sufficiently stable.
  - Pure empirical SD is used without adjustment.

This stabilization ensures that scaling by SD is consistent and resistant to noise across a wide range of dataset sizes. It preserves the empirical geometric structure of the data while suppressing variance artifacts that would otherwise distort Tensor Omics analyses, particularly in small or medium sample regimes.

3.1.1 LOESS Estimation of Gene-wise Standard Deviation

To obtain stabilized estimates of gene-wise standard deviations for small sample sizes, Tensor Omics applies a LOESS smoothing procedure based on the relationship between gene-wise mean expression and gene-wise empirical SD.

The procedure consists of the following steps:

- Calculate the empirical mean and empirical standard deviation for each gene across all samples.
- Plot each gene as a point in a scatter plot with
  - X-axis: gene-wise mean expression,
  - Y-axis: gene-wise empirical standard deviation.
- Fit a LOESS curve to these points to model the typical dependence of standard deviation on mean expression.
- For each gene, given its mean expression value, predict its standard deviation by evaluating the fitted LOESS curve at the gene's mean.

This method exploits the fact that gene-wise mean expression values are substantially more stable than standard deviation estimates, especially at low sample sizes. While the empirical standard deviation suffers from high variability when few data points are available, the mean value remains a more robust and lower-variance statistic. By basing the standard deviation estimation on the more reliable mean, the procedure suppresses noise-induced artifacts and yields smoothed, biologically plausible variance estimates. This LOESS-based adjustment is critical for preserving the integrity of the normalization process in small-sample regimes, ensuring that subsequent geometric analyses remain meaningful and reproducible.

4 Construction of Expression Vector Space

- Biological replicates belonging to the same axis (tissue, stressed tissue, etc.) are averaged post-normalization.
- Each gene is represented as a vector $\vec{p} \in \mathbb{R}^n$, where n is the number of conditions or tissues. Hence the analysis is embedded in an n-dimensional ambient space.
- Expression vectors are stored in Fortran arrays. *Importantly* Fortran is column major and vectors *must* be columns.
- Two K-D Trees are constructed:
  - A raw-space K-D Tree (unnormalized vectors) for Euclidean queries.
  - A normalized-space K-D Tree (unit-length vectors) for cosine similarity and angular queries.
- The first axis is designated the **Anchor Axis (AA)** for visualization.

4.1 Vector Space Storage in Fortran

All expression vectors in Tensor Omics are stored in a central memory structure optimized for performance and access in Fortran. This architecture ensures efficient storage, lookup, and downstream computation.

- **vec_store:** A two-dimensional Fortran array of size $n \times v$, where n is the number of expression dimensions (e.g., tissues or conditions), and v is the number of vectors (e.g., genes). Each column represents one expression vector.
- **cart_tree:** A K-D Tree constructed on the raw (unnormalized) vectors stored in `vec_store`. This structure enables rapid spatial neighborhood queries using Euclidean distance in the original expression space. This is not initialized until step 10.1.
- **sphere_tree:** A second K-D Tree constructed from unit-length normalized vectors derived from `vec_store`. This structure supports directional queries based on cosine similarity or angular distance. This is not initialized until step 10.1.

Together, these data structures allow both magnitude-sensitive and direction-sensitive operations across expression fields.

5 Gene Family Grouping and Centroid Estimation

- Gene families, populations, or functional groups are defined using metadata.
- Family centroids $\vec{o}$ are computed either as:
  - The mean vector of all family members.
  - The mean vector of a family's orthologs only.
- All metadata and vectors can be serialized into `.txdata` binary files.

6 Space Interpolation and Smoothing

To support field-based analysis and ensure consistent representation across expression space, a regular structured grid is constructed. This grid forms the structural foundation for kernel-based interpolation and vector field smoothing.

6.1 Grid Step Size Estimation

The grid resolution is determined empirically from intra-family expression variation. Specifically, for each gene family, the Euclidean distance between each member's expression vector and the corresponding family centroid is computed. These intra-family distances are aggregated across all families into a single distribution. The grid step size (e_g) is then set as the **first quartile** (25th percentile) of this distribution.

Note: Since this step requires family membership and centroid information, it must be performed after gene family grouping and centroid estimation (see Section 5).

6.2 Gaussian Kernel Interpolation

After grid construction, expression vectors are propagated onto the surrounding grid using a Gaussian kernel function centered at each grid vector's position. This operation produces a smoothly interpolated vector field that approximates local expression structure in the semantic space.

For a given expression vector $\vec{v}$ and a grid point $\vec{g}$, the interpolation weight is computed as:

$$w(\vec{v}, \vec{g}) = \exp\left(-\frac{\|\vec{v} - \vec{g}\|^2}{\sigma^2}\right), \text{ with } \sigma = e_g, \text{ the grid-step-size}$$

Only expression vectors located within a cutoff radius of $R = 3\sigma$ from the grid point contribute to its interpolated value. This follows standard statistical convention: for Gaussian kernels, 99.7% of the total mass lies within three standard deviations. This choice ensures inclusion of nearly all local signal while excluding remote vectors that would introduce unrelated semantic influences.

The kernel width σ is set equal to the estimated grid step size, aligning spatial influence with the empirically defined resolution of expression variability. Each grid point $\vec{g}$ receives the kernel-weighted average of all expression vectors within its support radius:

$$\vec{v}_{\text{interp}}(\vec{g}) = \frac{\sum_{\vec{v} \in \mathcal{V}_g} w(\vec{v}, \vec{g}) \cdot \vec{v}}{\sum_{\vec{v} \in \mathcal{V}_g} w(\vec{v}, \vec{g})}$$

where $\mathcal{V}_g$ is the set of all original expression vectors within range of $\vec{g}$. No interpolated vectors are used in this step; only original gene expression vectors contribute.

6.3 Gaussian Kernel Smoothing

After interpolation, a secondary smoothing step is applied to reinforce field coherence. This step uses the same Gaussian kernel formulation but is conceptually and operationally distinct from interpolation.

In smoothing, both the interpolated grid vectors $\vec{v}_{\text{interp}}$ and the original expression vectors $\vec{v}_{\text{orig}}$ contribute to the smoothed result. However, smoothing is applied only to the interpolated grid points, and the smoothed vectors are *not* recursively used in further smoothing steps. This design ensures mathematical clarity, avoids progressive drift or oversmoothing, and enables efficient parallel computation.

The smoothed vector at grid point $\vec{g}$ is computed as:

$$\vec{v}_{\text{smooth}}(\vec{g}) = \frac{\sum_{\vec{v} \in \mathcal{V}_g \cup \mathcal{G}_g} w(\vec{v}, \vec{g}) \cdot \vec{v}}{\sum_{\vec{v} \in \mathcal{V}_g \cup \mathcal{G}_g} w(\vec{v}, \vec{g})}$$

where:

- $\mathcal{V}_g$ are original expression vectors within $R = 3\sigma$ of $\vec{g}$,
- $\mathcal{G}_g$ are neighboring interpolated grid vectors (e.g., in a Moore neighborhood),
- All weights are computed using the same Gaussian kernel with width σ.

Importantly, smoothed vectors are not used to further smooth neighboring grid points. This avoids recursive blending and ensures that the smoothing operation can be executed independently across grid points, supporting massive parallelization on CPU or GPU backends.

Clarification regarding mathematical consistency: This two-step method preserves both the semantic geometry and the quantitative scale of the original expression space. Interpolation is based solely on raw expression vectors, and smoothing includes original anchors, thereby preventing topological drift. The choice of $R = 3\sigma$, combined with symmetric Gaussian kernels and fixed kernel width, ensures that no bias is introduced into the vector field and that all operations remain invariant under rotation and translation in expression space.

Interpretation: Interpolation constructs a continuous field over grid coordinates. Smoothing reinforces continuity and prepares the field for robust downstream operations such as flow estimation, rotation analysis, or module surface detection.

7 Empirical Distribution Estimation

To enable robust outlier detection and hypothesis-free exploration, Tensor Omics constructs empirical distributions of core metrics across all gene families.

7.1 Relative Distance Index (RDI)

For each gene family, we define the centroid $\vec{o}$ as the mean of all ortholog vectors $\vec{o}_i$ across species. To assess the magnitude of a gene's divergence from this conserved expression center, we compute the Euclidean distance:

$$d = \|\vec{p} - \vec{o}\|$$

where $\vec{p}$ is the expression vector of the gene being evaluated.

To normalize this distance and enable scale-independent comparison across families, we compute a family-specific scaling factor as the maximum distance between any ortholog vector and the ortholog centroid:

$$d_{\text{scale}} = \max_i \|\vec{o}_i - \vec{o}\|$$

This normalization is linear in the number of orthologs and avoids the quadratic cost of pairwise distance computation. The Relative Distance Index (RDI) is then defined as:

$$\text{RDI} = \frac{\|\vec{p} - \vec{o}\|}{d_{\text{scale}}}$$

In cases where ortholog annotations are unavailable or the family contains only a single ortholog, a statistical smoothing strategy is used to estimate an appropriate scaling factor. Specifically, for each gene family, we calculate:

$$\text{FS}_i = \text{median}_j \left(\|\vec{f}_j - \vec{o}_i\| \right) \quad \text{and} \quad \text{FD}_i = \text{sd}_j \left(\|\vec{f}_j - \vec{o}_i\| \right)$$

where $\vec{f}_j$ are the expression vectors of all family members and $\vec{o}_i$ is the family centroid (mean of all $\vec{f}_j$). A smoothing function $f(\cdot)$, e.g. via LOESS, is fit to the set $\{(\text{FS}_i, \text{FD}_i)\}$ across all families. This yields an empirical mapping from median distance to expected spread. For families lacking orthologs, the smoothed value $f(\text{FS})$ is used as the denominator in:

$$\text{RDI}_{\text{smoothed}} = \frac{\|\vec{p} - \vec{o}\|}{f(\text{FS})}$$

This hybrid strategy ensures that RDI remains well-defined, interpretable, and comparable across evolutionary, functional, and statistical use cases in Tensor Omics.

All RDI values are pooled across all families to build an empirical distribution of relative intra-family divergence. This enables percentile-based filtering of significant shifts (e.g., 95th percentile for outlier detection) without requiring parametric assumptions.

Note: While RDI values are used to identify significant divergence, the unnormalized distances d_i are retained for all downstream geometric computations, such as vector field construction and shift vector analysis.

7.2 Relative Angle Index (RAI)

The angular difference between a vector $\vec{p}$ and its centroid $\vec{o}$ is computed using the cosine formula:

$$\text{RAI} = \cos^{-1}\left(\frac{\vec{p}\cdot\vec{o}}{\|\vec{p}\|\cdot\|\vec{o}\|}\right)$$

This identifies divergence in direction. Thresholds (e.g. 95%) are derived from the pooled distribution of angles.

7.3 Relative Axis Signal (RAS)

For each unit-normalized expression vector $\vec{p}^{\text{norm}}$, we consider its component $\vec{p}_i^{\text{norm}}$ along each axis i (e.g., tissue). For each axis, the 5th percentile across all vectors is computed as the inactivity threshold. A gene is considered neofunctionalized in axis i if:

$$\vec{o}_i^{\text{norm}} < T_i^{(5\%)} \quad \text{and} \quad \vec{p}_i^{\text{norm}} > T_i^{(5\%)}$$

where $\vec{o}^{\text{norm}}$ is the centroid.

7.4 Rank-normalized Signal Index (RSI)

To assess the local signal strength of expression or shift vectors across the space, Tensor Omics introduces the *Rank-normalized Signal Index (RSI)*. This index provides a scale-independent estimate of the absolute magnitude of a vector, contextualized within the empirical distribution of magnitudes in the current field.

Given a set of vectors

$$\mathcal{V} = \{\vec{v}_1, \ldots, \vec{v}_N\} \subset \mathbb{R}^n$$

each vector's Euclidean norm is computed:

$$s_i = \|\vec{v}_i\|$$

These norms are assembled into a global magnitude distribution. To reduce the impact of noise and outliers, we extract the empirical 5th and 95th percentiles:

$$Q_5 = \text{Percentile}_{5\%}(\{s_i\}), \quad Q_{95} = \text{Percentile}_{95\%}(\{s_i\})$$

For each vector, the RSI is then computed as a clipped, percentile-normalized score:

$$\text{RSI}_i = \min\left(1,\ \max\left(0,\ \frac{s_i - Q_5}{Q_{95} - Q_5}\right)\right)$$

This formulation maps raw magnitudes into the unit interval $[0, 1]$, where values near 0 indicate weak or noise-like vectors, and values near 1 indicate vectors with strong signal relative to the field. Clipping ensures robustness even in the presence of heavy-tailed or multimodal distributions.

Note: RSI applies uniformly to expression fields and shift fields. In the context of signal-based subfield detection, it provides a direct and empirically grounded method for excluding noise vectors and prioritizing high-signal regions. In future extensions, RSI may also be computed on scalar fields derived from second-order tensors (e.g., Frobenius norm of surface tensors).

7.5 Relative Frobenius Index (RFI)

For any set of d vectors $\{\vec{v}_1, \ldots, \vec{v}_n\}$, e.g. gene families, populations, or genes belonging to the same biological process or molecular function group, the vectors are assembled as columns in a matrix $M \in \mathbb{R}^{n\times m}$. The signal strength of this vector-group (matrix) is computed as:

$$\text{RFI} = \frac{\|M\|_F}{\sqrt{m}} = \frac{\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} M_{ij}^2}}{\sqrt{m}}$$

This provides a normalized estimate of signal per vector and is used to assess subfield strength.

Note: The Frobenius norm is here used as a scalar estimate of the total signal energy contributed by a set of expression vectors. The matrix $M \in \mathbb{R}^{n \times m}$ contains m gene vectors (columns) embedded in a fixed n-dimensional semantic space. The RFI, defined as $\|M\|_F/\sqrt{m}$, reflects the average per-vector magnitude in a root-mean-square sense. This formulation emulates the cumulative effect of a gene set on a test particle exposed to their additive influence, without requiring square or self-adjoint structure. It supports comparison of sets with arbitrary cardinality m, all embedded in a shared space $\mathbb{R}^n$.

All empirical distributions are stored in Fortran arrays and used to assign significance thresholds for downstream classification and visualization.

8 Outlier Detection and Shift Vector Calculation

After constructing the vector space and identifying gene families, Tensor Omics detects divergent expression profiles within each family by comparing individual expression vectors to their respective family centroid.

Let $\vec{o} \in \mathbb{R}^n$ denote the centroid of a given family (typically the mean ortholog expression vector), and let $\vec{p}$ denote the expression vector of a candidate paralog. The *raw shift distance* between $\vec{p}$ and the centroid is defined as:

$$d_{po} = \|\vec{p} - \vec{o}\|$$

To determine whether this shift is significant, we normalize the raw distance using the maximum intra-family distance from the centroid. Let $\vec{f_i}$ denote all other expression vectors in the same family. The *Relative Distance Index (RDI)* is calculated as:

$$\text{RDI}_{po} = \frac{d_{po}}{\max_i \|\vec{f_i} - \vec{o}\|}$$

Vectors with an RDI greater than the 95th percentile of the empirical RDI distribution (pooled across all families) are flagged as significant outliers.

Important: Only the raw shift distance d_{po} is used for downstream geometric interpretation, such as vector field construction, trajectory analysis, and clock plot projection. The normalized RDI is used solely for filtering purposes, to robustly identify biologically meaningful divergence across gene families.

Identified outliers are stored as index references in a dedicated Fortran array. For each significant outlier, a corresponding *shift vector* is calculated:

$$\vec{s}_{po} = \vec{p} - \vec{o}$$

These shift vectors are added to the central vector space store (‘vec_store‘) and indexed in the relevant K-D Trees (see section 4.1).

9 Tissue Versatility Calculation

Cosine of the angle θ between an expression vector and the space diagonal constant ($\vec{d}$):

$$\cos(\theta) = \frac{\vec{p} \cdot \vec{d}}{\|\vec{p}\| \cdot \|\vec{d}\|}$$

where $\vec{d} = (1, 1, \ldots, 1)$ is the space diagonal and a constant.

This measure informs about a gene expression tissue (axis) specificity, or versatility, respectively. The larger θ the more the closer an expression vector's localization to one of the space's axes is. Thus its expression is more specific to the tissue represented by that axis. Conversely, the smaller θ the closer the expression vectors localization to tissue uniform expression along the space diagonal $\vec{d}$ and thus the more versatile the gene's expression in the tissues (axes).

10 Clock Plot Projection and Rotation Angle Computation

- All shift vectors are projected into the **Relative Axes Plane (RAP)**, orthogonal to the space diagonal and intersecting the origin.
- The following quantities are calculated for each outlier:
  - Projected shift vector in RAP.
  - Normalized (unit-length) projection.
  - Clock hand vector from origin to projected point.
  - Rotation angle (standardized as clockwise) between projected centroid and outlier.
- Results are stored in a structured Fortran matrix **rap_store**.
  - For each outlier i, store:
    - `Rows 1 to` n: $\vec{s}_i^{\text{RAP}}$
    - `Rows` $n+1$ `to` $2n$: $\vec{s}^{\text{RAP, norm}}$
    - `Rows` $2n+1$ `to` $3n$: $\vec{s}_i^{\text{clock}}$
    - `Row` $3n+1$: rotation angle in radians

10.1 Efficient Vector Lookup Using K-D Trees

To enable fast and scalable querying of gene expression vectors, Tensor Omics employs K-D Trees—an efficient data structure for multidimensional nearest-neighbor searches. The trees are constructed at the very end of the pipeline. Four trees in total, two for all expression and shift vectors, where both are contained in **vec_store** (see 4), and two further for the RAP vectors in **rap_store** (see 10).

Two distinct trees are constructed:

- **cart_tree:** A K-D Tree built from the raw (unnormalized) expression vectors. It supports neighborhood searches in the original Euclidean space, useful for detecting local density patterns or spatial coherence among genes.
- **sphere_tree:** A second K-D Tree constructed from unit-length normalized vectors. This tree enables fast lookup of vectors with similar direction, allowing angular queries based on cosine similarity.

These lookup structures are only built during the final preprocessing stage, prior to interactive analysis. This deferred construction minimizes computational overhead during earlier pipeline steps and ensures consistency with the final state of the vector space. Together, the two trees provide complementary geometric access to the expression landscape.

11 Geometric Tools for Semantic Interpretation

Tensor Omics provides a suite of geometric tools—referred to as *semantic geometry measures*—that support interpretation, subfield detection, and comparative analysis of expression vector fields. These tools operate directly on the vector space and are designed to align with intuitive biological hypotheses.

11.1 Direction-Based Subfield Collector

To detect expression subfields aligned with a biologically meaningful trend, the user defines a direction-of-interest (DOI) vector $\vec{v}_{\text{doi}} \in \mathbb{R}^n$. This may correspond to, for example, increased healing rate, higher oil content, or a linear combination of expression axes.

For each expression vector $\vec{v}_i$, the cosine similarity with the DOI is computed. The method supports two modes, controlled by a Boolean input argument:

- **Normalization enabled:** both vectors are normalized to unit length, and the cosine is computed as:

$$\cos(\theta_i) = \vec{v}_i^{\text{norm}} \cdot \vec{v}_{\text{doi}}^{\text{norm}}$$

  This mode emphasizes *direction* over magnitude and is suited for analyses where only the semantic trend matters, such as expression coordination in transcription factors or between tissues.

- **Normalization disabled:** the cosine is computed relative to the magnitude of the expression vector:

$$\cos(\theta_i) = \frac{\vec{v}_i \cdot \vec{v}_{\text{doi}}^{\text{norm}}}{\|\vec{v}_i\|}$$

  This mode retains the influence of vector magnitude and is recommended when working in fold-change spaces (e.g., stress responses), where high expression shifts carry biological meaning.

The resulting cosine values $\cos(\theta_i)$ are used to construct an empirical distribution. Vectors with the strongest projection onto the DOI (e.g., top 5% by cosine value) are selected as a directional subfield. These can then be characterized using the Subfield Descriptor (Section 11.3).

11.1.1 When to Normalize (Direction-Only Interpretation)

Normalization is recommended when:

- The biological question relates to the **direction of change** in expression across conditions.
- Expression magnitude is not meaningful (e.g., low-expression transcription factors).
- Comparing semantic shifts across gene families or tissues.

11.1.2 When Not to Normalize (Magnitude Matters)

Omitting normalization is appropriate when:

- The space encodes **magnitude changes**, e.g., fold-change after stress.
- High-magnitude responses are biologically meaningful, such as upregulation in immune response genes.
- The DOI is used to detect strong response contributors in a specific context.

11.1.3 Selection of Directional Subfield

The cosine values $\cos(\theta_i)$ are collected across all vectors and an empirical distribution is formed. The top 5% (or another percentile threshold) are selected as the directional subfield—those vectors with the strongest alignment (largest projection) onto the DOI.

The selected subset is returned as a list of vector indices and can be further analyzed using subfield descriptors (Section 11.3).

11.2 Signal-Based Subfield Detection

In addition to metadata-driven and direction-of-interest-based methods for subfield collection, Tensor Omics supports a purely signal-based approach for identifying coherent expression or shift regions. This method—referred to as **Signal-Based Subfield Detection**—operates solely on vector magnitude and can be applied to both expression and shift fields. It provides a simple yet robust means of discovering compact, high-activity modules without assumptions about directionality or coherence, and serves as an efficient precursor to tensor field analysis.

11.2.1 Vector Collection

The procedure begins by selecting the tensor field to be analyzed:

- the **expression field**, consisting of gene expression vectors $\vec{p} \in \mathbb{R}^n$, or
- the **shift field**, with shift vectors defined as $\vec{s} = \vec{p} - \vec{o}$, where $\vec{o}$ is the gene family's centroid.

The full set of vectors is assembled as:

$$\mathcal{V} = \{\vec{v}_1, \ldots, \vec{v}_N\} \subset \mathbb{R}^n$$

Each $\vec{v}_i$ is selected from one or more of the following:

- Original expression or shift vectors (raw),
- Interpolated vectors at grid points,
- Smoothed vectors obtained from post-interpolation Gaussian smoothing.

For each vector $\vec{v}_i$, the signal strength measure depends on the chosen field:

- In the **expression field**, signal strength is evaluated using the Rank-normalized Signal Index (RSI), as described in Section 7.4. This ensures percentile-based normalization across diverse signal ranges.
- In the **shift field**, signal strength is computed directly via the Euclidean norm $\|\vec{s}_i\|$, which reflects absolute expression change relative to the centroid.

This modular design enables application across multiple resolution levels, from raw data to interpolated or smoothed fields. While the default implementation targets interpolated shift vectors, the framework generalizes to any well-defined vector field, including future fields derived from second-order tensors.

Note: In future extensions, this method may be applied to scalar fields derived from surface or covariance tensors (e.g., Frobenius norm of second-order structures). In such cases, the same region detection logic can be applied to identify coherent high-curvature or high-rotation zones.

11.2.2 Magnitude Distribution and Noise Filtering

To isolate signal-dense regions, the Euclidean norms $\|\vec{v}_i\|$ of all vectors in $\mathcal{V}$ are computed. An empirical noise threshold is then derived by estimating the lower 5th percentile:

$$\tau = \text{Percentile}_{5\%}\left(\{\|\vec{v}_i\|\}_{i=1}^{N}\right)$$

Vectors with norm less than τ are considered sub-threshold and excluded from further region growth.

11.2.3 Region Growing

From each unmarked vector $\vec{v}_j$ with $\|\vec{v}_j\| \geq \tau$, a local subfield is grown radially:

- Initialize the region with the seed $\vec{v}_j$ at radius $r = 0$.
- Increment the radius stepwise, adding unmarked neighboring vectors within radius r, provided they meet the magnitude threshold τ.
- If a full radius expansion includes only sub-threshold vectors, the region terminates.

All spatial distances are computed in the native semantic space $\mathbb{R}^n$ using Euclidean distance. Grid vectors and original data vectors may both serve as candidates in the same collection, depending on analysis scope.

11.2.4 Region Marking and Continuation

All vectors added to a region are marked. The procedure continues iteratively over the remaining unmarked, non-noise vectors until no new regions can be seeded.

11.2.5 Interpretation

This method segments the vector field into coherent *signal-rich subfields* based solely on local vector magnitude. In the expression field, the resulting regions may reflect domains of elevated transcriptional activity. In the shift field, they may reveal adaptation zones, such as tissue-specific neofunctionalization or spatially localized responses to perturbation. Because no directionality or covariance modeling is involved, this method is resilient to noise and sample sparsity.

Note: The threshold τ may be adjusted (e.g., 1%, 10%) to modulate sensitivity. The procedure generalizes naturally to higher-order tensors by applying the same algorithm to scalar-valued summary fields (e.g., Frobenius norms of second-order tensors).

11.3 Subfield Descriptor

Each collected subfield is characterized by aggregating its expression vectors into a matrix $M \in \mathbb{R}^{n \times k}$, where each of the k columns is a gene expression vector embedded in the fixed n-dimensional semantic space. The structure of the subfield is quantified using the following derived quantities:

- **Flow direction:** The first eigenvector of the covariance matrix $C = MM^T$, representing the dominant direction of signal flow across the subfield.
- **Rotation vector:** The antisymmetric (skew-symmetric) component of the covariance matrix is defined as:
$$A = \text{skew}(C) := \frac{1}{2}(C - C^T)$$
This decomposition follows conventions from continuum mechanics and enables separation of directional flow and local rotational tendency. The antisymmetric matrix A encodes the local rotation axis and magnitude of net curl induced by the subfield's geometry.
- **Signal strength:** The Frobenius norm $\|M\|_F$ measures the total signal energy contributed by all expression vectors in the subfield.
- **Directional spread:** The matrix rank of M, estimated via singular value decomposition, indicates the number of independent directions represented in the subfield. A rank of one implies collinearity; higher ranks indicate multi-directional heterogeneity.

To enable comparison across subfields of varying size k, the signal strength can be normalized by the root of the cardinality:

$$\text{RFI} = \frac{\|M\|_F}{\sqrt{k}}$$

This Relative Frobenius Index (RFI) provides a density-independent measure of average vector magnitude per subfield.

Note: All direction vectors $\vec{v} \in \mathbb{R}^n \setminus \{\vec{0}\}$ are assumed to be non-zero, as normalization and angular analysis are undefined for the zero vector. If an unscaled antisymmetric component $A = C - C^T$ is used instead of the factorized form, this corresponds to measuring the total (unweighted) rotation strength. The scaled form $\frac{1}{2}(C - C^T)$ is preferred for decomposition consistency and interpretability.

11.4 Workbench for Interactive Exploration

The Tensor Omics workbench is a lightweight, browser-based environment for exploring the expression space and its geometric structures. It supports:

- 2D and 3D projections (e.g., RAP plane, PCA), with zoom and rotation.
- Efficient filtering via metadata or geometric queries using K-D Trees.
- Visual encoding of vector properties (color, shape, size) by metadata attributes.
- Highlighting of gene families, pathways, or other annotated subsets.

Plots are exportable (e.g., to SVG or PDF), and the current view state can be serialized and restored for reproducibility e.g. by attaching it as request parameters and thus enabling linking to preconfigured interactive plots. This supports sharing, publication, and reproducible workflows.

12 Time Series Analysis

Tensor Omics applies a two-stage process to prepare time series expression trajectories for vector field analysis. This approach addresses the common problem of sparsely sampled or unevenly spaced time points in transcriptomic time courses and ensures that divergence detection remains sensitive and robust across genes and families.

12.1 Stage 1: Per-Gene Trajectory Interpolation

For each gene, we construct an interpolated expression trajectory using Gaussian kernel-based interpolation. Let the original time points be $t_1, t_2, \ldots, t_n$, with corresponding expression vectors $\vec{e}_{t_i} \in \mathbb{R}^d$. These are interpolated to a denser trajectory with $T = \max(10n, 30)$ total time points, equally spaced over the original time interval.

Interpolation is performed independently per gene, without considering other genes. This preserves the native trajectory shape while increasing temporal resolution. For each interpolated time point t, a Gaussian kernel $K(t, t_i)$ is applied over all original points:

$$K(t, t_i) = \exp\left(-\frac{(t - t_i)^2}{2\sigma_i^2}\right)$$

where σ_i is the local kernel width, chosen adaptively based on sampling density (see below).

12.2 Stage 2: Contextual Smoothing Across Gene Families

To reduce noise and incorporate contextual biological similarity, each interpolated trajectory is smoothed using a Gaussian kernel over time, borrowing support from orthologs and paralogs.

Each smoothed time point $\vec{e}'_t$ is computed as a weighted average of temporally adjacent points from:

- The same gene (denoted as G)
- Orthologs within the family (denoted as O)
- Paralogs within the family (denoted as P)

The total contextual weights are distributed as:

$$\frac{6}{9} \text{ for } G, \quad \frac{2}{9} \text{ for } O, \quad \frac{1}{9} \text{ for } P$$

Within each group, weight is evenly divided and modulated by a temporal kernel:

$$w_{g_i} = \frac{6}{9|G|}, \quad w_{o_i} = \frac{2}{9|O|}, \quad w_{p_i} = \frac{1}{9|P|}$$

The Gaussian weight based on temporal distance is:

$$K(t, t_i) = \exp\left(-\frac{(t - t_i)^2}{2\sigma_i^2}\right)$$

The final smoothed vector is:

$$\vec{e}'_t = \frac{\sum_i w_i K(t, t_i) \vec{e}_{t_i}}{\sum_i w_i K(t, t_i)}$$

12.3 Adaptive Kernel Width

The kernel width σ_i adapts to local sampling density:

$$\sigma_i = \begin{cases} t_2 - t_1 & \text{if } i = 1 \\ \frac{1}{2}(t_{i+1} - t_{i-1}) & \text{if } 1 < i < n \\ t_n - t_{n-1} & \text{if } i = n \end{cases}$$

12.4 Shift Vector Construction

After interpolation and smoothing, the trajectory for each gene is now defined as a continuous series of vectors $\vec{p}_t$. For any gene family, the centroid trajectory $\vec{o}_t$ is computed as the mean expression trajectory of all orthologs. The shift vector at each time point is then:

$$\vec{s}_t = \vec{p}_t - \vec{o}_t$$

These shift vectors form the time-dependent shift field used for downstream divergence and angle-based analyses.

12.5 Angular Projection and Clock Plot Modes

For each time point t, we estimate the forward velocity of the gene's expression trajectory as:

$$\vec{v}_t = \vec{p}_{t+1} - \vec{p}_t$$

From this velocity vector, we define a local projection plane Π_t orthogonal to $\vec{v}_t$. Into this plane we project:

- the shift vector $\vec{s}_t = \vec{p}_t - \vec{o}_t$
- the anchor axis (AA), as a fixed reference direction in $\mathbb{R}^d$

The clockwise angle between these projected vectors forms the tissue preference rotation at time t, which is visualized in the time trajectory clock plot.

12.6 Clock Plot Visualization Modes

We define two modes for clock plot visualization based on the angular coherence of local velocity vectors $\vec{v}_t$:

Mode A (Low Angular Variance)

If the angular variance of the velocity vectors $\vec{v}_t$ is below a threshold, we compute the mean velocity vector $\vec{v}_{\text{mean}}$, derive a single global projection plane Π_{mean}, and use this for all projections. This yields an interpretable, consistent visualization with a coherent coordinate system origin and visible angle to the space diagonal.

Mode B (High Angular Variance)

If angular variance is high, each time point is projected into its own local plane Π_t. These planes are stacked in the final clock plot by superimposing their origins. The anchor axis and shift vector are rendered as if co-planar. This introduces spatial distortion but preserves directional interpretation at each time point. A supplementary plot indicates angular variance over time.

12.7 Angular Shift Velocity (ASV)

In Mode B, we additionally compute the angular velocity of shift direction over time. At each time point t, the ASV is:

$$\text{ASV}_t = \angle\left(\vec{s}_{t+1}^{\perp}, \vec{s}_t^{\perp}\right)$$

where $\vec{s}_t^{\perp}$ denotes the projection of the shift vector into Π_t. This metric captures turning behavior in shift trajectories and may reveal biological transitions or inflection points in tissue preference.

12.8 Mjölnir's Handle — Canonical Axis Reconstruction for Eitri Surface Extraction

- **Purpose:**
  - Defines a unified, smoothed tracing axis for surface reconstruction within a semantic subfield.
  - Replaces inconsistent per-axis Brokk cylinders with a continuous, empirically derived semantic rail.
  - Anchors all surface geometry: clock hands, pie slicing, and surface candidates.
- **Step 1 — Extract Brokk Axes:**
  - For expression fields: apply PCA to normalized vectors to get spread axes.
  - For shift fields: compute covariance matrix of shift vectors; first eigenvector is dominant flow.
  - Retain 1–3 eigenvectors if needed; typically use the first (dominant) as $\vec{b}$.
- **Step 2 — Compute Clock Hand Projections:**
  - For each vector $\vec{v}_i$ in the subfield, decompose:

$$\vec{v}_i = \vec{r}_i + \vec{v}_i^{\text{clock}}, \quad \text{where} \quad \vec{r}_i = \langle \vec{v}_i, \vec{b} \rangle \cdot \vec{b}, \quad \vec{v}_i^{\text{clock}} = \vec{v}_i - \vec{r}_i$$

  - The root vectors $\vec{r}_i$ define a rough point cloud aligned with the subfield's dominant axis.
- **Step 3 — Smooth Root Vector Cloud:**
  - Apply Gaussian kernel smoothing over the set of $\vec{r}_i$.
  - This produces a smooth curve through semantic space — the **Mjölnir handle**.
  - Recommended kernel width:

$$\sigma = 1.2 \cdot d_{\text{grid step}}, \quad \text{truncated at } 3\sigma$$

  - Adaptive σ may be used based on local density.
- **Step 4 — Surface Extraction via Eitri 2.0:**
  - Divide Mjölnir's handle into equal-length or quantile-based segments.
  - For each segment:
    - Construct a plane orthogonal to the handle (TRAP).
    - Form a cylindrical shell (tube slice) around the segment.
    - Project vectors into the TRAP and perform angular binning.
    - In each pie slice, select the vector with maximal radial length as a surface candidate.
  - Optional smoothing across overlapping segments yields continuous surfaces.
- **Interpretation:**
  - Mjölnir's handle is the *semantic spine* of the surface — the first structure carved before the geometry is wrapped.
  - Enables reversible and modular interpretation of subfield geometry.
  - Removes ambiguity from axis choice and enforces directional consistency across surface detection.

12.9 Instantaneous Acceleration (Future Extension)

As a potential extension, we define instantaneous acceleration as:

$$\vec{a}_t = (\vec{p}_{t+1} - \vec{p}_t) - (\vec{p}_t - \vec{p}_{t-1}) = \vec{p}_{t+1} - 2\vec{p}_t + \vec{p}_{t-1}$$

This second-order derivative could be used to detect abrupt transitions in expression dynamics, e.g., rapid shifts in cell state or bifurcation points in developmental or stress trajectories. It is currently not used in visualization but retained for future interpretability analyses.

12.10 Biological Interpretation

This time-aware geometric framework allows Tensor Omics to identify and interpret dynamic expression shifts. Local angular trajectories provide insight into progressive tissue reprogramming, while metrics such as ASV capture the smoothness or abruptness of functional divergence over time. The projection modes ensure interpretability across both smooth and complex expression paths, enabling application across diverse systems from developmental timelines to stress response series.

12.11 Implementation Notes

All trajectory interpolation, smoothing, projection, and angle computations are performed in Fortran using preallocated arrays for performance and reproducibility. Projection operations avoid

13 Gilgamesh Learning: Semantic Field-Aware Pattern Classification

13.1 Overview

The **Gilgamesh Learning Layer** is the first structured machine learning stage in Tensor Omics. It performs explainable, geometric pattern classification over semantically aligned vector fields, such as expression vectors or shift vectors, using robust aggregation of canonical field descriptors across a normalized subfield space. The method is fully deterministic, aligned with F42-compliant memory handling, and compatible with both supervised and unsupervised classification (e.g., SVMs or interpretable logistic regression).

13.2 Motivation and Biological Context

Traditional ML pipelines in omics use raw gene-level input or PCA-transformed data. These approaches:

- Do not respect semantic coherence of expression vector fields.
- Obfuscate spatial locality or directional geometry.
- Often ignore explainability in favor of brute predictive power.

In contrast, **Gilgamesh Learning** leverages the latent geometry of semantic fields constructed in TOX to provide biologically interpretable classification, tracing the activation of regions (cylinder bins) and their descriptors, and linking learned patterns back to cellular processes or clinical labels.

13.3 Method

Let $\mathcal{F} \subset \mathbb{R}^n$ be a subfield of expression or shift vectors, constructed post-normalization, smoothing, and grid interpolation. Let H denote the **Mjölnir handle**, i.e., the primary flow axis of the subfield, computed via kernel-smoothing over Brokk root vectors as described in the corresponding section.

13.3.1 Cylindrical Binning Along Mjölnir's Handle

1. Compute the set of **Brokk root vectors** by projecting each expression or shift vector onto H and recording the foot of the perpendicular dropped from each vector onto the handle. These form a dense sampling of trajectory-aligned coordinates.
2. Partition the handle H into g equal-sized bins based on the distribution of Brokk root vector coordinates, typically using empirical quantiles (e.g., five-percentile bins).
3. Around each segment of H, construct a cylindrical region $\mathcal{C}_i$ that captures spatially co-local vectors.
4. Assign each vector in $\mathcal{F}$ to its corresponding $\mathcal{C}_i$ using proximity to the associated Brokk root coordinate.

Directional Alignment Convention: To ensure consistent orientation, define "upward" along the handle as the direction in which the cosine between the global principal flow vector and the linearized Mjölnir handle is positive. If negative, reverse the order of bins. This ensures that all bin sequences respect a canonical semantic up–down structure across experiments and fields.

13.3.2 Canonical Descriptors

For each bin $\mathcal{C}_i$, construct a matrix M_i from collected vectors (as columns). Extract the following descriptors:

- **Signal Strength** $s_i = \|M_i\|_F$ (Frobenius norm)
- **Directional Spread** $r_i = \text{rank}(M_i)$
- **Flow Vector** $\vec{f}_i$: First eigenvector of $M_i M_i^\top$
- **Rotational Moment** $\vec{\rho}_i$: Skew-symmetric component of $M_i M_i^\top$

Construct a **concatenated feature vector**:

$$\vec{x} = \text{concat}(s_1, r_1, \vec{f}_1, \vec{\rho}_1, \ldots, s_g, r_g, \vec{f}_g, \vec{\rho}_g)$$

This $\vec{x}$ represents the full subfield $\mathcal{F}$ in structured geometric terms.

13.3.3 Training and Classification

Train a supervised classifier (e.g., linear SVM) using vectors $\vec{x}$ from labeled subfields. For a trained SVM with weights $\vec{w}$ and bias b, the decision function is:

$$\text{score}(\vec{x}) = \vec{w}^\top \vec{x} + b$$

13.4 Explainability

The contribution of each bin-descriptor pair can be extracted from $\vec{w}$:

- Rank the components $x_i \cdot w_i$ in descending order.
- Use cumulative sum until 80% of score($\vec{x}$) is explained.
- Associate top-contributing bins back to their positions in H.

This yields:

- Spatially localized regions of activation.
- Descriptor-level explanations (e.g., flow, rotation).
- Optional lexical/ontological annotations via linked metadata (cf. GAWD).

13.5 Semantic Alignment and Orientation of Subfields

To enable comparability and structured learning across subfields, a consistent orientation convention is required. This is achieved via a canonical alignment using the subfield's main flow axis and the global semantic spine.

Let H denote the Mjölnir handle, constructed as a smoothed curve through Brokk root vectors (the perpendicular foot points of vectors projected onto the main axis). Let $\vec{v}_{\text{flow}}$ be the first eigenvector of the covariance matrix $C = MM^\top$ for all vectors in the subfield.

We define:

$$\text{orientation sign} = \text{sign}\left(\vec{v}_{\text{flow}} \cdot \vec{v}_H\right)$$

where $\vec{v}_H$ is the unit vector pointing from the start to the end of the linearized handle H.

- If orientation sign > 0, the handle is already aligned "upward" (biologically or semantically increasing).
- If orientation sign < 0, the bin sequence is reversed to restore canonical alignment.

This alignment step ensures that all subfields are sliced and encoded in a directionally consistent fashion, crucial for both classification and downstream glyph interpretation. It also preserves directional semantics across layers (e.g., Gilgamesh $\rightarrow$ Nubecita) and across different samples or biological conditions.

13.6 Glyph Description via Lexical Metadata

After training, apply the SVM model to each of the original training subfields. Compute the decision score for each. Extract the subfields falling into the top 5th percentile of decision scores. For each such activated subfield $\mathcal{F}_i$, compute its mean lexical vector $\vec{\ell}_i$ from associated metadata (e.g., Gene Ontology, KEGG, Mapman, patient notes).

The resulting set $\{\vec{\ell}_1, \ldots, \vec{\ell}_k\}$ forms the ontological glyph description of the learned class. These mean vectors can be further clustered, projected, or plotted to study the coherence or diversity of the learned concept. By default, all such $\vec{\ell}_i$ are retained.

13.7 Biological Applications

- Identify class-specific tissue shifts (e.g., cancer vs. control).
- Trace developmental trajectories (e.g., stage-specific expression flow).
- Localize divergent processes across species in comparative studies.
- Link clinical outcomes to geometric shifts in expression fields.
- Extract data-native glyphs summarizing learned biological programs.

13.8 Advantages Over Conventional ML

- Fully explainable: input dimensions are geometric and interpretable.
- Robust to noise: uses smoothed field descriptors, not raw vectors.
- Compatible with GAWD and Eitri for recursive analysis.
- Aligned with TOX principles: no imposed parametric model, minimal transformation.
- Empowers symbolic downstream use: enables second-layer learning (Nubecita).

13.9 Example Pseudocode (Sketch)

```
function gilgamesh_vector(subfield, mjolnir_handle, n_bins):
    bins = split_handle_into_bins(mjolnir_handle, n_bins)
    result = []
    for bin in bins:
        vectors = collect_vectors_in_cylinder(subfield, bin)
        M = form_matrix_from_vectors(vectors)
        s = frobenius_norm(M)
        r = matrix_rank(M)
        f = first_eigenvector(M * transpose(M))
        rho = rotation_vector(skew(M * transpose(M)))
        result.append([s, r, f, rho])
    return concatenate(result)
```

14 Density-Based Bobito Variant: Spherical and Conical Accretion

14.1 Summary

- Use spherical kernels at each grid point to measure **vector density** (count of vectors inside radius r).
- Label each point with this count.
- Sort descending.
- Start growth from the densest point.
- Iteratively grow the sphere, aggregating nearby high-density spheres until **running density derivative drops** (signal plateau).
- In shift fields, alternatively use **cones** to study whether directional flows converge toward dense regions. This addresses directional accumulation and potential biological attractors.

14.2 Interpretation by Vector Type

Expression vector fields (use spheres):

- Best suited for detecting **isotropic clustering** in high-dimensional semantic expression space.
- Reveals:
  - **Conserved regulatory hubs**
  - **Functionally coherent modules**
  - **Developmental plateaus** or **tissue convergence attractors**
  - **Cell-type specific modules** conserved across samples or species

Shift vector fields (use cones):

- Best suited for tracing **directional convergence** of functional shifts.
- Reveals:
  - **Strong directional pull** from paralog groups or neofunctionalizing subsets
  - **Convergent evolutionary innovations**
  - **Stable transcriptomic transitions** across conditions or species
- Use of cones can identify whether directional flow leads into dense attractor zones.

14.3 Verdict

Highly promising. Integrate as a "Bobito-density mode" (working name: **Kolob**), and pin as a method for **density-informed seed detection** in both expression and shift fields.

14.4 Note

While spheres are the natural geometry for isotropic density detection, cones are preferable in *shift vector fields* where direction matters. Cones reveal whether vector flows converge into dense subregions, offering geometric insight into evolutionary or functional attractor zones.

15 Detection of Subfunctionalization and Dosage Effects via Dynamic Programming

This section formalizes the algorithmic strategies to identify subfunctionalization and dosage effect patterns among in-paralogous gene sets, based on semantic expression vectors in Tensor Omics. The procedures leverage dynamic programming and bitmask representations to achieve efficient, branch-pruned exploration of possible paralog subsets.

15.1 Overview

Given:

- The ancestral ortholog expression vector $\vec{o}$.
- A set of n in-paralog expression vectors $\vec{p}_1, \vec{p}_2, \ldots, \vec{p}_n$, stored as columns of a Fortran 2D array.

The goal is to identify subsets of paralogs whose summed expression vectors approximate $\vec{o}$, within a threshold defined as the lower five percentile of family centroid distances (RDI threshold). Two detection scenarios are supported:

1. **Subfunctionalization:** The paralogs exhibit significant angles to the ortholog, partitioning ancestral expression.
2. **Dosage effect:** The paralogs have small angles to the ortholog and sum to a magnitude significantly exceeding $\|\vec{o}\|$.

15.2 Data Structures

- All subset membership is encoded as bitmasks of the ranks of the paralog indices, stored as integer values.
- A dynamic programming storage array M holds the active subsets to be extended at each iteration, also represented as bitmasks.
- An array of norms for each paralog, sorted ascending, is maintained for pruning decisions.
- A logical vector marks which paralogs remain candidates at each step.

15.3 Algorithm Steps

Initialization

- For all single paralogs, measure their angle to $\vec{o}$.
- Apply angle threshold filters:

  - Subfunctionalization: only paralogs with angles greater than the gene-family median angle are marked active.
  - Dosage effect: only paralogs with angles below the gene-family median or lower five percentile are marked active.
- Generate all pairs (k=2) of active paralogs to begin.

Dynamic Programming Iteration For each iteration k:

1. Iterate over the bitmask-encoded subsets of size $k-1$ stored in M.
2. For each such subset:
   (a) Extend the subset by adding one more paralog, using bitwise operations to set zero bits to one.
   (b) Compute the sum of the expression vectors of the extended subset.
   (c) Compute the residual:
   $$\vec{r} = \vec{o} - \sum_{i \in S} \vec{p}_i$$
   (d) Test whether:
   $$\|\vec{r}\| < \text{RDI threshold}$$
   If true, store this bitmask in the results array and *do not* pass it to the next iteration.
   (e) Otherwise, apply pruning logic:
   (f) For *subfunctionalization*:
   - Check the minimal length of any still-active paralog:
   $$\min_i \|\vec{p}_i\| < \|\vec{r}\|$$
   If this fails, prune the subset as further extension is impossible.
   (g) For *dosage effect*:
   - For dosage effect, measure the angle of the current subset sum to $\vec{o}$ and prune if this exceeds the acceptable bound.
   (h) If the subset is marked to continue, store its bitmask in M for the next iteration.
3. Increment k and repeat until k exceeds the maximum paralog search size or no active subsets remain.

Final Filtering At the end of the dynamic program, all positive hits are stored as bitmask-encoded subsets. These represent:

- Subfunctionalized paralog subsets whose summed expression reconstructs $\vec{o}$ within the RDI threshold.
- Dosage-effect subsets whose sum is aligned with $\vec{o}$ but exceeds its magnitude.

15.4 Notes on Implementation

- Subset bitmasks can be efficiently enumerated by setting zero bits in rank order to one.
- Final validation should always be performed using the RDI threshold, regardless of pruning.
- A global maximum paralog subset size (e.g., 15) is recommended to control complexity.
- For large families, consider pre-filtering based on tissue specificity, expression variance, or percentile angles.

15.5 Pseudocode

```
Initialize:
  results = []
  active_sets = all bitmask pairs from angle-filtered paralogs

for k = 2 to MAX_PARALOGS:
  next_active_sets = []
  for each bitmask in active_sets:
    for each still-active paralog not in bitmask:
      candidate_mask = bitmask with paralog bit set
      sum_vector = sum of paralogs in candidate_mask
      residual = o - sum_vector
      if norm(residual) < threshold:
        store candidate_mask in results
      else:
        if dosage:
          if angle(sum_vector, o) too large:
            continue
        if min(paralog norms) >= norm(residual):
            continue
        next_active_sets.append(candidate_mask)
  active_sets = next_active_sets
```

This algorithm achieves robust and scalable detection of subfunctionalization and dosage patterns by systematic, bitmask-driven dynamic programming with domain-motivated pruning rules.

16 Detection of Explanatory Contributions via "Explanatory Contribution Analysis"

The nickname we give this "Factor–Response Trajectory Analysis" or "Trajectory Explanatory Contribution Analysis" is Urd.

This section formalizes the Urd methodology, a Tensor Geometrics module to quantify the explanatory contribution of candidate factors (axes or linear combinations of axes) to one or multiple dependent variables. The method operates on discrete trajectories (time points or ordered conditions) and provides two complementary perspectives: (i) overall explanatory contribution integrated across the entire trajectory, and (ii) local spike contributions at individual time points.

Note that in all modules and - hopefully papers to be published use `TECA` as an abbreviation for Explanatory Contribution Analysis.

16.1 Goal of the Method

The goal of Urd is to detect and quantify whether a factor (or a defined linear combination of factors) provides significant explanatory power for a dependent variable or a defined linear combination of dependent variables. The method supports:

1. Assessing explanatory power over the **whole trajectory**, i.e. whether a factor consistently aligns with the dependent variable across time.
2. Detecting **local spikes**, i.e. individual time points where a factor exerts unusually strong explanatory influence.
3. Applying an **empirical significance criterion** using percentile thresholds derived from the data (e.g., top five percent).

16.2 Overall Trajectory Contribution

Given a factor trajectory $x(t)$ and a dependent trajectory $y(t)$ with $t = 1, \ldots, T$ discrete time points:

1. **Magnitude-aware contribution (dosage-sensitive):**

$$C = \frac{\sum_{t=1}^{T} x(t)\, y(t)}{\sqrt{\sum_{t=1}^{T} x(t)^2} \cdot \sqrt{\sum_{t=1}^{T} y(t)^2}}.$$

   This metric measures the cosine similarity of x and y, capturing both pattern alignment and magnitude.

2. **Direction-only contribution (RAP-normalized):** Normalize each trajectory to unit length:

$$\tilde{x}(t) = \frac{x(t)}{\sqrt{\sum_{t=1}^{T} x(t)^2}}, \quad \tilde{y}(t) = \frac{y(t)}{\sqrt{\sum_{t=1}^{T} y(t)^2}}.$$

   The contribution is then computed as

$$C_{\mathrm{RAP}} = \sum_{t=1}^{T} \tilde{x}(t)\, \tilde{y}(t).$$

   This metric captures purely angular alignment, independent of absolute amplitude.

16.3 Local Spike Contributions

At each discrete time point t, Urd defines a spike score:

1. **Magnitude-aware spike:**

$$s(t) = x(t)\, y(t).$$

2. **Direction-only spike (RAP-normalized):**

$$\tilde{s}(t) = \frac{x(t)\, y(t)}{\sqrt{\sum_{\tau=1}^{T} x(\tau)^2} \sqrt{\sum_{\tau=1}^{T} y(\tau)^2}}.$$

Spike distributions can be pooled across time points, individuals, or factors, depending on the analysis design.

16.4 Empirical Significance Criterion

Urd employs empirical thresholding to distinguish significant contributions:

1. Construct the empirical distribution of contributions C (for whole-trajectory analysis) or spikes $s(t)$ (for local analysis).

2. Define the **top five percentile** of the distribution as the threshold. Contributions or spikes exceeding this threshold are deemed significant.

3. Alternative thresholds (e.g., top ten percentile, permutation-based nulls) can be applied depending on dataset size and variance structure.

16.5 Extensions and Options

1. **Linear combinations of factors:** Urd supports multivariate predictors by forming a factor matrix $X(t)$ with columns $x_i(t)$. Contributions are then computed by projecting $y(t)$ into the subspace spanned by $X(t)$ using least-squares regression.

2. **Multiple dependent variables:** If $y(t)$ is multidimensional, contributions are defined via Frobenius norm or by projection onto linear combinations $c^\top y(t)$, where c is user-defined or derived empirically (e.g., by principal component analysis).

3. **Stability improvements:** With few time points, stability of percentile thresholds can be improved by pooling across individuals, applying short-window smoothing (e.g., three-point rolling averages), or using permutation-based null distributions.

16.6 Summary

Urd provides an efficient and explainable framework for:

1. Quantifying how strongly a factor explains a dependent variable across an entire trajectory.

2. Detecting when in time a factor exerts unusually strong influence.

3. Applying empirical, percentile-based thresholds to control for noise and experimental biases.

The method is linear-algebraic, computationally efficient, and applicable to both univariate and multivariate factor/response settings.