Knowledge Manifold Analysis Specification (2D, with Uniform-Placement Constraint) - Revised Version v2.0

Purpose of This Revision

This revised version prevents the issue in which geodesic optimization could be skipped by a runtime fallback and the straight initial path could be output as-is. The specification is designed so that the geodesic is reliably output from the beginning as an optimized path.

In this specification, the primary objective of the 2D knowledge map is to preserve semantic distance relationships based on cosine distance, while the secondary objective for visualization is to spread points as uniformly as possible within the region `-1 < x < 1`, `-1 < y < 1`.

Methods not described in this specification must not be changed arbitrarily. Any steps where results may vary due to differences in the execution environment or library versions must be recorded in `manifest.json` and `analysis_report.md` under "Sources of possible variation".

---

1. Common Conditions

1. Extract PDF body text in page order. Figure/table captions and references must be included. Do not use OCR. Record page numbers only for pages from which text cannot be extracted.
2. Text preprocessing is limited to Unicode NFKC normalization, lowercasing of alphabetic characters, and replacement of consecutive whitespace with a single space. Do not perform stop-word removal, stemming, translation, or summarization.
3. TF-IDF must use an implementation equivalent to scikit-learn's `TfidfVectorizer`, with the following settings fixed:
   - `analyzer="char_wb"`
   - `ngram_range=(4,7)`
   - `sublinear_tf=True`
   - `norm="l2"`
   - `max_features=250000`
   - `min_df=1`
4. If the number of features exceeds 250,000, fix the TF-IDF vocabulary by following the library's standard vocabulary ordering and feature-limit handling, not by frequency order.
5. Use `random_state=0` for every procedure that uses randomness.
6. L2-normalize every document vector. All subsequent similarities are cosine similarities.
7. Define the distance between documents as follows:

```
cosine distance = 1 - cosine similarity
```

---

2. 2D Knowledge Map

2.1 Selection of Four Representative Papers

1. Select the representative four papers by exhaustive search over all 4-paper combinations from the 20 papers. Choose the combination that maximizes the sum of pairwise cosine distances among the four papers.
2. If there is a tie, select the first combination in lexicographic order of file names.
3. Sort the selected representative four papers in lexicographic order of file names and fix them to the following four corners in order:
   - 1st paper: `(-1,-1)`
   - 2nd paper: `(1,-1)`
   - 3rd paper: `(1,1)`
   - 4th paper: `(-1,1)`

2.2 Placement of the Remaining 16 Papers

Place the remaining 16 papers while keeping the representative four papers fixed. The placement must preserve semantic distance relationships based on cosine distance while spreading points as uniformly as possible within the 2D region. Uniformization is a secondary objective for visualization; semantic distance preservation is the primary objective.

Determine the coordinates of the remaining 16 papers by minimizing the following objective function `J`:

```
J = E_stress + lambda_rep E_rep + lambda_var E_var + lambda_center E_center
```

Distance-Preservation Term

```
E_stress = sum_{i<j} w_ij ( ||y_i-y_j|| - alpha d_ij )^2 / sum_{i<j} w_ij (alpha d_ij)^2
```

Here, `d_ij` is the cosine distance between documents i and j, `y_i` is the 2D coordinate of document i, and `w_ij=1`. `alpha` is a fixed scale factor chosen so that the average 2D distance among the representative four papers corresponds to the average cosine distance among the representative four papers.

Repulsion Term

```
E_rep = 2/(N(N-1)) sum_{i<j} exp( -||y_i-y_j||^2 / (2 sigma^2) )
```

Use `N=20` and `sigma=0.35`.

Variance Term

```
E_var = ((var_x - 1/3)^2 + (var_y - 1/3)^2) / (1/3)^2
```

Centering Term

```
E_center = mean_x^2 + mean_y^2
```

Coefficients

```
lambda_rep = 0.10
lambda_var = 0.05
lambda_center = 0.01
```

2.3 Optimization Conditions

1. Use `L-BFGS-B` for optimization.
2. Impose the following bounds on the 16 papers other than the representative four:
   - `-0.98 <= x <= 0.98`
   - `-0.98 <= y <= 0.98`
3. Fix only the representative four papers to the corners `(-1,-1)`, `(1,-1)`, `(1,1)`, and `(-1,1)`.
4. The initial values for optimization must be the SMACOF/MDS result with `random_state=0`, affinely aligned to the representative four papers.
5. Do not apply a single linear scaling to all final coordinates.
6. Save the final coordinate CSV as `locked_knowledge_map_coordinates.csv`. In all subsequent SPH, GPR, gradient, and geodesic analyses, do not recompute the 2D map; instead, load this locked coordinate CSV.

2.4 Outputs for the 2D Knowledge Map

The following outputs are required:

1. Coordinate CSV
2. `locked_knowledge_map_coordinates.csv`
3. Cosine similarity matrix CSV
4. Cosine distance matrix CSV
5. Placement objective-function summary JSON
6. 2D knowledge map PNG
7. 2D knowledge map PDF
8. Schematic PNG
9. Schematic PDF

The objective-function summary JSON must include:

- `E_stress`
- `E_rep`
- `E_var`

- E_center
- J
- Mean nearest-neighbor distance
- Minimum nearest-neighbor distance
- Standard deviation of nearest-neighbor distance
- var_x
- var_y
- mean_x
- mean_y

---

3. SPH Interpolation

3.1 Evaluation Point

```
p = (0.25, 0.75)
```

3.2 SPH Kernel

Use a 2D cubic spline kernel as the SPH kernel. For q=r/h, define it as follows:

```
0 <= q < 1: W = 1 - 1.5 q^2 + 0.75 q^3
1 <= q < 2: W = 0.25 (2-q)^3
q >= 2: W = 0
```

To use all 20 papers, fix the smoothing length h as follows:

```
h = max_i distance(p,p_i) / 1.98
```

Normalize SPH weights as follows:

```
w_i = W_i / sum_j W_j
```

Define the interpolated knowledge vector as follows:

```
v(p) = normalize(sum_i w_i x_i)
```

Here, x_i is the L2-normalized TF-IDF vector of document i.

3.3 Outputs for SPH Interpolation

The following outputs are required:

1. SPH weights by document CSV
2. Normalized contribution rate by document CSV
3. Top TF-IDF features CSV
4. Semantic category aggregation CSV
5. Verbalization Markdown
6. SPH weight schematic PNG
7. SPH weight schematic PDF

Extract top TF-IDF features and verbalize them in Japanese using character n-gram fragments as evidence. Do not add concepts that are not present in the TF-IDF features.

---

4. SPH Knowledge Gradient

4.1 Definition

The knowledge gradients in the x and y directions are the partial derivatives of the normalized SPH interpolation vector `v(x,y)`.

```
x direction: partial v / partial x
y direction: partial v / partial y
```

If analytic differentiation is difficult to implement, use a central difference with `delta=1e-3`.

4.2 Similarity Between Directions

Evaluate the similarity between directions using both of the following:

1. Inner product `<partial v/partial x, partial v/partial y>`
2. Cosine similarity

4.3 Direction Definitions

```
0-degree direction: partial v / partial x
45-degree direction: (partial v / partial x + partial v / partial y) / sqrt(2)
90-degree direction: partial v / partial y
```

For each direction, separately verbalize in Japanese the TF-IDF features that increase in the positive direction and those that increase in the reverse direction.

4.4 Outputs for SPH Knowledge Gradient

The following outputs are required:

1. Gradient summary CSV
2. Direction-wise TF-IDF features CSV
3. Direction similarity CSV
4. Semantic-category gradient CSV
5. Verbalization Markdown
6. Direction-vector schematic PNG
7. Direction-vector schematic PDF

---

5. GPR

5.1 Model Definition

GPR takes 2D coordinates as input and outputs TF-IDF knowledge vectors. Do not apply GPR directly to the high-dimensional TF-IDF vectors. Instead, compress them to a 10-dimensional latent representation using `TruncatedSVD` and estimate kernel hyperparameters there. Use `random_state=0`.

Fix the kernel as follows:

```
ConstantKernel * RBF(length_scale=[lx,ly]) + WhiteKernel
```

Use the following GPR settings:

```
n_restarts_optimizer=10
random_state=0
normalize_y=False
```

Compute the predictive mean as a linear combination of the 20 document vectors using the posterior linear coefficients obtained from the learned GPR kernel.

5.2 Uncertainty

Output the following uncertainty values:

1. GPR posterior standard deviation
2. GPR prior standard deviation
3. Relative uncertainty = posterior standard deviation / prior standard deviation

5.3 Contribution Rates

Compute each paper's contribution rate as follows:

```
abs(posterior linear coefficient) * max(cosine(document vector, predictive mean), 0)
```

Then normalize so that the sum over the 20 papers equals 1. If a posterior linear coefficient is negative, explain that this does not mean the paper is unnecessary; rather, it is a corrective contribution that adjusts the local mean field.

5.4 Outputs for GPR

The following outputs are required:

1. GPR summary JSON
2. Bayesian contribution-rate CSV
3. Top TF-IDF features of the GPR predictive mean CSV
4. Semantic category CSV
5. Verbalization Markdown
6. GPR schematic PNG
7. GPR schematic PDF

---

6. Generation of a Predicted Paper

Based on the knowledge vector estimated at the evaluation point `(0.25,0.75)`, generate an approximately 500-character Japanese abstract for a hypothetical/predicted paper corresponding to this position.

1. The generated text must be based on the TF-IDF features, SPH interpolation result, and GPR predictive mean.
2. Do not add unsupported new concepts, nonexistent experimental results, or nonexistent numerical values.
3. Use a style similar to an academic abstract and include the target, method, direction of results, and significance.

---

7. Geodesic (Revised Reliability Version)

7.1 Basic Definition

Define the induced metric from the SPH interpolation map `v(x,y)` as follows:

```
g_ij = < partial v / partial p_i, partial v / partial p_j >
```

Here, `p_1=x` and `p_2=y`.

Compute the geodesic as an endpoint-fixed path with the following settings:

```
Start point: (0.25, 0.75)
End point: (-0.3, 0.35)
Number of discrete points: 31
```

Define the discrete energy as follows:

```
E = sum_k Delta p_k^T g((p_k+p_{k+1})/2) Delta p_k
```

Restrict path coordinates to the following range:

```
-1 <= x <= 1
-1 <= y <= 1
```

7.2 Fixed Conditions for Metric Evaluation

1. Evaluate the metric `g` based on the SPH interpolation map constructed from the locked coordinates `locked_knowledge_map_coordinates.csv` and the L2-normalized TF-IDF document vectors.
2. Do not recompute the 2D knowledge map during geodesic analysis.
3. Compute `partial v / partial x` and `partial v / partial y` by central difference.
4. Use `delta=1e-3` as the central-difference width.
5. `v(x,y)` must always be the L2-normalized SPH interpolation vector.

6. Apply the following regularization only if the metric matrix becomes numerically singular or non-positive-definite:

```
g_regularized = g + epsilon I
epsilon = 1e-10
```

7. Record the number of points where regularization was applied, the maximum regularization amount, and the minimum eigenvalue before regularization in `geodesic_optimization_status.json`.

7.3 Optimization Variables

1. Of the 31 points, optimize only the 29 internal points excluding the start and end points.
2. Apply the bounds `-1 <= x <= 1` and `-1 <= y <= 1` to every internal point.
3. Keep the start and end points fixed; they must not be changed during optimization.

7.4 Initial Paths and Retry Conditions

Use the Euclidean straight line as the first initial path. However, to prevent the geodesic from being output as the unoptimized straight initial path, the following deterministic retries must be implemented.

Initial Path Candidates Let `a` be the amplitude of a sinusoidal perturbation in the unit-vector direction perpendicular to the straight line.

```
candidate_0: a = 0.00  # straight line
candidate_1: a = 0.01
candidate_2: a = -0.01
candidate_3: a = 0.03
candidate_4: a = -0.03
candidate_5: a = 0.05
candidate_6: a = -0.05
```

For internal points `k=1,...,29`, define the perturbation as follows:

```
p_k = straight_p_k + a * sin(pi k / 30) * n_perp
```

Here, `n_perp` is a unit vector perpendicular to the straight-line direction from the start point to the end point. If any point goes outside the allowed range after perturbation, clip each coordinate to `[-1,1]`.

7.5 L-BFGS-B Settings

Run `L-BFGS-B` for every candidate using the following settings:

```
method = "L-BFGS-B"
maxiter = 2000
```

```
maxfun = 200000
ftol = 1e-12
gtol = 1e-8
maxls = 50
```

If possible, let the optimization library use its standard approximation for the objective-function gradient rather than implementing numerical differences manually. Even in that case, record the settings above.

7.6 Success Criteria and Adoption Rules

Record the following for every candidate:

- `candidate_id`
- `initial_amplitude`
- `optimizer_success`
- `optimizer_message`
- `nit`
- `nfev`
- `initial_energy`
- `final_energy`
- `straight_energy`
- `energy_drop_from_straight`
- `relative_energy_drop_from_straight`
- `max_lateral_deviation_from_straight`
- `mean_lateral_deviation_from_straight`

Choose the adopted path using the following rules:

1. If one or more candidates have `optimizer_success=True`, adopt the one with the minimum `final_energy` among them.
2. If no candidate has `optimizer_success=True`, but at least one candidate satisfies `final_energy < straight_energy - 1e-12`, provisionally adopt the candidate with the minimum `final_energy` and record `geodesic_status="energy_improved_but_optimizer_not_success"`.
3. If no candidate satisfies either of the above conditions, do not output the straight path as a "geodesic". In this case, set `geodesic_status="failed"` and add `straight_baseline` to output file names. In `analysis_report.md`, explicitly state: "Geodesic optimization failed, so the path was not adopted as a geodesic."

7.7 Prohibited Fallbacks

The following are prohibited:

1. Reporting the straight initial path unconditionally as a geodesic due to runtime limits or optimization failure.
2. Saving only the straight path as `geodesic_path.csv` while `optimizer_success=False`.

3. Drawing only the straight initial path in the geodesic figure as if it were the optimized geodesic.
4. Omitting the energy difference between the straight path and the geodesic, maximum lateral deviation, or success status.

7.8 Comparison With the Straight Path

Always compare the straight path with the adopted geodesic and record the following in JSON and Markdown:

- Straight-path energy
- Adopted-geodesic energy
- Energy difference = straight-path energy - adopted-geodesic energy
- Relative energy reduction rate
- Straight-path Riemannian length
- Geodesic Riemannian length
- Difference in Riemannian length
- Maximum lateral deviation
- Mean lateral deviation
- Maximum deviation from the endpoint line segment
- Maximum path bend angle
- Mean path bend angle
- Optimizer success
- Optimizer message
- Adopted initial-path candidate ID

7.9 Output of Metric Variation, as a Curvature-Like Indicator

A strict Riemann curvature tensor is not required. However, to show how much the metric varies along the path, the following curvature-like indicators must be output.

Compute the following at each path point or at each interval midpoint:

- `g_xx`
- `g_xy`
- `g_yy`
- `trace_g = g_xx + g_yy`
- `det_g`
- `lambda_min`
- `lambda_max`
- `anisotropy_ratio = lambda_max / lambda_min`
- `coupling = abs(g_xy) / sqrt(g_xx g_yy)`
- `frobenius_norm_g`

Additionally, compute the following from differences between adjacent points:

- `||dg||/ds`
- `d_trace_g/ds`

- `d_det_g/ds`
- `d_anisotropy_ratio/ds`
- `d_coupling/ds`

Record the following summary values:

- `metric_variation_mean_norm_dg_ds`
- `metric_variation_max_norm_dg_ds`
- `anisotropy_ratio_min`
- `anisotropy_ratio_max`
- `coupling_min`
- `coupling_max`
- `trace_g_min`
- `trace_g_max`
- `det_g_min`
- `det_g_max`

7.10 Geodesic Outputs

The following outputs are required:

1. `geodesic_reoptimized_path.csv`
2. `geodesic_straight_baseline_path.csv`
3. `geodesic_reoptimized_metric.csv`
4. `geodesic_reoptimized_metric_variation.csv`
5. `geodesic_reoptimized_comparison.json`
6. `geodesic_optimization_status.json`
7. `geodesic_reoptimized_report.md`
8. `geodesic_reoptimized_schematic.png`
9. `geodesic_reoptimized_schematic.pdf`
10. `geodesic_reoptimized_metric_variation.png`
11. `geodesic_reoptimized_metric_variation.pdf`

The figure must include all of the following:

- The 20 document points
- Start point and end point
- Straight path
- Adopted geodesic
- Position of maximum lateral deviation
- Even if the adopted geodesic almost coincides with the straight path, display the energy difference and maximum lateral deviation in the figure caption.

---

8. Outputs and Reproducibility Checks

8.1 Required Outputs

Save all numerical results as CSV or JSON. Save every figure in both PNG and PDF formats.

The final response must explicitly state the following:

1. Main parameters
2. Random seed
3. TF-IDF dimensionality
4. Representative four papers
5. Evaluation point
6. SPH smoothing length
7. GPR kernel
8. GPR uncertainty
9. Direction similarity
10. Geodesic optimization status
11. Adopted geodesic candidate ID
12. Straight-path energy
13. Geodesic energy
14. Energy difference
15. Relative energy reduction rate
16. Maximum lateral deviation
17. Geodesic length
18. Comparison with the straight path
19. Metric-variation summary
20. Main output files
21. Sources of possible variation

8.2 Reproducibility Files

For reproducibility checks, the following outputs are required:

1. `manifest.json`
2. `pdf_text_extraction_summary.csv`
3. `analysis_report.md`
4. Python version used
5. Versions of major libraries used
6. List of input PDF file names
7. Extracted character count for each input PDF
8. `geodesic_optimization_status.json`
9. CSV or JSON log for all geodesic optimization candidates

8.3 Sources of Possible Variation

The following must be listed:

- Differences in PDF text-extraction libraries
- Differences in scikit-learn's TF-IDF vocabulary ordering and feature-limit handling
- Differences in SMACOF/MDS implementations
- Differences in L-BFGS-B implementations and stopping criteria
- Local optimum differences in GPR hyperparameter optimization
- Floating-point arithmetic and BLAS/LAPACK differences
- Font and rendering differences in matplotlib

8.4 Final-Response Template for Geodesic Analysis

When the final response includes geodesic analysis, report at least the following in the format below:

```
Geodesic optimization: success / failed / energy_improved_but_optimizer_not_success
Adopted candidate ID: ...
Straight-path energy: ...
Geodesic energy: ...
Energy difference: ...
Relative energy reduction rate: ...
Maximum lateral deviation: ...
Mean metric variation ||dg||/ds: ...
Maximum metric variation ||dg||/ds: ...
Anisotropy eigenvalue ratio: min ... / max ...
Coupling coefficient |g_xy|/sqrt(g_xx g_yy): min ... / max ...
```

---

9. Important Changes in This Revised Version

1. Prohibited the straight-line fallback for geodesic optimization.
2. Specified the maximum iterations, tolerances, and line-search count for L-BFGS-B.
3. Fixed multiple initial path candidates, including the straight path and positive/negative sinusoidal perturbations, and required deterministic retries.
4. Explicitly defined success criteria and adoption rules.
5. Required output of energy difference, relative reduction rate, and maximum lateral deviation even when the geodesic is close to a straight line.
6. Required metric "curvature-like" variation to be output as pathwise variation in `||dg||/ds`, anisotropy, coupling coefficient, trace, and determinant, rather than as strict curvature.
7. Prohibited displaying or naming the straight path as the geodesic when geodesic optimization fails.