**Spectral Clustering: Methodology and Implementation Guide**

**Executive Overview**

This project reproduces and benchmarks **normalized** and **unnormalized spectral clustering algorithms** as described in Ulrike von Luxburg’s tutorial paper "A Tutorial on Spectral Clustering". It provides Python implementations and visual/quantitative evaluations using the Iris and Wine datasets.

**1. Fundamentals of Spectral Clustering**

**Spectral clustering** uses eigenvectors of a matrix (graph Laplacian) derived from the data, transforming clustering into a geometric problem involving data representation. The methodology is grounded in graph theory and linear algebra.

**Key Steps:**

* Data points are represented as nodes in a similarity graph.
* A similarity (affinity) matrix is computed, often using a Gaussian kernel.
* From this, a Laplacian matrix is derived.
* The first eigenvectors transform the data into a lower-dimensional space.
* Standard clustering (e.g., k-means) is performed on this new representation.

**2. Methodological Procedure**

**A. Data Preparation**

* Use standard datasets: **Iris** (multiclass, well-separated clusters) and **Wine** (multiclass, with more features).
* Preprocess (scaling, normalization) where necessary.

**B. Similarity Matrix Construction**

* Compute affinity between data points using a **Gaussian similarity function**:
* For large or structured data, a **k-nearest neighbor graph** can also be used.

**C. Laplacian Matrix Computation**

* **Unnormalized Laplacian (L):**

Where is the degree matrix (), and is the similarity matrix.

* **Normalized Laplacian (L**sym**):**

**D. Eigen-Decomposition**

* Compute the **first eigenvectors** of the Laplacian.
* For Lsym, normalize each row of the eigenvector matrix to unit length (Ng et al. 2002).[[1]](#fn1)

**E. Clustering in Embedded Space**

* Form a new matrix using the first (number of clusters) eigenvectors.
* Use **k-means clustering** on the rows of this matrix.

**F. Evaluation and Label Alignment**

* Quantitative metrics:
  + **Adjusted Rand Index (ARI):** Measures clustering similarity to the ground truth.
  + **Clustering Accuracy:** After optimal matching (Hungarian algorithm) between predicted and true labels.
  + **Graph Cut Value:** Sum of edge weights between different clusters.
* Visualizations:
  + **Graph plots**: Nodes colored by cluster label, edges thresholded for clarity.
  + **Scatter plots (PCA/feature space):** Compare true labels and clustering results.

**3. Implementation Details**

**Unnormalized Spectral Clustering**

Files: IRIS-UNNORMALLIZED.py, WINE-UNNORMALIZED.py

* Construct similarity matrix (Gaussian or kNN).
* Compute degree matrix and Laplacian .
* Extract first eigenvectors.
* Cluster with k-means.
* For label alignment, use the Hungarian algorithm.
* Evaluate performance using ARI, clustering accuracy, and graph cut value.
* Visualize graph structure and clustering in reduced dimensions.

**Normalized Spectral Clustering (Ng et al. 2002)**

Files: IRIS-NORMALIZED.py, WINE-NORMALIZED.py

* Similar affinity matrix construction.
* Compute normalized Laplacian .
* Extract first eigenvectors.
* **Row normalization:** Normalize each row vector to unit length before clustering.
* Evaluation and visualization as above.

**References**

Tutorial: "A Tutorial on Spectral Clustering", Ulrike von Luxburg, 2007.

**4. Thought Process and Design Decisions**

* **Why spectral clustering?**
  + Handles complex cluster shapes, non-globular clusters, and manifold structures.
  + Connects clustering to graph theory, enabling partitioning based on global structure.
* **Why both normalized and unnormalized?**
  + Unnormalized clustering is mathematically simpler but can fail when graph degree varies widely.
  + Normalized clustering (especially with row normalization) is robust to degree variation and often yields better, more balanced clustering results.[[1]](#fn1)
* **Parameter Choices:**
  + **Sigma/gamma (Gaussian):** Chosen based on data distribution or by domain heuristics.
  + **Number of clusters ():** Set to true number for datasets with labels (can use eigengap heuristic otherwise).
* **Evaluation:**
  + Use both external (true labels) and internal (cut value) validation for credibility.
  + Visualizations help interpret results and spot clustering issues.

**5. How to Extend**

* Swap in other datasets via the provided scripts/notebooks.
* Change similarity graph construction (e.g., k-nearest neighbor).
* Benchmark new clustering algorithms.
* Use as a template for reproducing additional spectral clustering papers.

**6. Folder Structure Reference**

* **docs/**: This methodology and technical reference.
* **clustering/**: Implementation code (modular).
* **experiments/**: Notebooks for dataset-wise benchmarking.
* **results/**: ARI, accuracy, cut values, and visualizations.
* **papers/**: Contains key reference papers.[[1]](#fn1)

**Conclusion**

This project is a reproducible benchmark suite for spectral clustering, illustrating principle-based implementations from key literature. Anyone can follow the methodology above to understand, reproduce, and extend these algorithms with new data or settings.

⁂