Spatial Analysis and Modeling

Spatial Clustering and Regionalization

Corso di formazione su ML e DL
Fondazione LINKS
25/09/2025

ISI Foundation

Why Spatial Clustering?

Standard Clustering

- Groups similar observations
- Ignores spatial relationships
- May produce fragmented results
- Distance in feature space only

Spatial Clustering

- **Spatial autocorrelation** matters
- Geographic proximity + similarity
- Spatially coherent clusters
- Real-world geographical meaning

> **Tobler's First Law**: "Everything is related to everything else, but near things are more related than distant things"

Key Challenges in Spatial Data

- **Balancing Act**
  - **Attribute similarity** vs. **spatial proximity**
  - Risk of spatially fragmented clusters
  - Need for **geographical coherence**
- **Spatial Outliers**
  - Observations different from neighbors
  - Not global outliers, but local anomalies
  - Can distort conventional algorithms
- **Contiguity Constraints**
  - Need for connected regions
  - Applications: districts, territories, zones
  - **Regionalization** problem

Spatial Clustering

Density-Based Methods

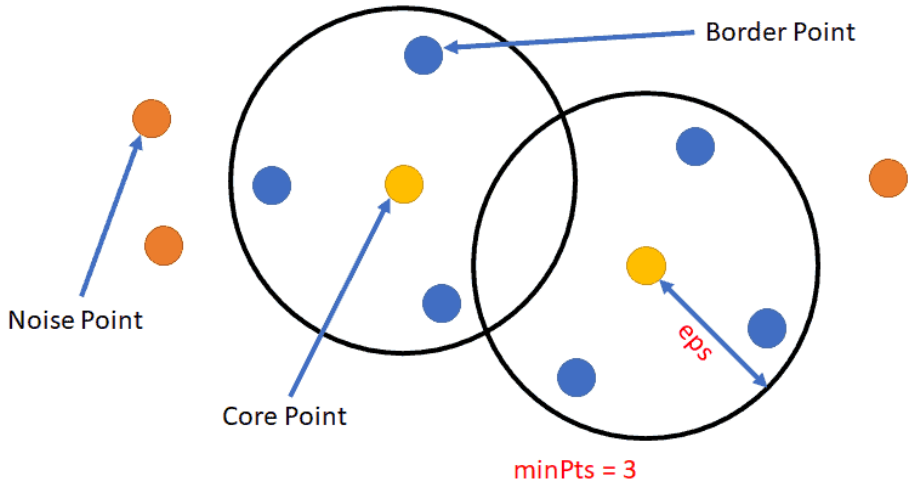
DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Key Parameters:

- **eps (ε)** : Maximum neighbor distance
- **MinPts** : Minimum points per cluster

Advantages:

- Arbitrary cluster shapes
- Robust to noise/outliers
- No predefined cluster count



Limitations:

- Struggles with clusters of varying density
- Poor performance in high-dimensional spaces (distance concentration) and with many noisy/irrelevant features.
- Border-point ambiguity
- Not hierarchical — no multi-scale cluster hierarchy (use OPTICS/HDBSCAN for that).
- Can be computationally expensive
- Sensitive to non-uniform sampling and clusters with very different sizes.

DBSCAN Variants and Extensions

Enhanced Versions:

- **ADBSCAN**: Adaptive parameter estimation
- **SS-DBSCAN**: Stratified sampling for ε estimation
- **STRP-DBSCAN**: Parallel processing for trajectory data

Applications:

- Crime hotspot detection
- Environmental monitoring
- Urban planning
- Transportation analysis

HDBSCAN — Hierarchical DBSCAN

What is HDBSCAN?

- HDBSCAN is a **hierarchical**, **density-based** clustering extension that builds a cluster hierarchy from DBSCAN-like density estimates and extracts the most stable clusters.
- **Key ideas:** **no fixed eps**, clusters persist across scales, stability score for clusters.
- Good when data have **variable-density** clusters and when tuning a single ε is hard.

Key parameters

- **min_cluster_size** — minimum size of any cluster (controls granularity)
- **min_samples** — (optional) controls how conservative the algorithm is (affects core distances)
- **metric** — same idea as DBSCAN (use **haversine** for lat/lon)

HDBSCAN

Advantages

- Automatically finds clusters at multiple density levels.
- Robust to variable-density clusters and noise.
- Provides soft membership through cluster probability / stability.

Tradeoffs

- More computationally intensive than plain DBSCAN.
- Still needs thoughtful choice of **`min_cluster_size`** / **`min_samples`**.
- Interpreting hierarchical output requires inspecting cluster stability.

OPTICS — Ordering Points To Identify the Clustering Structure

What is OPTICS?

- OPTICS creates an ordering of points based on density (reachability plot) instead of a single fixed ε.
- Good at revealing clustering structure across many density levels and finding clusters of varying density.
- Produces: **reachability_**, **ordering_**, **core_distances_** — useful diagnostic plots.

Key parameters

- **min_samples** — minimum points in a neighborhood (controls cluster granularity)
- **max_eps** — maximum neighborhood radius (set to large value to explore all scales)
- **metric** — use **haversine** for lat/lon (fit on radians)

Key Features:

- Specifically designed for spatial data
- Uses randomized search strategy
- Based on PAM (Partitioning Around Medoids) but more efficient
- Handles irregular cluster shapes

Advantages

- Reveals full density structure (**reachability plot**) — helps pick meaningful cluster scales.
- Handles **variable-density** data better than single-ε DBSCAN.
- Can be post-processed to extract flat clusters at chosen eps or using **xi**-style extraction.

Tradeoffs

- More effort to interpret.
- Computationally heavier than DBSCAN (but still scalable).
- Often paired with visual/manual inspection or automatic extraction rules.

Regionalization

What is Regionalization?

Spatial Contiguity Constraint

Unlike standard clustering, regionalization ensures that:

- All cluster members are **geographically connected**
- Forms single, unbroken areas
- No spatial fragmentation
- Practical for administrative boundaries

Applications:

- Electoral districts
- Sales territories
- Ecological zones
- Health service areas
- School catchment areas

SKATER Algorithm

SKATER (Spatial K'luster Analysis by Tree Edge Removal)

Algorithm Steps:

1. Build Minimum Spanning Tree (MST)
2. Remove edges to maximize dissimilarity
3. Creates spatially contiguous regions
4. Balances homogeneity and contiguity

Parameters:

- Number of clusters
- Floor constraint (minimum size)
- Attribute variables

```python
import spopt
from libpysal.weights import Queen

# Spatial weights
w = Queen.from_dataframe(gdf)

# SKATER clustering
model = spopt.region.Skater(
    gdf, w, attrs_name,
    n_clusters=5,
    floor=3
)
model.solve()

# Get results
gdf["clusters"] = model.labels_
```

References: Assunção et al. (2006), Rey & Anselin (2007)

Max-P Regions

Max-P: Maximum number of regions with constraints

Objective:

- Maximize number of regions
- Subject to threshold constraints
- Maintain spatial contiguity
- Minimize within-region heterogeneity

Example Use Cases:

- Population-based districts
- Service coverage areas
- Resource allocation zones

References: Duque et al. (2012), Laura & Duque (2017)

Python Implementation:

```
# Max-P regionalization
model = spopt.region.MaxPHeuristic(
    gdf, w, attrs,
    threshold_name="population",
    threshold=10000
)
model.solve()

print(f"Found {model.p} regions")
gdf["maxp_labels"] = model.labels_
```

AZP — Automatic Zoning Procedure

Objective:

- Iteratively relocate spatial units to improve a global objective (typically within-region homogeneity).
- Starts from an initial partition (random or seeded) and moves boundary units between neighboring zones when such moves reduce the objective.

Key ideas

- Minimize **within-region variance** (or another heterogeneity measure).
- Parameters: number of regions (**p**), **floor** (minimum region size), **attributes driving homogeneity** .
- Stochastic method: multiple random starts to avoid poor local optima.

Advantages

- Flexible and fast for moderate-sized problems.
- Produces compact, contiguous regions suitable for administrative or operational zones.
- Easy to tune for a target number of regions.

Tradeoffs

- Local optima (use many starts).
- Sensitive to initial seed and floor constraints.
- Not guaranteed to find a global optimum; evaluation and multiple runs are required.

Hierarchical Spatial Clustering

Ward with Spatial Constraints

Traditional Approach:

- Standard linkage criteria
- No spatial consideration
- May create fragmented clusters

Spatially Constrained:

- Connectivity matrix from spatial weights
- Only adjacent units can merge
- Guarantees spatial contiguity

```
from sklearn.cluster import AgglomerativeClustering
from libpysal.weights import Queen

# Spatial weights
w = Queen.from_dataframe(df)

# Constrained clustering
model = AgglomerativeClustering(
    linkage="ward",
    connectivity=w.sparse,
    n_clusters=5
)
model.fit(scaled_data)
```

Mixed Approaches

ClustGeo: Geography + Attributes

- Combines geographical and feature space dissimilarity
- Weighted balance between spatial and attribute similarity
- Parameter α controls spatial constraint strength

GWR-Based Clustering:

- Uses Geographically Weighted Regression coefficients
- Identifies regions with similar local relationships
- Captures spatial heterogeneity in relationships

STICC (Spatial Toeplitz Inverse Covariance-Based Clustering):

- Markov random field approach
- Considers both attributes and spatial relationships
- Encourages spatial consistency

References: Chavent et al. (2018), Kang et al. (2022)

Tools

spopt - Spatial Optimization

- **SKATER**, **Max-P**, **AZP** algorithms
- Part of PySAL ecosystem
- Comprehensive regionalization tools

scikit-learn - Machine Learning

- **DBSCAN** with spatial metrics
- **AgglomerativeClustering** with connectivity
- Preprocessing and evaluation tools

Resources

Jupyter Notebooks:

1. **Geographic Data Science Book**
   - https://geographicdata.science/book/notebooks/10_clustering_and_regionalization.html
   - Complete tutorial with San Diego data
   - K-means, hierarchical, and regionalization
2. **PySAL Tutorials**
   - https://github.com/sjsrey/pysal-scipy22
   - SciPy conference materials
   - Hands-on geodemographic analysis
3. **spopt Documentation**
   - https://pysal.org/spopt/notebooks/skater.html
   - SKATER tutorial with Chicago Airbnb data

Key Parameters and Evaluation

Parameter Selection Guidelines:

DBSCAN:

- Use k-distance plots for ε
- Consider geographic scale
- Haversine metric for lat/lon

SKATER:

- Floor parameter for minimum size
- Attribute standardization crucial
- Balance homogeneity vs. contiguity

Max-P:

- Threshold drives number of regions
- Multiple runs for best solution
- Consider population distribution

Hierarchical:

- Linkage criterion matters
- Spatial weights construction
- Dendrogram interpretation

Evaluation Metrics

Spatial Coherence:

- **Isoperimetric Quotient**: Measures compactness
- **Join Count Statistics**: Tests spatial contiguity
- **Spatial Autocorrelation**: Moran's I within clusters

Feature Coherence:

- **Calinski-Harabasz Score**: Cluster separation
- **Silhouette Score**: Individual assignment quality
- **Within-cluster Sum of Squares**: Homogeneity

Solution Comparison:

- **Adjusted Mutual Information**: Compare different solutions
- **Rand Index**: Clustering agreement
- **Modularity**: Network-based quality

References

- Ester, M., et al. (1996). A density-based algorithm for discovering clusters. *KDD-96*.
- Ng, R. T., & Han, J. (1994). Efficient and effective clustering methods for spatial data mining. *VLDB*.
- Assunção, R. M., et al. (2006). Efficient regionalization techniques for socio-economic geographical units. *International Journal of Geographical Information Science*.
- Kang, Y., et al. (2022). STICC: A multivariate spatial clustering method. *arXiv:2203.09611*.
- Chavent, M., et al. (2018). ClustGeo: an R package for hierarchical clustering with spatial constraints. *Computational Statistics*.