TasteRank Explorer

Technical Appendix

Mathematical Framework, Algorithms, and Implementation

Jure Skarabot · May 2026

Companion documents: Summary · Technical Appendix · Methods Primer · Data Appendix · Grape Reference

---

A. Mathematical Framework

A.1 Feature Space Definition

Let $V = \{v_1, v_2, \ldots, v_{101}\}$ be the set of 101 grape varieties. Each variety v_i is represented by a profile vector

$$\mathbf{p}_i = (p_{i,1}, p_{i,2}, \ldots, p_{i,13}) \in \mathbb{R}^{13},$$

where each dimension corresponds to one of the thirteen sensory dimensions, scored on integer scales as defined in the Data Appendix. The profile matrix $P \in \mathbb{R}^{101 \times 13}$ collects all variety profiles row-wise.

A.2 Cosine Similarity

The pairwise similarity between varieties i and j is computed as

$$s_{ij} = \cos(\theta_{ij}) = \frac{\mathbf{p}_i \cdot \mathbf{p}_j}{\|\mathbf{p}_i\|_2 \, \|\mathbf{p}_j\|_2} = \frac{\sum_k p_{i,k}\, p_{j,k}}{\sqrt{\sum_k p_{i,k}^2}\,\sqrt{\sum_k p_{j,k}^2}}.$$

This produces the full similarity matrix $S \in \mathbb{R}^{101 \times 101}$ with $s_{ii} = 1$ and $0 \leq s_{ij} \leq 1$ (since all profile entries are non-negative). The choice of cosine over Euclidean distance is motivated by three considerations: (1) cosine is scale-invariant, measuring profile shape rather than magnitude; (2) it is bounded in $[0, 1]$ for non-negative vectors, providing a natural similarity interpretation; (3) it handles the tannin dimension (which is structurally 0 for most whites) more gracefully than Euclidean distance.

A.3 kNN Graph Construction

The graph $G = (V, E, w)$ is constructed as follows.

Step 1 (Neighbour selection). For each variety v_i, identify the $k = 5$ varieties with highest cosine similarity:

$$N_k(v_i) = \operatorname*{top\text{-}k}_{j \neq i} \{s_{ij}\}.$$

Step 2 (Symmetrisation). Create an undirected edge between v_i and v_j if $v_j \in N_k(v_i)$ *or* $v_i \in N_k(v_j)$. The edge weight is $w_{ij} = s_{ij}$.

Step 3 (Maximum-weight union). When both varieties include each other in their top-k, the single edge carries the cosine similarity weight (which is identical by symmetry of cosine).

The resulting graph has $|E| = 341$ edges. The average degree is $2 \cdot 341/101 \approx 6.75$, reflecting the symmetrisation which can push effective degree above k.

A.4 Weighted Adjacency Matrix

The weighted adjacency matrix $W \in \mathbb{R}^{101\times 101}$ is defined as

$$W_{ij} = \begin{cases} s_{ij} & \text{if } (v_i, v_j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

W is symmetric ($W = W^\top$), non-negative, and sparse (density $= 341/5050 \approx 0.0675$).

A.5 Eigenvector Centrality (TasteRank)

A.5.1 The Eigenvalue Problem and the Nature of the Definition

The weighted adjacency matrix W is a real, symmetric, non-negative 101×101 matrix. By the spectral theorem for real symmetric matrices, W has exactly 101 real eigenvalues (counted with multiplicity) and a corresponding set of orthogonal eigenvectors:

$$W\mathbf{x}_k = \lambda_k \mathbf{x}_k, \qquad k = 1, 2, \ldots, 101,$$

where the eigenvalues are ordered $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{101}$. Eigenvector centrality uses only the leading eigenpair $(\lambda_1, \mathbf{x}_1)$ — the eigenvector associated with the largest eigenvalue. The TasteRank score of variety i is defined as $x_{1,i}$, the i-th component of this leading eigenvector, normalised so that $\|\mathbf{x}_1\|_2 = 1$.

A critical point: TasteRank has no closed-form analytical expression. There is no formula of the form $\mathrm{TR}(i) = f(W_{i,1}, W_{i,2}, \ldots)$ that can be evaluated independently for a single variety by examining only its row of the adjacency matrix. The definition is implicit: $\mathrm{TasteRank}(i) = x_{1,i}$ where $\mathbf{x}_1$ is the vector that simultaneously satisfies the system of 101 coupled linear equations $W\mathbf{x}_1 = \lambda_1\mathbf{x}_1$. Every variety's score depends on every other variety's score.

This implicit, system-level nature is not a limitation but the defining feature of the measure. It is exactly analogous to Google's PageRank: there is no formula for any individual web page's rank; the rank is a fixed point of the entire hyperlink graph. Similarly, a grape variety's TasteRank is not a local property of its sensory profile or its immediate neighbours — it is a global property of the variety's position within the full network topology. Changing a single edge anywhere in the graph, even between two varieties far removed from variety i, can in principle alter variety i's TasteRank score. This is what makes eigenvector centrality fundamentally relational.

The operational definition of TasteRank — the way the implicit system is actually solved — is the limit of the power iteration process:

$$\mathbf{x}_1 = \lim_{t\to\infty} \frac{W^t \mathbf{z}_0}{\|W^t \mathbf{z}_0\|}$$

for any starting vector $\mathbf{z}_0$ with positive projection onto $\mathbf{x}_1$. This convergence is guaranteed by the Perron–Frobenius theorem (Section A.5.2) and is practically achieved in 35–45 iterations (Section A.5.3).

A.5.2 Why the Largest Eigenvalue Determines the Ranking

Consider any initial vector of scores $\mathbf{z}_0$. Repeatedly applying the adjacency matrix produces $\mathbf{z}_t = W^t\mathbf{z}_0$. Since $\mathbf{z}_0$ can be decomposed in the eigenbasis as $\mathbf{z}_0 = c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \cdots + c_{101}\mathbf{x}_{101}$, applying W repeatedly gives

$$\mathbf{z}_t = c_1\lambda_1^t\mathbf{x}_1 + c_2\lambda_2^t\mathbf{x}_2 + \cdots + c_{101}\lambda_{101}^t\mathbf{x}_{101}.$$

As t grows, the term $c_1\lambda_1^t\mathbf{x}_1$ dominates because $\lambda_1 > \lambda_2$ (by Perron–Frobenius, the leading eigenvalue of a connected non-negative matrix is strictly dominant). All other terms decay exponentially relative to the first: the ratio $|\lambda_2/\lambda_1|^t \to 0$ as $t \to \infty$. After normalisation, $\mathbf{z}_t/\left\|\mathbf{z}_t\right\|$ converges to $\mathbf{x}_1$ regardless of the starting vector.

This is the mathematical justification for why λ_1 alone determines the centrality ranking. The higher eigenvalues $\lambda_2, \lambda_3, \ldots$ describe successively weaker modes of network structure that are relevant to community detection and spectral clustering (see A.7), but they wash out of the centrality computation.

A.5.3 The Spectral Gap and Convergence

The rate at which power iteration converges to $\mathbf{x}_1$ depends on the spectral gap, defined as the ratio $|\lambda_2/\lambda_1|$. A smaller ratio means faster convergence. For the TasteRank graph, the spectral gap is approximately 0.72, meaning that each iteration of power iteration reduces the contribution of the second eigenvector by a factor of approximately 0.72. After t iterations, the second-order contamination is $(0.72)^t$ — after 25 iterations this is below 10^{-3}, and after 50 iterations below 10^{-7}. This explains why the computation converges well within 50 iterations at tolerance $\varepsilon = 10^{-6}$.

The spectral gap also carries structural meaning. A large gap (small λ_2/λ_1) indicates that the network has a single, clearly dominant mode of centrality — a strong core–periphery structure. A small gap (ratio near 1) would indicate competing centres of influence. For TasteRank, the moderate gap of 0.72 reflects the fact that Community C0 (Mediterranean reds) forms a dense core, but Communities C2–C4 provide secondary structure that is not negligible.

A.5.4 The Recursive Interpretation

The eigenvector equation $W\mathbf{x} = \lambda_1\mathbf{x}$ can be written component-wise as

$$x_i = \frac{1}{\lambda_1}\sum_j W_{ij}\,x_j.$$

This is not a formula that can be evaluated for a single variety in isolation. It is one equation in a system of 101 coupled equations: x_i depends on x_j for all neighbours j, each of which depends on its own neighbours' scores, and so on. The eigenvector is the unique self-consistent solution to this infinite chain of mutual dependencies. The TasteRank score of variety i is proportional to the similarity-weighted sum of the TasteRank scores of its neighbours, which are themselves defined by their neighbours' scores, ad infinitum.

A.5.5 Role of Higher Eigenvalues

While λ_1 and $\mathbf{x}_1$ determine the centrality ranking, the subsequent eigenvalues carry complementary structural information. The second eigenvector $\mathbf{x}_2$ (associated with λ_2) captures the primary axis of variation orthogonal to centrality — in network science, this is closely related to the Fiedler vector used for spectral bisection. For the TasteRank graph, $\mathbf{x}_2$ approximately separates red varieties (positive components) from white varieties (negative components), reflecting the strongest structural divide in the network.

The third and fourth eigenvectors further partition the network into sub-communities. Collectively, the first few eigenvectors define a low-dimensional embedding of the grape variety space (spectral embedding), which is closely related to the force-directed layout used in the TasteRank Explorer visualisation.

A.5.6 Power Iteration Algorithm

Algorithm 1 Power iteration for TasteRank

Require: Symmetric non-negative matrix W, tolerance ε, maximum iterations $T_{\max}$
Ensure: Leading eigenvector $\mathbf{x}_1$ and eigenvalue λ_1

$\mathbf{x}_0 \leftarrow (1/\sqrt{n}, 1/\sqrt{n}, \ldots, 1/\sqrt{n})$
for $t = 0, 1, 2, \ldots, T_{\max}$ **do**
  $\mathbf{x}_{t+1} \leftarrow W\mathbf{x}_t$
  $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_{t+1} / \|\mathbf{x}_{t+1}\|_2$
  if $\|\mathbf{x}_{t+1} - \mathbf{x}_t\|_2 < \varepsilon$ **then break**
  end if
end for
$\lambda_1 \leftarrow \mathbf{x}_{t+1}^\top W \mathbf{x}_{t+1}$ ▷ Rayleigh quotient
return $(\mathbf{x}_{t+1}, \lambda_1)$

For the TasteRank graph, convergence is achieved in approximately 35–45 iterations, well within the 2000-iteration budget. The final centrality vector $\mathbf{x}_1$ is then sorted in descending order to produce the TasteRank ranking.

A.6 PageRank as Robustness Check

A.6.1 From Eigenvector Centrality to PageRank

PageRank, developed by Brin and Page (1998) for ranking web pages, is a modified form of eigenvector centrality that addresses a specific pathology: in networks with dense subgraphs, pure eigenvector centrality can concentrate almost entirely within the densest cluster, leaving nodes in sparser regions with near-zero scores. PageRank mitigates this by introducing a damping (or teleportation) mechanism.

The key modification is to replace the adjacency matrix W with a convex combination of the row-normalised transition matrix M and a uniform matrix:

$$G = \alpha M + (1 - \alpha)\frac{1}{n}J,$$

where $M_{ij} = W_{ij} / \sum_k W_{ik}$ is the transition probability from i to j (proportional to edge weight), J is the $n \times n$ all-ones matrix, and $\alpha \in (0, 1)$ is the damping factor. The PageRank vector $\boldsymbol{\pi}$ is

the stationary distribution:

$$\boldsymbol{\pi} = G^\top \boldsymbol{\pi}, \qquad \text{equivalently:} \qquad \boldsymbol{\pi} = \alpha M^\top \boldsymbol{\pi} + (1-\alpha)\frac{1}{n}\mathbf{1}.$$

A.6.2 The Random Surfer Interpretation

The damping factor α admits a probabilistic interpretation. A random taster navigates the variety network. At each step, the taster either (a) with probability α follows an edge to a similar variety, choosing among neighbours with probability proportional to cosine similarity weight, or (b) with probability $(1-\alpha)$ teleports to a random variety. The PageRank of each variety is the long-run fraction of time the random taster spends at that variety. With $\alpha = 0.85$, the random taster follows edges 85% of the time and teleports 15% of the time. This ensures that even isolated or peripheral varieties receive some baseline importance (the teleportation floor of $(1-\alpha)/n \approx 0.0015$ per variety), preventing the extreme concentration that can occur with pure eigenvector centrality.

A.6.3 Structural Differences from Eigenvector Centrality

The two measures agree strongly overall (Spearman $\rho \approx 0.92$) but diverge in predictable ways. PageRank rewards bridge varieties more than eigenvector centrality does. A variety connected to multiple communities serves as a bottleneck in the random walk — the random taster passes through it when moving between clusters. Eigenvector centrality, by contrast, rewards varieties embedded deep within the densest cluster, regardless of bridge position.

The most notable divergence is Sangiovese: TasteRank rank 40 vs. PageRank rank ~ 8. Sangiovese connects to seven varieties spanning Communities C0 and C2, acting as a structural bridge between full-bodied Mediterranean reds and mid-weight structured reds. The gap between the two measures is itself informative: large divergences identify bridge varieties (PageRank $\gg$ TasteRank) and cluster-core varieties (TasteRank $\gg$ PageRank).

A.7 Modularity and Community Detection

A.7.1 The Modularity Function

Community detection asks whether the graph can be partitioned into groups where within-group connectivity exceeds what would be expected by chance. The modularity function Q formalises this:

$$Q = \frac{1}{2m}\sum_{i,j}\left[W_{ij} - \frac{s_i s_j}{2m}\right]\delta(c_i, c_j),$$

where $m = \frac{1}{2}\sum_{i,j} W_{ij}$ is the total edge weight, $s_i = \sum_j W_{ij}$ is the weighted degree (strength) of node i, c_i is the community assignment of node i, and δ is the Kronecker delta. The term $s_i s_j / 2m$ represents the expected edge weight under a configuration-model null. Values of $Q > 0.3$ are generally considered indicative of significant community structure.

A.7.2 Connection to the Eigenvalue Spectrum

Spectral community detection methods partition the network using the leading eigenvectors of the modularity matrix $B_{ij} = W_{ij} - s_i s_j / 2m$. The number of positive eigenvalues of B provides

an upper bound on the number of meaningful communities. For the TasteRank graph, the first six eigenvalues of B are positive and substantially separated from the bulk spectrum, consistent with the six-community partition detected by the greedy algorithm.

A.7.3 The Greedy Algorithm

The Clauset–Newman–Moore algorithm is an agglomerative approach: it starts with each node as its own community and iteratively merges the pair of communities whose merger produces the largest increase in Q. For the TasteRank graph, the algorithm terminates with six communities and a modularity of $Q \approx 0.41$. The partition is robust: alternative methods (Louvain, spectral bisection) produce partitions with similar membership and comparable modularity.

B. Graph Statistics

Statistic	Value
Nodes	101
Edges	341
k (neighbours)	5
Mean edge weight	0.9829
Median edge weight	0.9860
Min edge weight	0.9092
Max edge weight	1.0000
Communities detected	6
Graph density	0.0675

The edge weight distribution is tightly concentrated: the mean cosine similarity among connected pairs is 0.9829 with a range of $[0.9092, 1.0000]$. This reflects the high baseline similarity among wine grape varieties in sensory space — most varieties share the general shape of a wine sensory profile, with differentiation occurring in the relative magnitudes of specific dimensions.

C. Implementation

C.1 Technology Stack

The TasteRank Explorer is implemented as a single self-contained HTML file with no server-side dependencies. The technology stack consists of D3.js v7.9.0 for force-directed graph layout and interactive visualisation, loaded from the Cloudflare CDN. All data (profiles, communities, TasteRank scores) is embedded directly in the JavaScript source. The visualisation uses the Fruchterman–Reingold force-directed algorithm as implemented by D3's `forceSimulation`.

C.2 Computational Pipeline

The analytical computations were performed in Python using NetworkX for graph construction, eigenvector centrality, PageRank, and community detection. The pipeline proceeds as follows.

Stage 1 (Profile encoding). Grape varieties are encoded as thirteen-dimensional vectors across the sensory dimensions defined in the Data Appendix. Scores are stored in a JSON dictionary keyed by variety name.

Stage 2 (Similarity computation). The full 101×101 cosine similarity matrix is computed using the formula in Section A.2.

Stage 3 (Graph construction). For each variety, the $k = 5$ nearest neighbours are identified. The graph is symmetrised and weighted edges are stored.

Stage 4 (Centrality computation). NetworkX's `eigenvector_centrality` function computes TasteRank with `max_iter=2000` and `tol=1e-06`. PageRank is computed with $\alpha = 0.85$.

Stage 5 (Community detection). Clauset–Newman–Moore greedy modularity optimisation identifies six communities.

Stage 6 (Visualisation export). All computed data is serialised into JavaScript objects and embedded in the HTML file.

C.3 Force-Directed Layout

The D3.js force simulation combines four forces. The *link force* provides attraction between connected nodes with distance $= 130 - 70w$ (where w is cosine similarity weight) and strength $= 0.25w$; higher-similarity pairs are drawn closer. The *charge force* is a repulsive many-body force with strength $= -100$, preventing node overlap. The *centre force* is a gravitational pull toward the viewport centre, maintaining the graph within view. The *collision force* uses radius-based collision detection with radius $= r(\text{TasteRank}) + 4$ pixels, where r is a square-root scale from 4 px (lowest TasteRank) to 30 px (highest). The simulation runs for approximately 300 ticks before stabilising; the viewport then auto-fits to the graph bounding box with 15% padding.

C.4 Visual Encoding

Node radius maps to TasteRank via a square-root scale: $r = \sqrt{\text{TasteRank}}$ scaled to $[4, 30]$ pixels, ensuring that area is proportional to centrality. Node colour maps to community assignment using a six-colour palette designed for colour-blind accessibility. Edge opacity defaults to 0.18; on node selection, edges connected to the selected node increase to 0.75 and other edges fade to 0.015. Edge width ranges from 0.4 to 2.4 pixels based on weight. Labels are shown for the top thirty varieties by default and are extended dynamically on selection.

C.5 Interactive Features

The interface supports six interaction modes: node click (inspect variety profile and connections), community click (isolate cluster), search (filter by variety name), drag (reposition nodes), zoom/pan (scroll wheel and drag), and keyboard reset (Escape key). The left panel displays contextual information: the top fifteen varieties in default view, a full sensory profile bar chart on node selection, or a community member list on cluster selection.

C.6 Python Reference Implementation

The core analytical pipeline is approximately 80 lines of Python using NumPy and NetworkX:

```
import numpy as np
import networkx as nx
from networkx.algorithms.community import greedy_modularity_communities

# Cosine similarity
def cosine_sim(a, b):
    return np.dot(a, b) / (np.linalg.norm(a) * np.linalg.norm(b))

# kNN graph construction (k=5)
G = nx.Graph()
for grape in grapes:
    sims = sorted([(g, cosine_sim(P[grape], P[g]))
                   for g in grapes if g != grape],
                  key=lambda x: -x[1])[:5]
    for neighbor, weight in sims:
        G.add_edge(grape, neighbor, weight=weight)

# TasteRank (eigenvector centrality)
tasterank = nx.eigenvector_centrality(G, max_iter=2000,
                                      tol=1e-06, weight='weight')

# PageRank robustness check
pagerank = nx.pagerank(G, alpha=0.85, weight='weight')

# Community detection
communities = greedy_modularity_communities(G, weight='weight')
```

C.7 JavaScript Implementation (D3.js)

The browser-side implementation recomputes cosine similarity and kNN edges at runtime from the embedded profile data, ensuring consistency. The core graph construction logic in JavaScript:

```
function cosineSim(a, b) {
    let d = 0, na = 0, nb = 0;
    for (let i = 0; i < a.length; i++) {
        d += a[i] * b[i]; na += a[i] * a[i]; nb += b[i] * b[i];
    }
    return d / (Math.sqrt(na) * Math.sqrt(nb) + 1e-9);
}

// Build kNN graph (k=5)
const K = 5;
for (const g of grapes) {
    const top = Object.entries(simMat[g])
                      .sort((a, b) => b[1] - a[1]).slice(0, K);
    for (const [nb, w] of top) {
        const key = [g, nb].sort().join('|||');
        edgeMap[key] = { source: g, target: nb, weight: w };
    }
}
```

TasteRank scores and community assignments are precomputed in Python and embedded as JavaScript constants (`TASTERANK` and `COMMUNITIES` objects) to avoid reimplementing eigenvector centrality in the browser.

References

- Bonacich, P. (1972). Factoring and weighting approaches to status scores and clique identification. *Journal of Mathematical Sociology*, 2(1), 113–120.
- Brin, S. & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1–7), 107–117.
- Clauset, A., Newman, M. E. J. & Moore, C. (2004). Finding community structure in very large networks. *Physical Review E*, 70(6), 066111.
- Fruchterman, T. M. J. & Reingold, E. M. (1991). Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11), 1129–1164.
- Meyer, C. D. (2000). *Matrix Analysis and Applied Linear Algebra*. SIAM.
- Newman, M. E. J. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23), 8577–8582.
- Newman, M. E. J. (2010). *Networks: An Introduction*. Oxford University Press.