A Multi-Modal Accessibility Score for NA, EU and APAC: Construction, Decomposition, and Compute

KR&A Research

July 23, 2026

Abstract

This paper describes a methodological reference and not an empirical study. It documents a unified Pedestrian, Drive and Public Transport accessibility score covering 38 countries in Europe, North America and Developed APAC (Australia, New Zealand, Japan, Korea and Singapore). The score extends existing studies, defining accessibility by an amenity reachability and an infrastructure quality component. The former uses a Dijkstra traversal to find reachable amenities of different categories, whilst the second component is mode specific. The score is completely decomposable, with all underlying attributes provided, to ensure transparency and allow for different weightings to be used within decision processes or research. We document the exact methodology that embodies the score and the pipeline that computes them.

1 Introduction

Accessibility metrics that summarise a location's connectivity to amenities are widely used in real-estate analysis, urban planning, and consumer-facing products. Two published methodologies anchor the field. Walk Score® [3] covers North America under a single national normalisation and has, since 2011, routed over a street network rather than using Euclidean buffers; OS-WALK-EU [1] is an open-source tool assessing residential walkability for 33 German city regions, including more attributes than just the amenity proximity and using European-centric amenity categories. More recently, *A Walk across Europe* [2] constructs a seven-component walkability index at 100 m resolution across Europe from Sentinel-2, OpenStreetMap, and CORINE inputs, including greenery and road slope.

Three gaps motivate the present work.

- **Cross-continental comparability.** None of the three publishes a score with flexible normalisation to permit comparison across Europe (continental) and national. Walk Score normalises nationally within North America, and OS-WALK-EU and [2] are European, whilst our normalisations allow flexibility.
- **Commensurable modes.** Most methodologies score walking only, and where drive-time or public transport accessibility does appear, it does not take into account both amenity reachability and infrastructure quality.
- **Re-tuning cost.** Methodology choices such as component weights, decay shapes and cohort definitions are conventionally baked into the per-cell values during the heavy offline computation; re-tuning a weight then requires re-running the network routing.

We address these with a system that computes three composites from a single typed multi modal edge graph over a 250x250m grid and stores all sub-components to allow different configurations. We also introduce an improved Public Transport score that includes both amenity density and infrastructure quality, something not found in existing studies. Section 2 specifies how the score is built up: the amenity-reach component, the cost model, the three composites, and the normalisation. Section 3 is the pipeline that computes it: the method, the per-market build, the static fill, the partitioning and model, the per-cell loop, and the routing engine. Section 4 covers how the score is accessed and decomposed.

2 The score

Three composites come back for a coordinate g. The public schema exposes them as `PR` (pedestrian), `DR` (drive), and `PT` (public transport). Each is reported as a raw value plus percentile ranks at four cohort scopes (local (within LUA), national, continental and global).

2.1 Amenity reach

Amenity reach is the kernel shared by all three composites. Eight categories are scored: *grocery*, *restaurants*, *shopping*, *coffee*, *parks*, *schools*, *health*, and *entertainment*. For a mode m,

$$A_m(g) = \frac{100}{W} \sum_{c \in \mathcal{C}} w_c \cdot f_m\big(d^*_{c,m}(g)\big) \cdot b(n_{c,m}(g)), \qquad W = \sum_c w_c = 14 \tag{1}$$

where $d^*_{c,m}$ is the effective network distance to the nearest reachable POI of category c (Section 2.2), $n_{c,m}$ the count of reachable POIs in that category, f_m the mode's distance-decay, and b the density bonus

$$b(n) = 0.7 + 0.3 \cdot \frac{\min(n, 10)}{10}. \tag{2}$$

Category weights are region-specific. The weight vector w_c is selected by the market in which the coordinate falls, on the `country_iso` prefix. All four vectors sum to $W = 14$ by construction, so the leading $100/W$ keeps them comparable: a regional re-weighting changes which categories carry the score, not the scale it is reported on.

Category	US/CA/AU/NZ	Europe	East Asia	Rationale for the regional split
grocery	2	3	3	Lower where grocery trips are less frequent
restaurants	3	2	3	
shopping	2	1	1	
coffee	2	2	1	
parks	1	2	1	
schools	1	1	1	
health	2	1	2	
entertainment	1	2	2	
Σ	**14**	**14**	**14**	

East Asia applies to Japan, Korea, and Singapore; *Europe* to the 31 European ISOs. A market matching none of these falls back to *Universal*.

Distance decay. The walk decay is a smoothstep (Hermite) cubic with a linear bleed, defined on $x = (d - 400)/2014$:

$$f_{\text{walk}}(d) = \begin{cases} 1 & d \le 400\,\text{m} \\ 1 - 0.88\,(3x^2 - 2x^3) - 0.12\,x & 400 < d < 2414\,\text{m} \\ 0 & d \ge 2414\,\text{m} \end{cases} \tag{3}$$

The coefficients 0.88 and 0.12 sum to unity by construction, so $f_{\text{walk}}(2414) = 0$ exactly rather than by truncation. Reference values: $f(400) = 1.00$, $f(1000) \approx 0.78$, $f(1609) \approx 0.36$, $f(2000) \approx 0.12$. Each piece of the shape carries a reason. The plateau grants full credit to anything within a ∼5-minute stroll: below 400 m, differences in distance do not change behaviour, so they should not change the score. The smoothstep gives a continuous, kink-free falloff; a POI at 401 m scores essentially the same as one at 399 m, so small geocoding errors cannot move the score. And the linear bleed exists because a pure smoothstep is *flat* at both ends: without the $0.12\,x$ term the far half of the catchment would barely discriminate, whereas with it a nonzero gradient runs all the way to the cap. Full score therefore holds to 400 m and decays to zero at 2414 m (≈ 1.5 miles), the amenity-reach catchment. Drive and public transport reuse the functional form on different support: a 180 s plateau and a 900 s zero for driving, 300 s and 1800 s for public transport: the three modes differ in their budget, not in their decay shape.

Two properties worth stating plainly. Foremost, A_m is *not* capped at 100 in either production path: a cell with many categories simultaneously near-saturated can exceed it, and a reader should not assume $A_m \in [0, 100]$. Separately, $b(n)$ returns 0.7 rather than 0 for a null count. That this is harmless at all rests, however, on $f_m(\text{null}) = 0$ zeroing the product: harmless by composition, not by construction. The unbounded support is one reason the score is reported through percentile normalisation (Section 2.7): the raw composite is an open-ended index, not a bounded grade, and only its rank within a cohort is interpretable.

2.2 Effective network distance for multi modal graphs (Public Transport)

The effective distance from g to a POI p along a path $\pi = n_0 \to e_1 \to \cdots \to e_k \to n_k$ over the typed multi-edge graph is

$$d_{\text{eff}}(g, p) = \alpha^m_{\text{snap}} \cdot d_{\text{snap}}(g) + \sum_{i=1}^{k} \text{cost}(e_i) + \sum_{i=1}^{k} \pi(n_i) + \alpha^m_{\text{snap}} \cdot d_{\text{snap}}(p). \tag{4}$$

- $d_{\text{snap}}(g)$, $d_{\text{snap}}(p)$: pre-stored perpendicular distances from the cell centroid (resp. the POI) to its nearest network node, in metres. Both off-network legs are treated symmetrically. The origin leg is the term most often omitted by production scoring systems that route from a snap node only.
- α^m_{snap}: mode multiplier, 1.5 for walk and 3.0 for drive, converting a perpendicular offset into a plausible detour. The public transport legs take the walk multiplier: the approach is on foot regardless of arrival mode.
- $\text{cost}(e_i)$: edge cost in the mode's unit: metres for walk, seconds for drive and public transport. Drive cost is $\text{length}_m/(0.90\,v)$, the 0.90 discounting to a maximum legal speed. Since `maxspeed` coverage is patchy in most countries, most edges resolve to a class default rather than to a real limit, which is inferred from local traffic laws
- Public Transport cost is the *median* observed gap between consecutive stops on a trip, not a single scheduled delta. `wait` edges are a self-loop per stop at min(600, 1800/dep_per_hour), half the headway in seconds, capped at 600 s, so that a stop served once a day is not charged a twelve-hour wait. `transfer` edges add a flat 180 s between any two stops within 200 m. We note that this last is a purely geometric clique: `parent_station` is never consulted, so "transfer" here means proximity and is thus an estimation.
- $\pi(n_i)$: the node's `intersection_penalty` in seconds, computed or inferred by junction tags or simply existence of intersection: traffic signals 20 s, stop 5 s, give-way 2 s, roundabout entry 5 s. It is applied to *drive* junctions only; walk routing is in metres and carries no node penalty. The sum in the equation is notional: the penalty is baked into edge weights at graph-build time, half onto each entering edge and half onto each leaving edge, so that a traversal $A \to B \to C$ accumulates $\pi(B)/2 + \pi(B)/2 = \pi(B)$ without the search having to look at nodes at all.

Destination-leg caps are tighter for driving than for walking, 100 m against 250 m, and the asymmetry follows from how each journey ends. A pedestrian can plausibly cover 250 m of unrouted ground; footpaths through parks, plazas, and block interiors are exactly what the routable network under-represents. A car cannot: a drive ends where the drivable network ends, and a destination more than 100 m from any drivable edge is not reached *by car*; the last leg is a walk, and crediting it to drive access would double-count what the walk score already measures. Public transport takes the walk cap, since its final approach is on foot. Search caps are 2414 m for walk, 900 s for drive, and 1800 s for public transport.

Snap legs as scalar offsets. The shortest-path search runs over the static on-network graph; both off-network legs are applied arithmetically around it. The origin leg is deducted from the budget *before* the search ($\text{cap}_{\text{eff}} = \text{cap} - \alpha_{\text{snap}} \cdot d_{\text{snap}}(g)$), which preserves early termination and the destination leg is added to the returned network distance before the cap test: an offset applied around the search, not inside it.

2.3 Pedestrian score: two branches

The pedestrian composite has two forms. Which one runs is selected by `country_iso` prefix.

Multi-dimensional markets (the 31 European countries, Japan, Korea and Singapore):

$$PR_{\text{raw}}(g) = 0.60\, A_{\text{walk}}(g) + 0.25\, I(g) + 0.15\, R(g). \tag{5}$$

Amenity-modifier markets (United States, Canada, Australia, New Zealand):

$$PR_{\text{raw}}(g) = A_{\text{walk}}(g) \cdot \text{clip}(1 + 0.10\,(z(\rho_{\text{int},500}) - z(\ell_{\text{block}}))\,,\ 0.90,\ 1.10) \tag{6}$$

where $z(\cdot)$ is standardised per market build unit (the loaded `country_iso`, a state for the United States or a province for Canada) over all cells with the feature present; the fitted scope is stored under the label `country_urban`, but no settlement-class filter is applied. Both evaluation paths consume the same fitted statistics: the offline recomposition fits and applies them in SQL, and the live path loads the same fitted rows per request (unified July 2026; live responses served by builds predating that unification left the statistics

unloaded, making the live modifier identically 1.0). The modifier is bounded to ±10% and degenerates to exactly 1.0 whenever a feature or its fitted standard deviation is absent or zero, in which case $PR_{\text{raw}} = A_{\text{walk}}$. Infrastructure and reachability do not enter the pedestrian score in these markets at all: what ships there is an amenity score carrying a bounded modifier, not a multi-dimensional composite.

The infrastructure term I is a fixed-reference rescale, not a within-cohort percentile:

$$I(g) = 0.40\,\text{conn}(g) + 0.35\,\text{green}(g) + 0.25\,\text{slope}^{-1}(g) \tag{7}$$

with $\text{slope}^{-1} = \max(1 - \min(\text{slope}_{\text{deg}}, 15)/15,\ 0) \cdot 100$.
Connectivity is the same linear rescale on both evaluation paths:

$$\text{conn}(g) = \min(L_{\text{walkable},500}(g)/1500,\ 1) \cdot 100 \tag{8}$$

Greenness blends mapped and observed vegetation. The green term is a 50/50 blend of the mapped green/blue polygon area share within 500 m and satellite NDVI (annual median composite), the latter rescaled linearly to full credit at NDVI = 0.6:

$$\text{green}(g) = 0.5 \cdot \text{green\_blue}_{500} \cdot 100 + 0.5 \cdot \min(\max(\text{NDVI}_{500},\ 0),\ 0.6)\,/0.6 \cdot 100 \tag{9}$$

The two halves are complementary: the mapped half captures designated, accessible green and water; NDVI captures actual vegetation that mapping misses, such as street trees and private greenery. Where NDVI is unavailable the term falls back to the mapped share alone rather than being halved.

The reachability term R normalises the 15-minute walking reachable area against a free-space disc of radius 1275 m:

$$R(g) = \min\left(\frac{A_{\text{iso}}(g)}{\pi \cdot 1275^2},\ 1\right) \cdot 100, \qquad \pi \cdot 1275^2 \approx 5.105\ \text{km}^2 \tag{10}$$

The 1275 m reference is 15 min at 1.417 $\text{m}\,\text{s}^{-1}$ (5.1 $\text{km}\,\text{h}^{-1}$), the project's walking-speed constant. It is not the 2414 m amenity catchment of Section 2.1: two references, two purposes, and worth keeping distinct.

2.4 Drive score

$$DR_{\text{raw}}(g) = 0.80\,A_{\text{drive}}(g) + 0.20\,\rho^{\text{norm}}_{\text{road}}(g), \qquad \rho^{\text{norm}}_{\text{road}} = \min\left(\frac{\rho_{\text{road},2000}}{8000},\ 1\right) \cdot 100 \tag{11}$$

The 8000 m reference within the 2 km disc is an absolute constant in both the live and offline paths. We flag it as a known weakness, not a settled choice: the code itself labels it a placeholder, and the intended normalisation is the percentile of ρ_{road} within scope.

2.5 Public Transport score

$$TR_{\text{raw}}(g) = 0.20\,P(g) + 0.20\,F(g) + 0.10\,D(g) + 0.50\,E(g) \tag{12}$$

Amenity reach via public transport, E, is the dominant term at half the weight. This score is summed with the following elements:

- $P = f_{\text{walk}}(d_{\text{nearest stop}}) \cdot 100$: stop proximity, reusing the walk decay, on network walking distance.
- $F = \min(\ln(1 + \bar{\varphi})/\ln(1 + 60) \cdot 100,\ 100)$: frequency, where $\bar{\varphi}$ is the mode-weighted mean departures per hour over stops within 1 km *network walking* distance (not Euclidean), saturating at 60 dep/h. Mode weights: rail and metro 2.0; ferry and cable 1.5; tram, bus, and other 1.0.
- $D = \min(\ln(1 + \nu)/\ln(1 + 8) \cdot 100,\ 100)$: route diversity over distinct routes within the same 1 km walk, saturating at 8 routes.
- E: amenity reach over the union of all six edge modes, using the public transport decay and the same eight categories and regional weights as Section 2.1.

2.6 Reachable-area construction

The reachable area is a by-product of the network traversal. For each cell the 15-minute walking reachable area is constructed from the walk distance map, which is already in memory once the search returns:

1. Filter to the reached set $\mathcal{R}_T = \{n : d_{\text{network}}(n_g, n) \leq T\}$, where $T = 1275 - \alpha_{\text{snap}} \cdot d_{\text{snap}}(g)$.
2. For each frontier edge (n, m) with $n \in \mathcal{R}_T$ and $d_{\text{network}}(n) + \text{cost}(n, m) > T$, interpolate the point along the edge geometry at which the budget expires; add it to the hull-input set $\mathcal{P}$.
3. $A_{\text{iso}}(g) = \text{Area}(\text{Buffer}_{50}(\text{ConvexHull}(\mathcal{P})))$, the 50 m buffer capturing off-network slack of the front-yard/setback kind.

2.7 Normalisation

Raw composites are mapped to percentile ranks against breakpoint vectors fitted at four cohort scopes. Every cohort is additionally split by a binary urban/rural bucket derived from a satellite-based degree-of-urbanisation grid: urban is code ≥ 21 (urban centre 30 plus the 23/22/21 cluster classes), rural is everything below (the 13/12/11 rural classes and 10 water). The raw code is folded to this bucket at both fit and lookup time; cells with no classification value are excluded from fitting and surface as a data-quality gap at lookup.

Scope	Grouping key
Local	Functional Urban Area (2019 delineation). Fitted only for urban cells, all urban classes pooled into one cohort per FUA
National	(country × urban/rural).
Continental	(pool × urban/rural), pools being NA (US, CA), ANZ (AU, NZ), APAC (JP, KR, SG), and EU as the catch-all.
Global	(urban/rural) across all markets.

The Local scope is not always local. Exactly three conditions collapse it onto National: *(i)* the cell is rural (bucketed against the national rural breakpoints); *(ii)* the cell falls outside any FUA polygon; *(iii)* its FUA holds fewer than 200 urban cells and was never fitted. The scope is a percentile lookup.

Breakpoints are fitted with `percentile_cont` (linear interpolation) and stored as 100-element vectors spanning percentiles 0–99.

One implementation note to keep in mind is that the lookup takes the *highest* index whose breakpoint does not exceed the raw value, which is not tie-aware: in a sparsely-served rural cohort, in which many cells share an identical raw value, the whole tied group resolves to the top of its tie rather than its midpoint. Rural percentiles near 99 should be read in that light.

Sample construction. The calibration grid is 250 m, and the filter applied at grid build is pop > 0. Consequently, upon calculation do we filter out other points with no POIs in a 2.4 km area, by omitting the network traversal. This mechanic, however does not allow us to document exactly how many cells were actively calculated and which one were not. Our estimate is a reduction from 56M to 15M cells. This is however not a measured property of the sample and should not be cited as one.

3 The pipeline

What follows is the built system. We take it in run order and give, for each step, technical details that made it viable computationally and temporally.

3.1 Phases and run order

A market goes from nothing to scored in five phases. Each is a separate entrypoint, and each is resumable which is indispensable with such a complex and distributed calculation. P2.30 numbers under the load rather than beside it, which is why the table below carries six rows against five phases.

Phase	What it does
P0 *partitions*	Assigns every node a `partition_id` from a 150 km tile grid.
P1 *grid*	Generates the 250 m `cell_static` skeleton over the market, keeping cells with pop > 0.
P2 *load*	Seven per-market loaders (Section 3.3), then one global stitch pass.
P2.30 *static fill*	The seven step-3 columns, by two different routes (Section 3.4).
P3 *score*	The per-cell Dijkstra loop across forked workers (Section 3.7).
P4 *recompose*	Composites by SQL, then percentile breakpoints (Section 2.7).

The grid is not, however, part of the load orchestrator, and it must exist first: the loaders assert on its absence rather than create it. P1 is therefore a precondition of P2, not a step inside it.

3.2 Database schema

Four schemas carry the work.

- `calculation.*`: the routable graph. `graph_nodes` holds every node with its `node_type`, `country_iso`, and `intersection_penalty`; `edges_partitioned` holds all six edge modes with `cost_s`, `length_m`, and `partition_id`; `pois` holds the amenity points, each with a per-mode snap node and a `snap_distance_m` that is recorded but not bounded. `transit_stops` carries the frequency side wide rather than long, per-mode columns (`dep_per_hour_{rail,bus,tram,metro,ferry,other}` and the `route_count_*` analogues) rather than a mode key and a value.
- the staging schema: raw polygons. One table, `green_blue`: dissolved park, forest, water, and wetland geometry under a GiST index.
- `static.*`: the per-cell snapshot. `cell_static` is the 250 m skeleton: administrative tags, population, the four pre-stored origin-snap columns, the seven step-3 columns, and `fua_id`. `raster_grid_100m` holds tiled PostGIS rasters. `cell_scores` is the wide 64-column raw-measurement table written by scoring, and it is held apart from the skeleton for a write-pattern reason rather than a modelling one: a re-score can truncate and reinsert without inflicting MVCC bloat on the static columns.
- `normalisation.*`: `breakpoints` (percentile vectors per cohort) and `feature_stats` (the per-scope mean and standard deviation behind the US-branch z-scores).

3.3 Per-market build

The build unit is the sub-region, not the market. That is visible in the registry itself, which holds 230 entries, of which 195 are sub-regions: 56 for the United States, 51 for the United Kingdom, 22 for France, 16 each for Germany and Spain, 13 for Canada, 8 for Japan. Germany can only be loaded per Bundesland. For each unit, seven loaders run in a fixed order.

Loader	Writes
network ingest	`graph_nodes` plus walk and drive edges, from a tag-filtered extract of the street network. A separate pass assigns each drive junction its `intersection_penalty`.
green/blue	the staged `green_blue` polygons, dissolved.
timetable aggregate	`transit_stops` with the per-mode frequency columns, from the busiest Tuesday's 07:00–09:00 peak.
`transit_edges`	Stop nodes, then `connector`, `wait`, `transit`, and `transfer` edges.
`poi_snap`	`pois` with per-mode snap nodes and distances.
`raster_zonals`	`raster_grid_100m`, via `raster2pgsql` at 256×256 tiles.
`cell_origin_snap`	The four `cell_static.snap_*` columns.

Snapping is KNN to a junction. Both the POI snapper and the cell-origin snapper find the nearest existing junction node and store the distance; the network itself is not modified. Snap distance is not capped at load time; every POI gets a row at whatever distance it lands, and the 250 m and 100 m destination caps of Section 2.2 are applied downstream, at scoring, which is what makes them re-tunable without a reload.

Snapping is tiled for performance. Both snappers work through 10 km tiles with a 1 km buffer, each tile queried against a per-market unlogged table carrying its own GiST index. Against a single global index the query planner falls into a pathological plan on dense markets and per-tile time degrades from roughly 50 ms to over five minutes; keeping each query against a small local index preserves the fast plan. Tiling changes only the execution plan, never the result: the nearest junction is the nearest junction either way.

Public Transport edges. Four synthetic edge types are built on top of the stop nodes.

Mode	Construction and cost
`connector`	Stop's walk-snap node → stop node, one direction, at $\max(d_{\text{snap}}, 1)/1.417$ s. Unbounded: the 200 m limit its docstring claims is not in the live predicate.
`wait`	Self-loop per stop, $\min(600,\ 1800/\sum_{\text{modes}} \text{dep\_per\_hour})$ s.
`transit`	Consecutive stops on a trip, cost = the median observed gap, with pairs over two hours apart discarded.
`transfer`	Any two stops within 200 m, flat 180 s, both directions.

The multimodal graph is the pipeline's central innovation. A timetable is time-dependent, and routing over one directly using time-expanded graphs and RAPTOR-style search answers "when should I leave?" at a cost that is prohibitive for ten million origins. The score needs only expected reachability, and expectation admits a static graph: the four edge types collapse the timetable into time-independent weights on top of the pedestrian network. A door-to-door journey is then priced by construction: walk to the stop, wait half the headway on the stop's self-loop, ride at the median inter-stop time, transfer for a flat 180 s, walk on to the destination. Frequency is therefore not a separate sub-score but a cost inside the journey: a ten-minute-headway metro charges 300 s of expected wait, and the 600 s cap keeps a twice-a-day rural bus priced as inconvenient rather than unreachable. This method allows true multi modal travel, by including these nodes and edges into the network.

Cross-border stitching. Loading per market buys parallelism and resumability, and it costs topology: adjacent extracts assign different node identifiers to the same border road, so a road crossing a market boundary would otherwise dead-end at it. One global pass, run after every market is in, adds *stitch edges* between cross-market node pairs sharing a CRS: walk-class within 100 m, drive within 5 m. The stitches are additive rather than node merges, which preserves source attribution and reversibility: a mis-placed stitch edge is at worst an extra option the search takes only when genuinely shorter.

3.4 Static fill: the seven step-3 columns

Seven columns carry the infrastructure inputs, and every one of them is a buffer statistic around the cell: walkable-network length, junction density, block length, green/blue share, road density, NDVI, and slope. They are filled once per market at load time rather than per cell at scoring time, and the reason is measured rather than assumed. Computed live they cost roughly 200 ms per cell, some 40% of total scoring cost and the dominant bottleneck once the Dijkstras were vectorised; precomputed they are a dictionary lookup of about 0.5 ms.

Two routes fill them, and which column goes down which route is the most consequential engineering decision in the loader.

Column	Radius	Route	Definition
`l_walkable_500`	500 m	CTAS	Walk-edge length clipped to the disc
`rho_int_500`	500 m	**conv**	Drive-junction count in the disc
`block_length_m`	500 m	CTAS	Mean length of walk edges lying wholly inside the disc
`green_blue_500`	500 m	**conv**	Green/blue area fraction of the disc
`rho_road_2000`	2 km	**conv**	Drive-network length in the disc
`ndvi_500`	—	CTAS	Point sample at the centroid
`slope_deg`	—	CTAS	Point sample at the centroid

3.5 Buffer statistics as convolutions

Three of the seven columns, junction density, green/blue share, and road density, sum a quantity over a fixed-radius disc centred on every cell of a regular grid. That operation *is* a convolution: of the quantity's density raster with a binary disc kernel. Expressed as a spatial join it is one indexed query per cell, millions of queries per market; expressed as a convolution it is a single FFT pass for the whole market. The pipeline therefore rasterises the source layer at 100 m, convolves once, and samples each cell at its containing pixel:

1. *Rasterise.* Junctions count +1 into their pixel; road segments accumulate their length into their midpoint's pixel; green and blue polygons are rendered as a 0/1 mask, so overlapping parks are not double-counted, and downsampled to an area fraction.
2. *Convolve.* `fftconvolve` against the disc kernel. The area-fraction column divides through by the kernel's sum; the count and length columns need no normalisation.
3. *Sample.* One pixel lookup per cell, then a single bulk `UPDATE`.

The performance gap is not incremental; it separates a load step that finishes from one that does not:

Column	Market	Spatial join	Convolution
Road density	Netherlands	2,707 s	8 s
Junction density	Korea (6.3M cells)	>7 h, unfinished	seconds
Green/blue share	Japan (one sub-region)	>3 h	seconds

3.6 Partitioning and parallelism

A partition is a 150 km tile. Roughly nine partitions cover the Netherlands. Partitions are hard-edged by design, and continuity at the edge is handled in the graph rather than by duplicating geometry: an edge that crosses a boundary belongs to its source node's partition, and the graph loader pulls the partition's own nodes together with the sources and targets of all its edges, so a search that reaches the tile edge continues into the first ring of nodes beyond it.

Workers are forked processes coordinating through Postgres. Eight by default, with no shared state. Each pulls a batch of 1,000 cells with `FOR UPDATE … SKIP LOCKED`, so two workers asking at the same moment get disjoint rows, and a crashed worker's rows are picked up by whoever pulls next. Delivery is therefore at-least-once, and the write absorbs that by upserting. Commits land per batch rather than per cell: per-cell commits cost 3–5× in fsync overhead.

3.7 The per-cell loop

Each cell passes through eight steps in a fixed order. The order is load-bearing rather than incidental: step 4 builds a distance map that steps 4b and 4c read back, which saves two redundant Dijkstras per cell.

Step	Reads	Writes
1 origin snap	the cell row	the four snap fields; no compute
2 nearest stop	walk graph, `transit_stops`	`d_nearest_stop_m`; own Dijkstra at a 1 km cap
3 fixed buffer	the cell row	the seven step-3 columns; a dictionary lookup
4 walk reach	walk graph, `pois`	16 walk columns, and the distance map
4b isochrone	*reuses* the distance map	`A_iso_walk_m2`
4c walkable public transport	*reuses* the distance map	`phi_bar_walk1km`, `nu_walk1km`
5 drive reach	drive graph, `pois`	16 drive columns
6 public transport reach	multimodal graph, `pois`	16 public transport columns

3.8 The routing engine

One engine serves both paths: the offline batch scorer and the live API run the same in-memory routing core, so a stored score and a live score can differ only where the formulas differ (Section 2.3), never because two engines disagree.

The core is deliberately small. When a worker or the API first touches a partition, it pulls that partition's edge list from Postgres once and compiles it into a `scipy` sparse CSR adjacency matrix, one matrix per mode. Walk and drive are ordinary road graphs; the multimodal matrix appends the synthetic wait, transfer, and transit edges to the walk graph, so a single search prices the whole door-to-door journey. Scoring a cell is then one bounded single-source Dijkstra per mode: from the cell's snap node out to the mode's cost cap, returning a distance to every reachable node. Everything downstream: per-category amenity scores, isochrone area, transit-supply terms, is array arithmetic over that distance map. The compiled matrix is cached, so consecutive cells in the same partition pay for the graph load once and nothing thereafter.

Two engineering choices explain most of the engine's speed. First, pgRouting, the conventional PostGIS routing extension, was removed: `pgr_drivingDistance` re-materialises its edge set from the database on every call, which for a three-million-edge partition adds 8–30 s per request; untenable live, wasteful in batch. Second, the CSR is stored with both (i, j) and (j, i) entries and the search is run with `directed=True`: asking `scipy` for an undirected search over an already-symmetric matrix makes it re-symmetrise the matrix internally, which profiling put at 77% of worker self-time. Skipping that work was verified bit-identical over 90 source nodes and made pedestrian searches 2.4× faster, drive 11×, and multimodal 13.5×.

4 Using the score

4.1 Access and decomposition

A coordinate is scored through `GET /score?lat=&lon=`. The response is nested JSON: three raw composites plus twelve percentile ranks (three composites × four scopes). For research use the relevant parameter is `detail=1`, which returns the full sub-component decomposition behind those numbers:

- `amenities`: per category and per mode, the pair $[\, n_{c,m},\ d^{*}_{c,m} \,]$: how many were reachable and how far the nearest one sat. This is the measurement layer of Section 2.1 exposed directly, not a summary of it.

- `infrastructure`: walkable network length, intersection density, block length, green/blue share within 500 m, road density within 2 km, slope. The block is present in multi-dimensional markets only; in the US, Canada, Australia, and New Zealand a `ped_modifier` takes its place in the pedestrian block, per Section 2.3.
- `isochrone_walk_15min_m2`, `transit_supply`, and a `components` block giving each composite's decomposition into its weighted terms.

Two reading rules travel with the response. A `null` reads as *no data*, not zero, no public transport stop within range is not a stop at distance zero. The decomposition is what makes the score a research instrument rather than a black box: any composite traces back to the counts and distances that produced it, and a disagreement with an external methodology localises to a term rather than to the score as a whole.

4.2 Two evaluation paths

The same URL serves two computations. The distinction is material to anyone benchmarking the system.

Path	Behaviour
`live=1` (default)	Snaps the exact coordinate to the nearest walk and drive junction, runs three bounded Dijkstras, and composes. Sub-cell accurate. Several hundred milliseconds to a few seconds per point.
`live=0`	Maps the coordinate to the containing pre-computed 250 m cell and returns its stored composites. Near-immediate, but cell-centroid rather than address accurate.

Live is the default, and it is the intended setting for delivery. A performance figure quoted for "the API" must therefore come from the live path: the pre-computed lookup measures a database read, not the score. One operational caveat rides with any latency claim, however: the first request into a region that has not been queried recently loads that region's network into memory and can take minutes, after which the region is warm.

5 Discussion

We have specified a multi-modal accessibility score. Three commitments carry it: (i) network-realistic routing, with both off-network legs handled symmetrically; (ii) a decoupling of weight-free measurement from weighted composition sharp enough that methodology iteration costs a recomposition rather than a recomputation; and (iii) normalisation at four cohort scopes, of which the cross-continental pool is the one absent from comparable published work.

For an outside researcher the decomposition is likely the most useful property. Because every composite resolves to per-category counts and nearest distances, a disagreement with another methodology can be localised to a term rather than argued at the level of the headline score; the columns that make the score auditable are the columns that make re-tuning cheap. That is one design seen from two directions, not two properties.

References

[1] S. Fina, C. Gerten, B. Pondi, L. D'Arcy, N. O'Reilly, D. Sousa Vale, M. Pereira, S. Zilio. *OS-WALK-EU: An open-source tool to assess health-promoting residential walkability of European city structures*. Journal of Transport & Health, 27:101486, 2022. doi:10.1016/j.jth.2022.101486.

[2] N. Patel, H.-H. Nguyen, J. van de Geest, A. Wagtendonk, M. J. S. Raju, P. Dadvand, K. de Hoogh, M. Cirach, M. Nieuwenhuijsen, T. M. Lam, J. Lakerveld. *A Walk across Europe: Development of a high-resolution walkability index*. Health & Place, 96:103544, 2025. doi:10.1016/j.healthplace.2025.103544.

[3] Walk Score, Inc. *Walk Score Methodology*. `walkscore.com/methodology.shtml`, accessed 2025.