

“Graph Analytics”
UNIMORE course

Air-Aware Walking Routes

Air Quality Analysis and Path Finding on Modena City

04/03/2025
Giorgia Bertacchini matr. 193871



Today I can take it
slow... and maybe even
breathe cleaner air.



The shortest route isn't always the best!

How help people find the best path based on multiple factors, including air pollution?

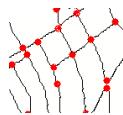
The solution:



Collect real-time sensor data on air pollution.



Create an interpolated pollution map.



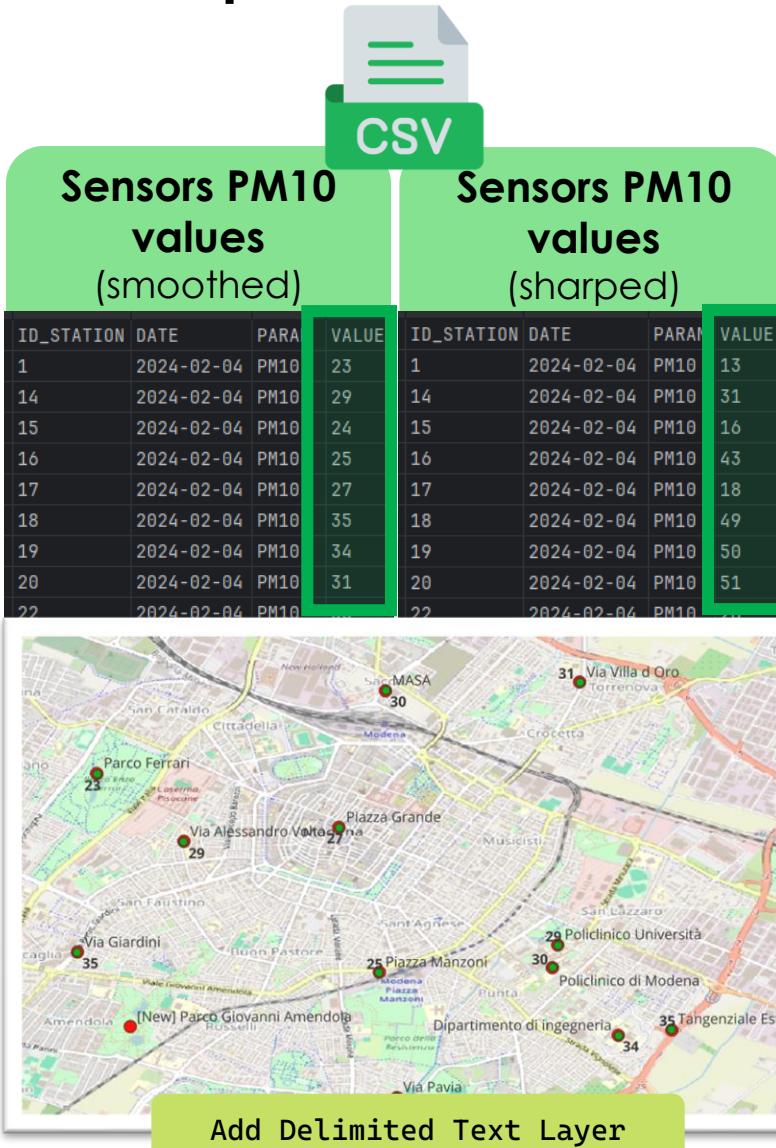
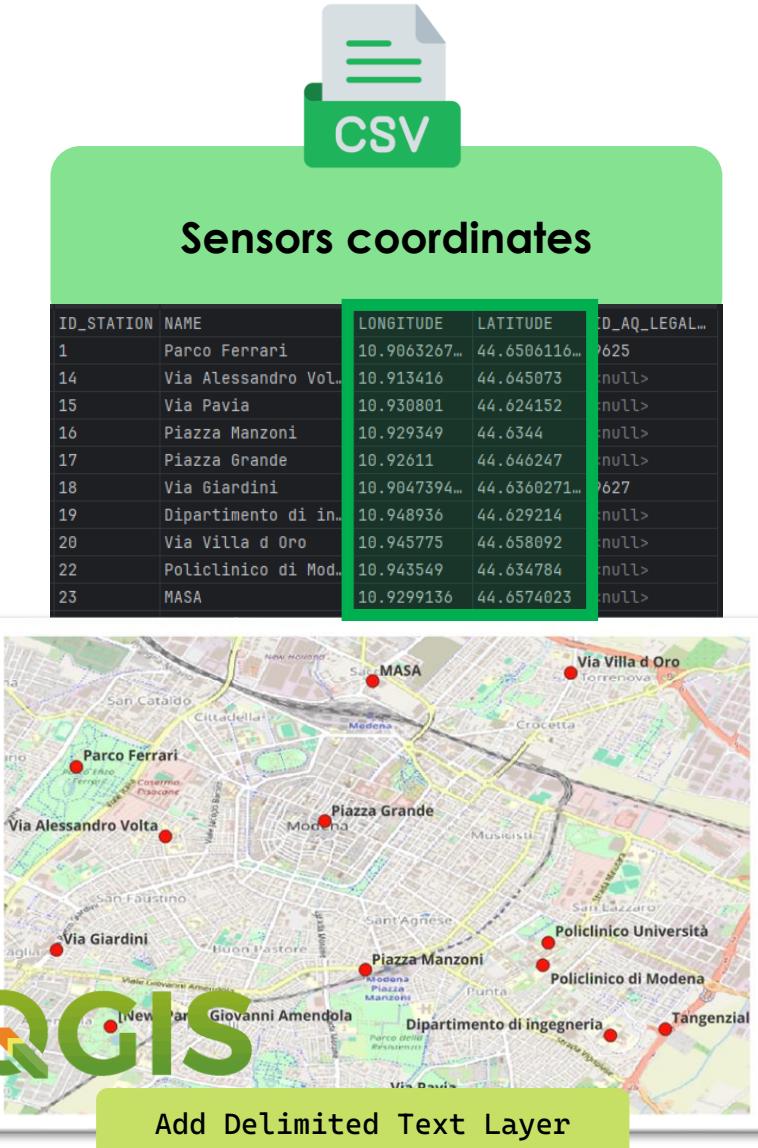
Model the city as a graph, where roads have also the pollution values.



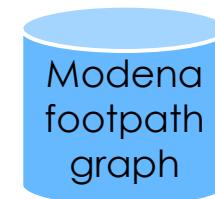
Find the optimal path based on air quality instead of just distance.

A smarter, healthier way
to navigate the city!

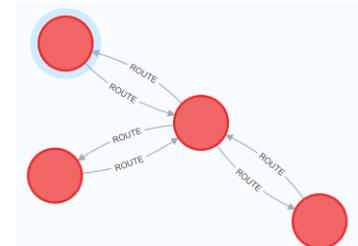
Input Data



Data Analysis of



neo4j



RoadJunction node: **Longitude & Latitude**

```
MATCH (n:RoadJunction) RETURN
min(n.lon) AS min_lon, max(n.lon) AS max_lon,
min(n.lat) AS min_lat, max(n.lat) AS max_lat
```

min_lon	max_lon	min_lat	max_lat
10.8623693	10.9887539	44.6009373	44.6908201

The width and height of the IDW interpolation **raster** are given by the **minimum** and **maximum coordinates** of the Modena road junctions (with 5% extended limits).

ROUTE edge: **Green_area**

It can be seen that many roads do **not have green areas** (equal to 0). This is a variable we can consider when looking for the optimal routes.

```
MATCH (s:RoadJunction)-[r:ROUTE]->(d:RoadJunction) WITH
sum(CASE WHEN r.green_area = 0 THEN 1 ELSE 0 END) AS count_0_green_area,
sum(CASE WHEN r.green_area > 0 THEN 1 ELSE 0 END) AS count_with_green_area
RETURN count_0_green_area, count_with_green_area
```

count_0_green_area	count_with_green_area	r.green_area
77714	8852	0

```
MATCH (s:RoadJunction)-[r:ROUTE]->(d:RoadJunction)
RETURN DISTINCT r.green_area
```



ROUTE edge: **Distance**

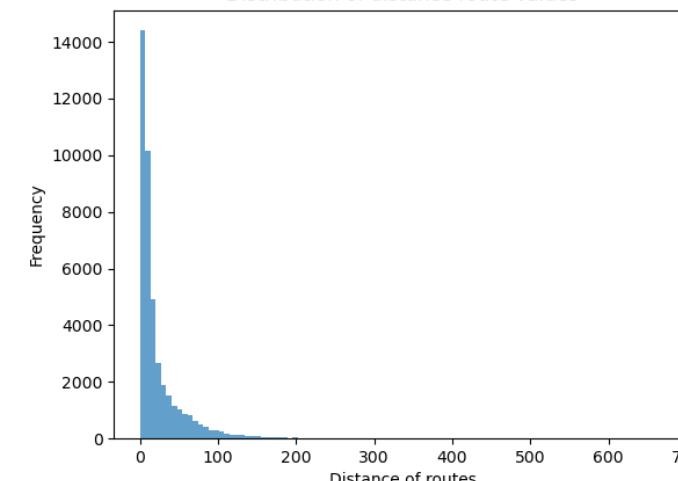
```
MATCH (s:RoadJunction)-[r:ROUTE]->(d:RoadJunction)
RETURN
max(r.distance) AS max_distance, min(r.distance) AS min_distance,
avg(r.distance) AS avg_distance, stdev(r.distance) AS stdev_distance
```

max_distance	min_distance	avg_distance	stdev_distance
672.21881434655	0.08402824390788019	24.710080175546477	37.054355228573726

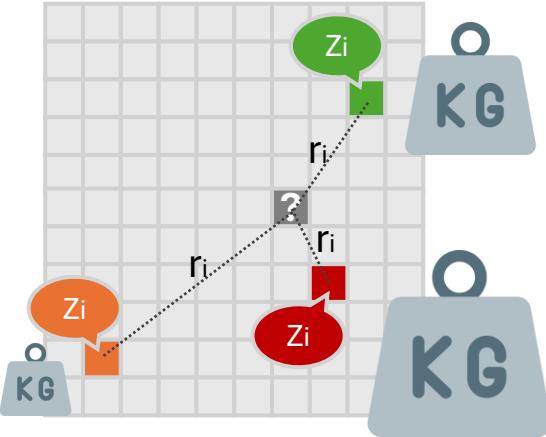
The **range** of distances is quite broad, but most roads have a distance of less than 100 meters.

However, this is a value that should be considered when adding the **PM10 value**, as we want to minimize the time spent on roads with high PM10 levels.

Distribution of distance route values

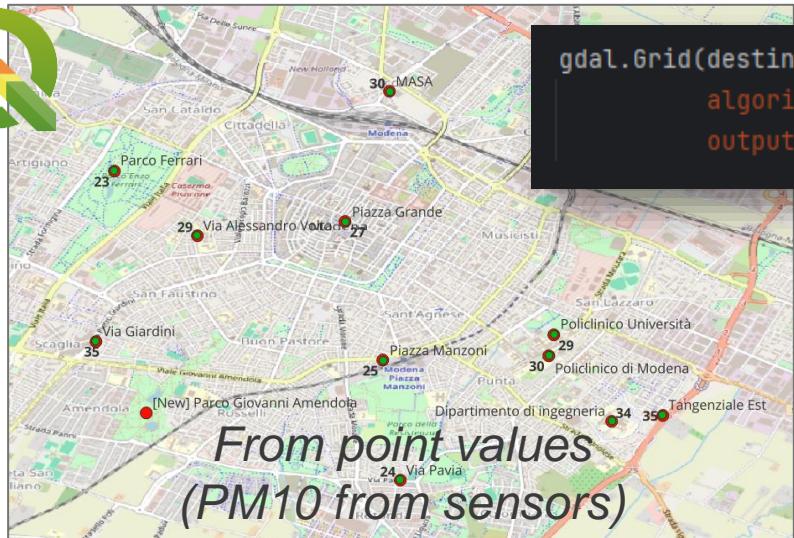
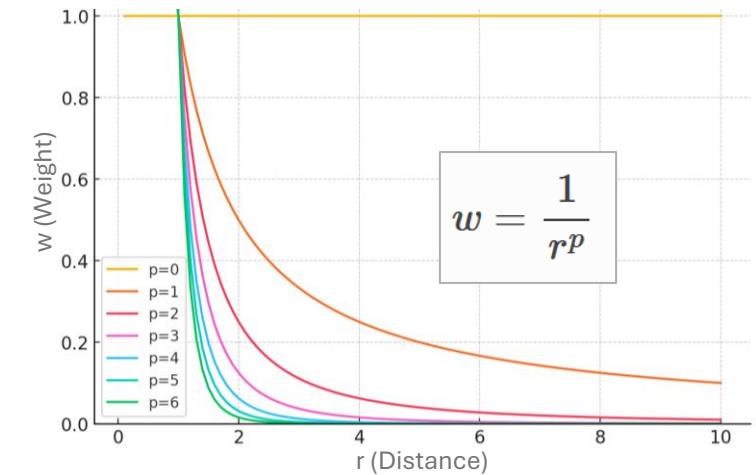


IDW Raster



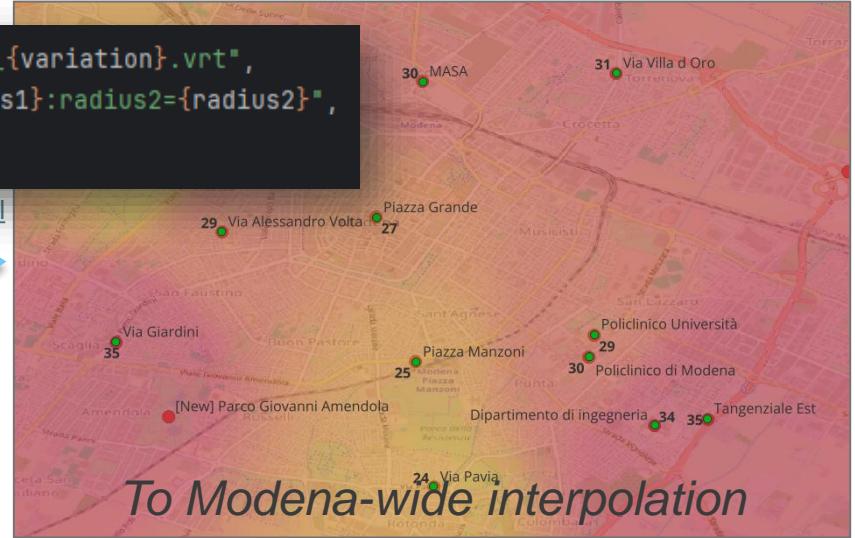
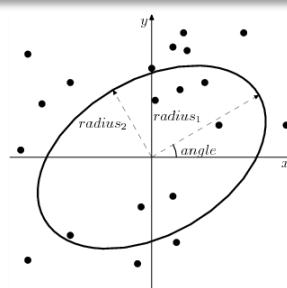
IDW (Inverse Distance Weighting) is a **spatial interpolation** method that estimates a point's value by weighting nearby known points based on **distance**.

$$Z = \frac{\sum_{i=1}^n \frac{Z_i}{r_i^p}}{\sum_{i=1}^n \frac{1}{r_i^p}}$$

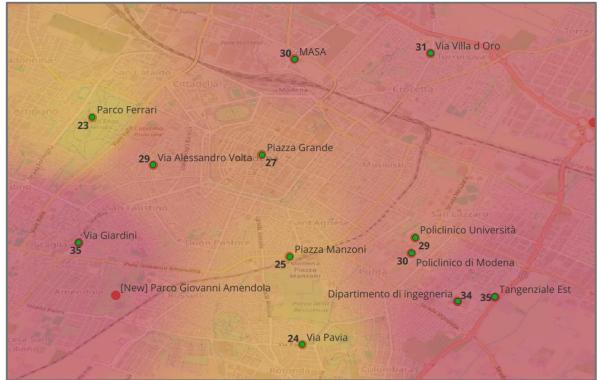


```
gdal.Grid(destination_path, srcDS: f"../output/sensors/meas_{variation}.vrt",
          algorithm=f"invdist:power={power}:radius1={radius1}:radius2={radius2}",
          outputBounds=[x_min, y_min, x_max, y_max])
```

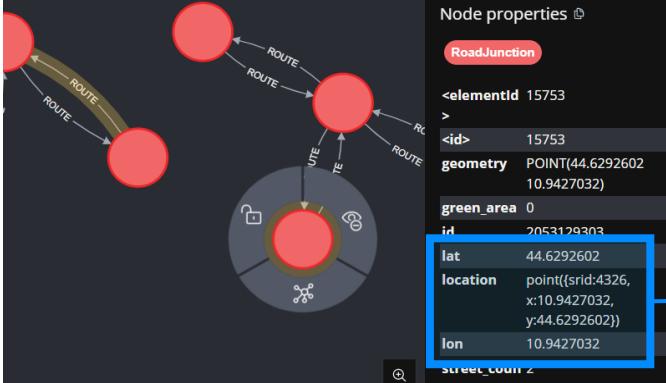
https://gdal.org/en/stable/tutorials/gdal_grid_tut.html



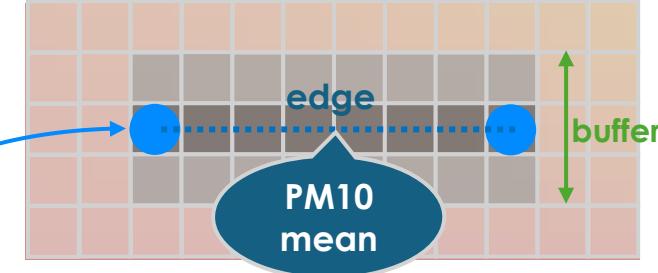
Integration with the Footpath Network



PM10 Raster



Road junction coordinates



To reduce the number of database calls and improve operation time, I used a **single Cypher query** to efficiently load the PM10 value as an edge property in bulk.

UNWIND \$pairs AS pair
MATCH (s:RoadJunction)-[r:ROUTE]->(d:RoadJunction)
WHERE s.id = pair.source AND d.id = pair.destination
SET r.pm10 = pair.mean_air_quality
WITH s, d, pair
MATCH (d)-[r2:ROUTE]->(s)
SET r2.pm10 = pair.mean_air_quality
RETURN pair.mean_air_quality



Roads with PM10 value

Path finding

Relationship weight property

Distance

Simple search for the shortest route.

Distance: is a property of the **ROUTE** edge, measured in meters.



PM10 exposure of route

$$PM10_{route} = PM10 \cdot d$$

PM10: is the average value previously calculated on the raster along the entire edge.

d: for a value of total exposer to PM10 along the route.



Green area of route

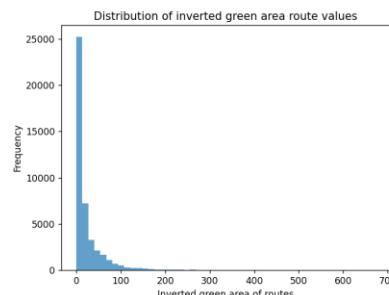
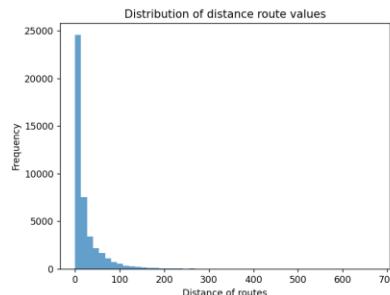
$$GA_{route} = \frac{d}{greenArea / 100} + \epsilon$$

inverse: we need a property that the algorithm minimizes

ϵ is a small value (e.g. 1) to avoid division by zero.

/100: to reduce the impact of the presence of green area.

d: for a value related to the route distance with green area.



Combined weight

$$normPM10_{route\ ij} = \frac{PM10_{route\ ij} - PM10_{route\ min}}{PM10_{route\ max} - PM10_{route\ min}}$$

$$normGA_{route\ ij} = \frac{GA_{route\ ij} - GA_{route\ min}}{GA_{route\ max} - GA_{route\ min}}$$

	PM10 routes	Green area routes
max	~20712.293	~672.219
min	~1.900	~0.084

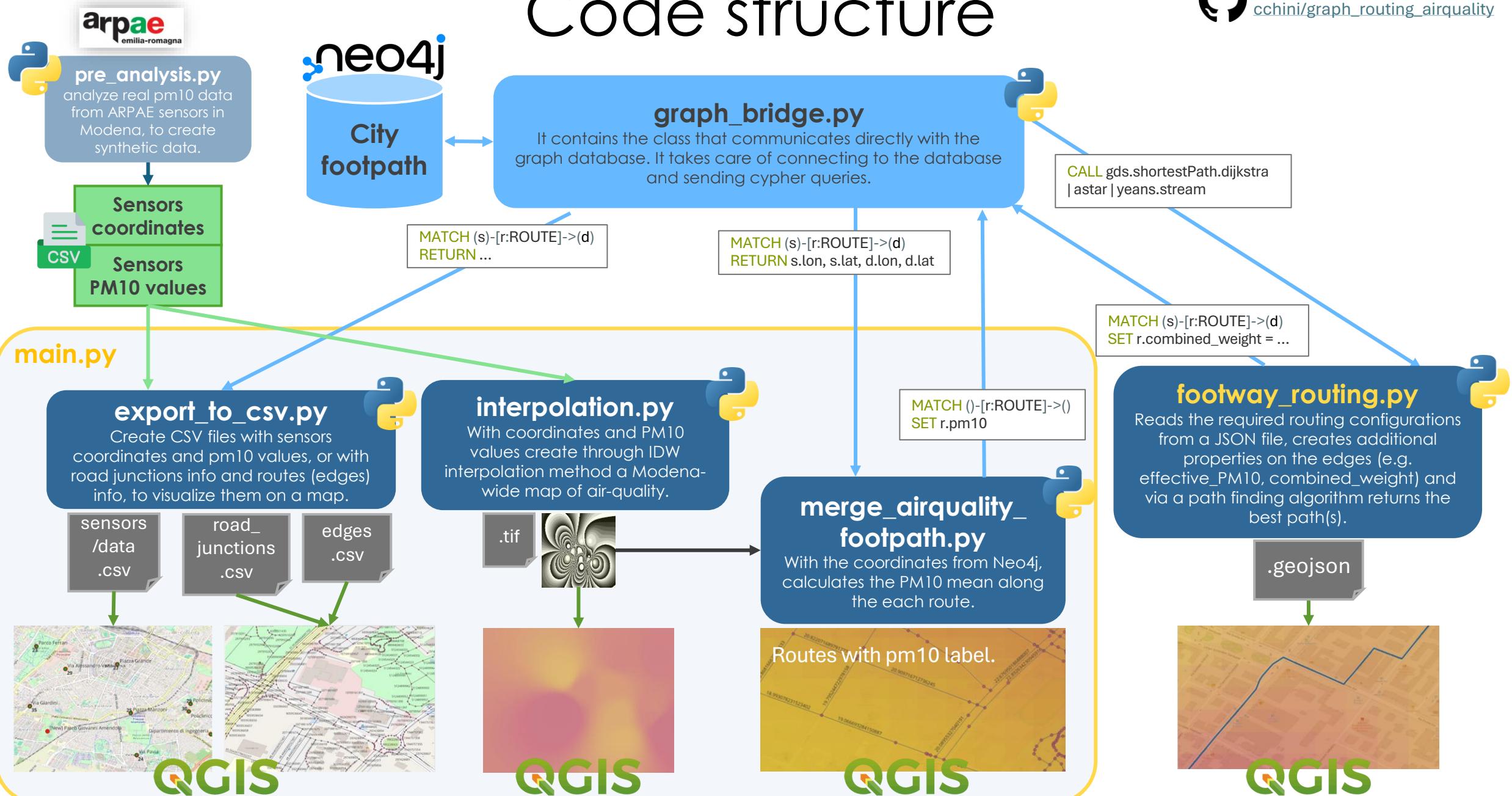
$$weight_{ij} = w_{pm10} \cdot normPM10_{route\ ij} + w_{ga} \cdot normGA_{route\ ij}$$

Normalization: To make the variables comparable, it is essential to account for differences in their **units** and **scales**, ensuring consistency across measurements.

Weights: assigned to each component make it possible to attribute varying **degrees of importance** to them, ensuring a more flexible and detailed evaluation.



Code structure



Path finding algorithms

Dijkstra Source-Target Shortest Path

"supports weighted graphs with positive relationship weights."

A* Shortest Path

"is an informed search algorithm as it uses a heuristic function to guide the graph traversal."

Yen's Shortest Path

"computes a number of shortest paths between two nodes."

```
MATCH (source:Station {name: 'Kings Cross'}), (target:Station {name: 'Kennington (Town)'})
CALL gds.shortestPath.astar.stream('myGraph', {  
    sourceNode: source,  
    targetNode: target,  
    latitudeProperty: 'lat'  
    longitudeProperty: 'lon'  
    relationshipWeightProperty: 'cost'  
})  
YIELD index, sourceNode, targetNode, totalCost, nodeIds, costs, path  
RETURN  
    index,  
    gds.util.asNode(source).name AS sourceNodeName,  
    gds.util.asNode(targetNode).name AS targetNodeName,  
    totalCost,  
    [nodeId IN nodeIds | gds.util.asNode(nodeId).name] AS nodeNames,  
    costs,  
    nodes(path) AS path  
ORDER BY index  
  
MATCH (source:Location {name: 'A'}), (target:Location {name: 'F'})
CALL gds.shortestPath.dijkstra.stream('myGraph', {  
    sourceNode: source,  
    targetNodes: target,  
    relationshipWeightProperty: 'cost'  
})  
YIELD index, sourceNode, targetNode, totalCost, nodeIds, costs, path  
RETURN  
    index,  
    gds.util.asNode(sourceNode).name AS sourceNodeName,  
    gds.util.asNode(targetNode).name AS targetNodeName,  
    totalCost,  
    [nodeId IN nodeIds | gds.util.asNode(nodeId).name] AS nodeNames,  
    costs,  
    nodes(path) AS path  
ORDER BY index
```

<https://neo4j.com/docs/graph-data-science/current/algorithms/pathfinding/>

Example

LEGEND

- Distance (value: total distance)
- Route PM10 (value: total PM route)
- Route Green area (value: total green area)
- Combined weight

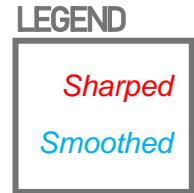


	Shortest Distance	Lowest PM10	Highest Green Area	Balanced (PM10 + Green area)
Distance [m]	~ 9234.748	~ 9810.031	~ 9539.620	~ 9784.344
Avg PM10 [$\mu\text{g}/\text{m}^3$]	~ 33.604	~ 30.052	~ 33.151	~ 30.452
Total green area distance [m]	0	0	~ 1110.718	~ 778.070

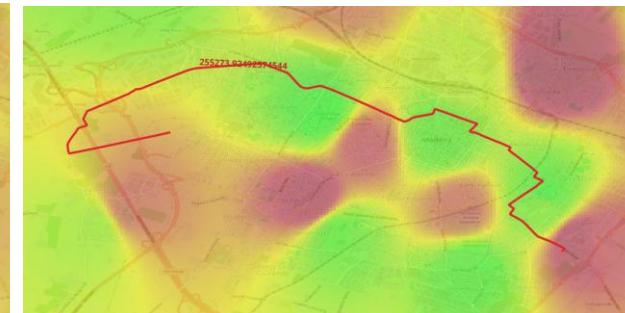


Routing Results

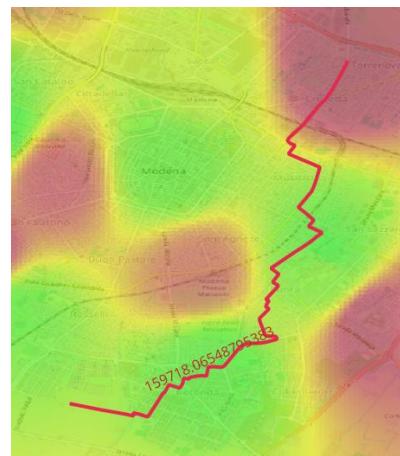
Comparing routes with PM10 values:
'sharped' (deviation standard of 20) and 'smoothed' (deviation standard of 10)



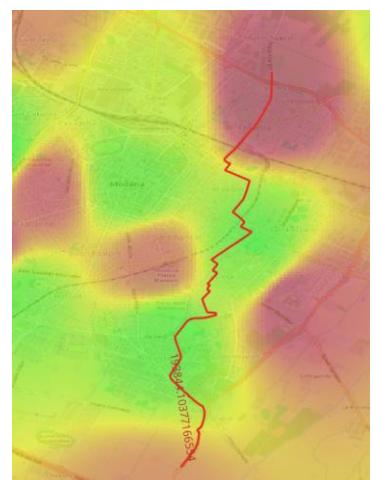
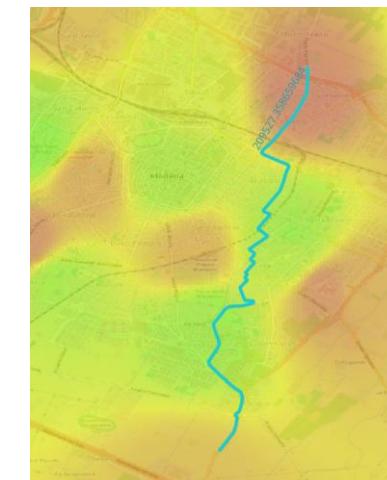
	Total distance [m]	Avg PM10 [$\mu\text{g}/\text{m}$]
Sharped	~11744.035	21.736
Smoothed	~9810.031	30.052



From these tests, we can observe that scenarios with sharper variations in PM10 levels and those with more uniform PM10 distributions can lead to different optimal routes, depending on air pollution patterns.



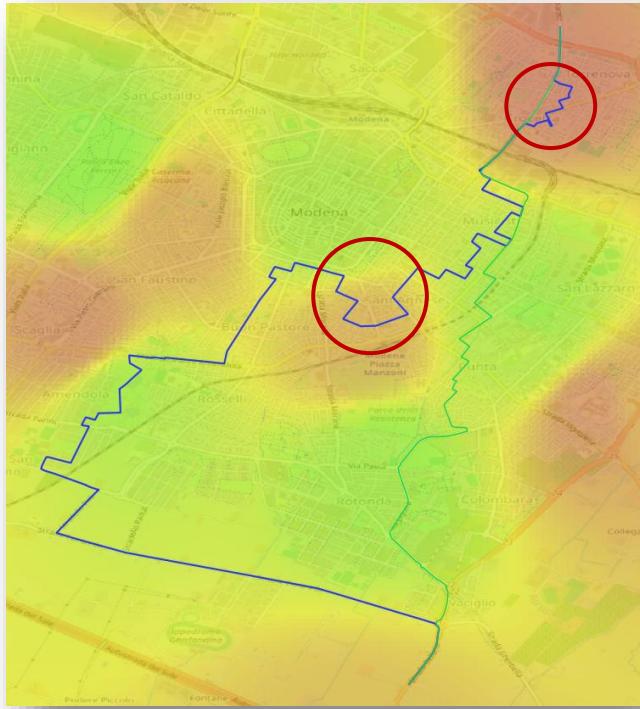
	Total distance [m]	Avg PM10 [$\mu\text{g}/\text{m}$]
Sharped	~6977.433	22.891
Smoothed	~5982.913	28.626



	Total distance [m]	Avg PM10 [$\mu\text{g}/\text{m}$]
Sharped	~7765.125	24.963
Smoothed	~7656.867	27.365

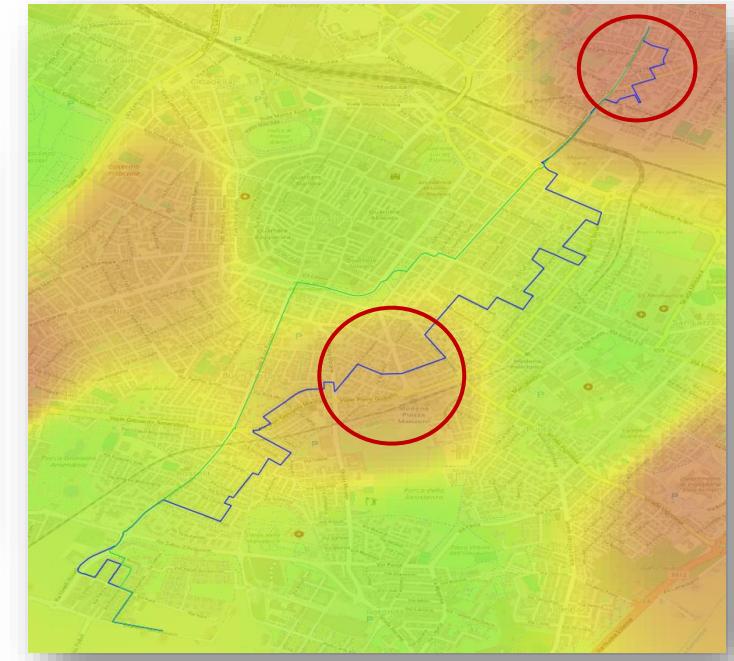
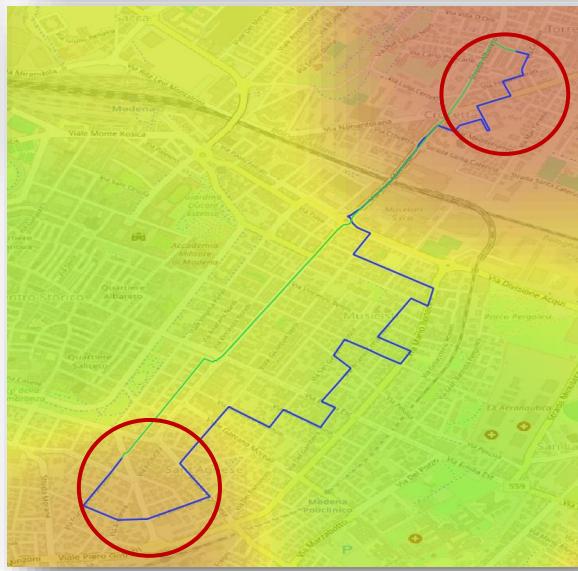
Routing Results

Comparing routes with PM10 multiplied or divided by the distance of the route



LEGEND

$(r.\text{pm10}) * (r.\text{distance})$
 $(r.\text{pm10}) / (r.\text{distance})$



From the tests I conducted, it seems that when the algorithm calculates the optimal route based on the $\text{PM10} \times \text{distance}$ value, the results are more accurate and consistent.

On the other hand, paths calculated based on the $\text{PM10}/\text{distance}$ value seem more irregular, often choosing to unnecessarily pass through more polluted areas.

Routing Results

Comparing paths between Dijkstra and A* algorithm

Dijkstra algorithm

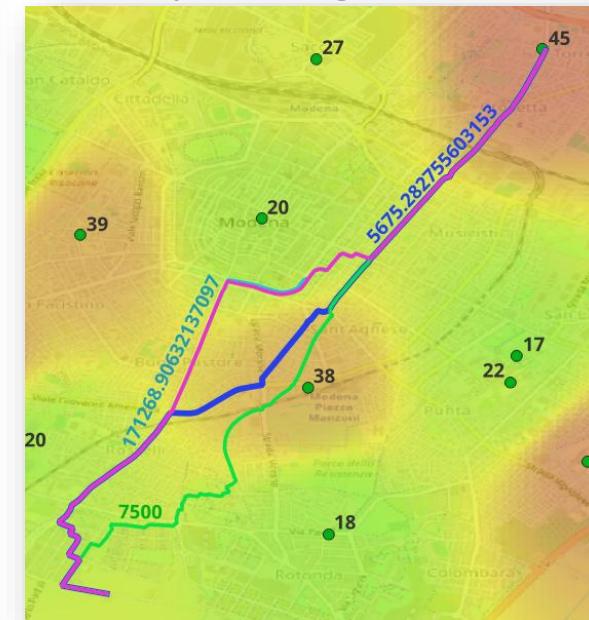


A* algorithm

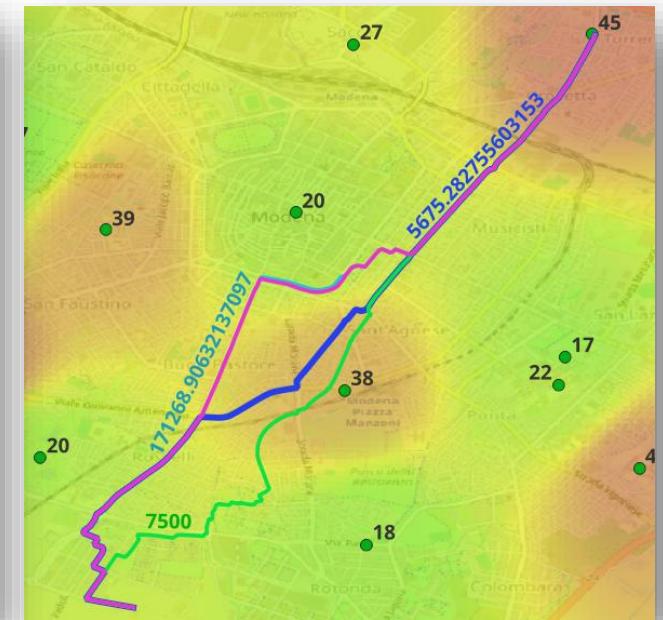


From these examples, we can observe that the results provided by both algorithms — in this case, the calculated optimal paths — are exactly the same.

Dijkstra algorithm



A* algorithm



LEGEND

Distance (value: total distance)

PM10 route (value: total PM route)

Green area route (total green area)

Combined weight

Routing Results

Evaluating whether to divide the combined weight
by the distance

combined_weight = (ratio_pm10 * norm_pm10) + (ratio_greenArea * norm_greenArea)

combined_weight = (ratio_pm10 * norm_pm10) + (ratio_greenArea * norm_greenArea)/distance

Similarly, about the combined weight of the two factors, it appears that dividing by distance results in more random routes that deviate from those suggested by the optimal path based on total PM10 and total green areas.



LEGEND

Distance (value: total distance)

PM10 route (value: total PM route)

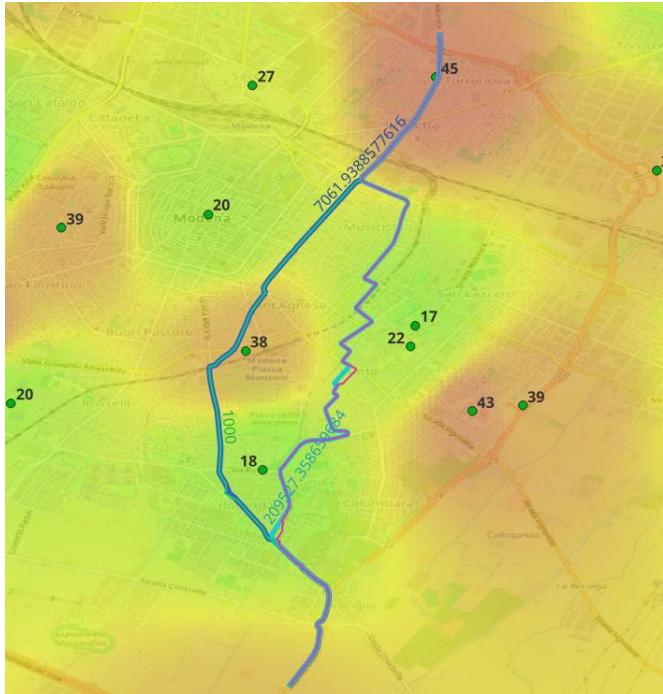
Green area route (total green area)

Combined weight

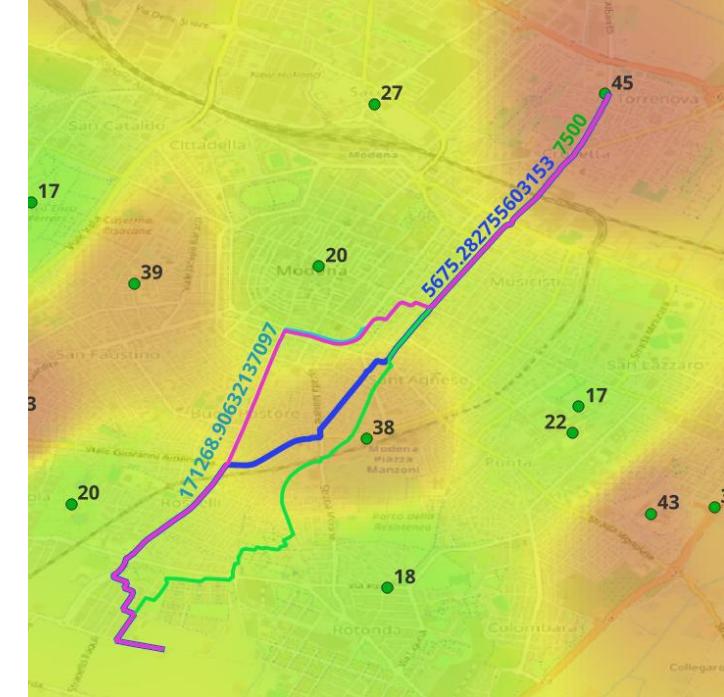
Combined weight

Routing Results

Other complete examples



	Shortest Distance	Lowest PM10	Highest Green Area	Balanced (PM10 + Green area)
Distance [m]	~ 7061.939	~ 7656.867	~ 7085.069	~ 7704.265
Avg PM10 route [$\mu\text{g}/\text{m}$]	~ 30.988	~ 27.365	~ 30.947	~ 27.30,3
Total green area distance [m]	0	0	60.435	60.435



	Shortest Distance	Lowest PM10	Highest Green Area	Balanced (PM10 + Green area)
Distance [m]	~ 5675.283	~ 5982.913	~ 5858.856	~ 5982.164
Avg PM10 route [$\mu\text{g}/\text{m}$]	~ 30.822	~ 28.626	~ 31.061	~ 28.686
Total green area distance [m]	0	346.329	1343.691	751.711

A black and white photograph showing the lower half of a person walking across a crosswalk. The person is wearing dark pants and light-colored sneakers. A long, dark shadow of the person is cast onto the asphalt of the road. The crosswalk consists of several white diagonal stripes.

**Thank You for Your
Attention!**