T-SPARK

v 1.0.0 | June 2025

Transit System Performance Analysis & Redesign Kit

Principal Developer	Reza Mahmoudi, PhD
Supervisor	Prof. Amer Shalaby, PhD
Contributors	Diego Da Silva, PhD
	Shahrin Islam, MSc

T-SPARK is developed at Transit Analytics Lab, University of Toronto

Contents

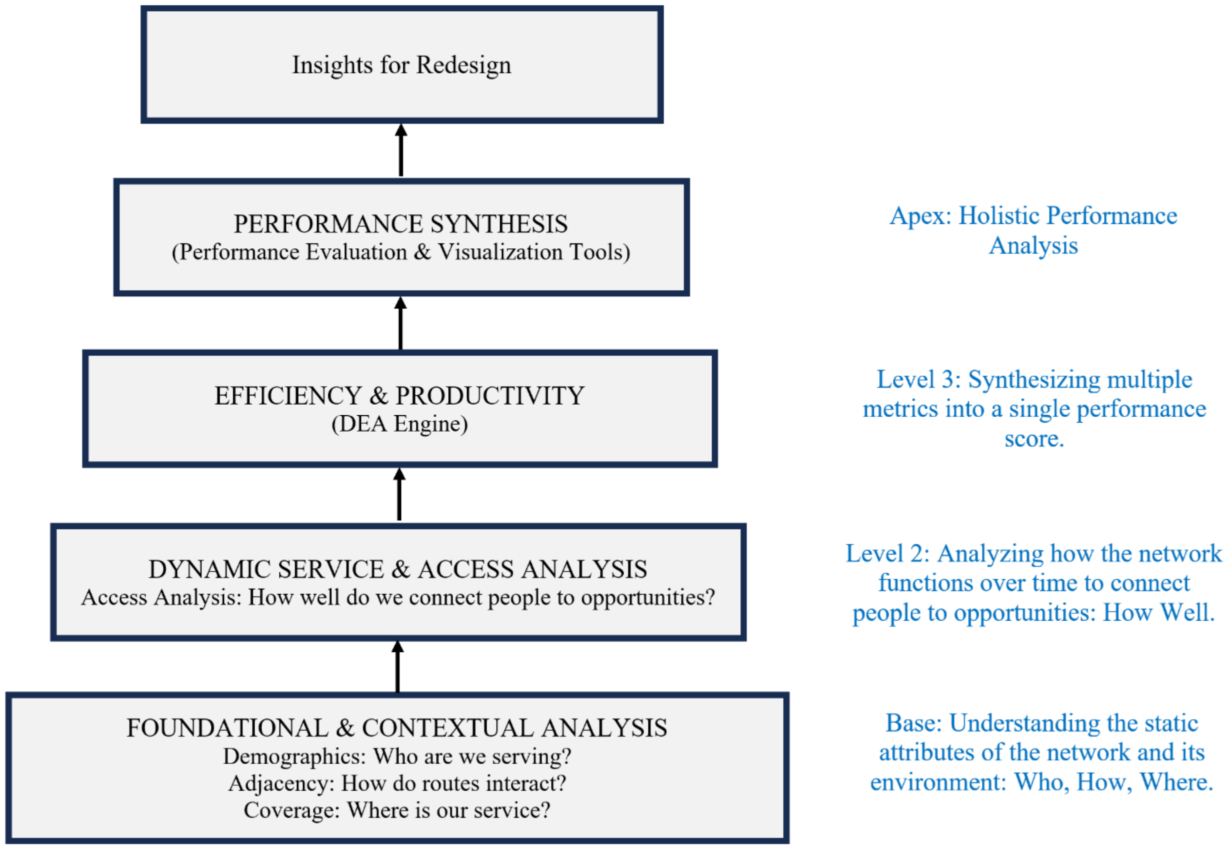
1. Introduction

Welcome to **T-Spark: Transit System Performance Analysis and Redesign Kit**! T-Spark is a powerful, web-based application designed to provide a comprehensive suite of tools for in-depth analysis of surface public transit networks. Whether you are a transit planner, a transportation researcher, or a student, T-Spark offers the tools you need to evaluate, visualize, and gain insights into system performance, coverage, and accessibility of your transit system, almost everything that you need to make informed decisions about the future of an urban transit network.

T-Spark comprises six different tools as follows:

1. **Data Envelopment Analysis (DEA) Engine for Relative Route Performance Evaluation:** Efficiency and productivity analysis using Data Envelopment Analysis, Malmquist Productivity Index, and traditional absolute performance analysis.
2. **Access Analysis:** Analyzing accessibility to different critical services within the city.
3. **Demographic Analysis:** Visualizing socioeconomic attributes for different zones within the city.
4. **Coverage Analysis:** Catchment area and transit desert identification.
5. **Adjacent Network Analysis:** Route overlap and adjacency transit service analysis.
6. **Performance Evaluation:** Spatial display of multi-period efficiency scores, trends, and changes in system performance.

The figure below shows T-Spark's hierarchical workflow, culminating in the alternatives to redesign of an existing transit network to achieve higher performance efficiency.



As illustrated, T-SPARK leverages data-driven techniques to provide redesign alternatives for an existing transit network. Data-driven network analysis is a critical initial step in the transit planning problem (whether at the strategic or operational planning levels) and the associated network design problem: if you don't understand where the problems are, what the potential causes might be and what lessons can be learned from effective route operations, the network redesign problem would not fully informed.

This document will guide you through each of the six tools in the T-Spark toolkit, explaining their purpose, required inputs, and the methodologies behind them (if applicable).

2. Getting Started

Before you begin, please keep the following in mind:

- Data Formats: Each tool requires specific data formats. Please pay close attention to the "Required Inputs" section for each tool. Common formats include GTFS feeds (as a .zip file), ESRI Shapefiles (.shp and related files, often zipped), and Excel files (.xlsx).
- Coordinate Reference System (CRS): For consistency and to ensure accurate mapping, all your spatial data (e.g., shapefiles) should be in the WGS 84 (EPSG:4326) coordinate system. The application will attempt to reproject your data, but it's best to start with the correct CRS.

To run the T-Spark web application, you will need to install a number of Python packages. Here is a breakdown of what each package is used for in your application:

Core Web Framework

- **flask**: The micro-framework that powers the entire web application, handling routing, requests, and rendering templates.

Data Handling & Scientific Computing

- **pandas**: The primary library for data manipulation and analysis. It's used extensively for reading Excel files, handling GTFS data tables, and organizing results.
- **numpy**: A fundamental package for numerical computation. It's a dependency for pandas and geopandas and is used for numerical operations.
- **openpyxl**: A required engine for pandas to read and write modern Excel files (.xlsx).
- **lxml**: An essential library for processing XML and HTML files, used by other libraries for parsing data.

Geospatial Analysis

- **geopandas**: A critical library that extends pandas to allow for spatial operations. It's used for reading shapefiles, performing spatial joins, and managing geographic data.
- **fiona**: The reading and writing engine for geopandas. It provides the link to underlying C libraries (like GDAL) for handling various vector data formats (e.g., shapefiles).
- **shapely**: The library that handles the geometric objects themselves (Points, LineStrings, Polygons) and provides the logic for geometric operations.
- **pyrosm**: Used to read OpenStreetMap data from .pbf files, which is a required input for the Access Analysis tool's network building.
- **r5py**: A key library for the Access Analysis tool. It's a Python wrapper for the R5 (Rapid Realistic Routing) engine, used to build multimodal transport networks and calculate travel times. This package has special requirements.

Mapping & Visualization

- **folium**: The library used to create all the interactive maps in your application. It builds a bridge between Python and the Leaflet.js mapping library.
- **branca**: A dependency of folium that is used to create colormaps and other HTML/JS elements for the maps.

Optimization

- **pulp**: An optimization modeling library used to formulate and solve the linear programming problems for the DEA (CCR, BCC, SBM) models in your DEA Engine.

Transit Network Analysis

- **partridge**: A library for fast and easy reading of GTFS feeds into pandas DataFrames.
- **peartree**: A library used for converting GTFS feeds into graph representations, often used for network analysis.
- **networkx**: A library for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

Critical Prerequisite for r5py:

The r5py library requires a Java Development Kit (JDK), version 21 or newer, to be installed on the system.

You must ensure that Java is available in your system's PATH. You can check this by opening a terminal or command prompt and running java --version. If this command fails, you need to install a JDK (like OpenJDK) and configure your system's environment variables.

For access analyze try to create an environment in T-SPARKS path on your computer and do not forget to active it before doing your analysis:

cd "C:\Path\to\T-SPARK\app.py file" & .\.venv\Scripts\activate

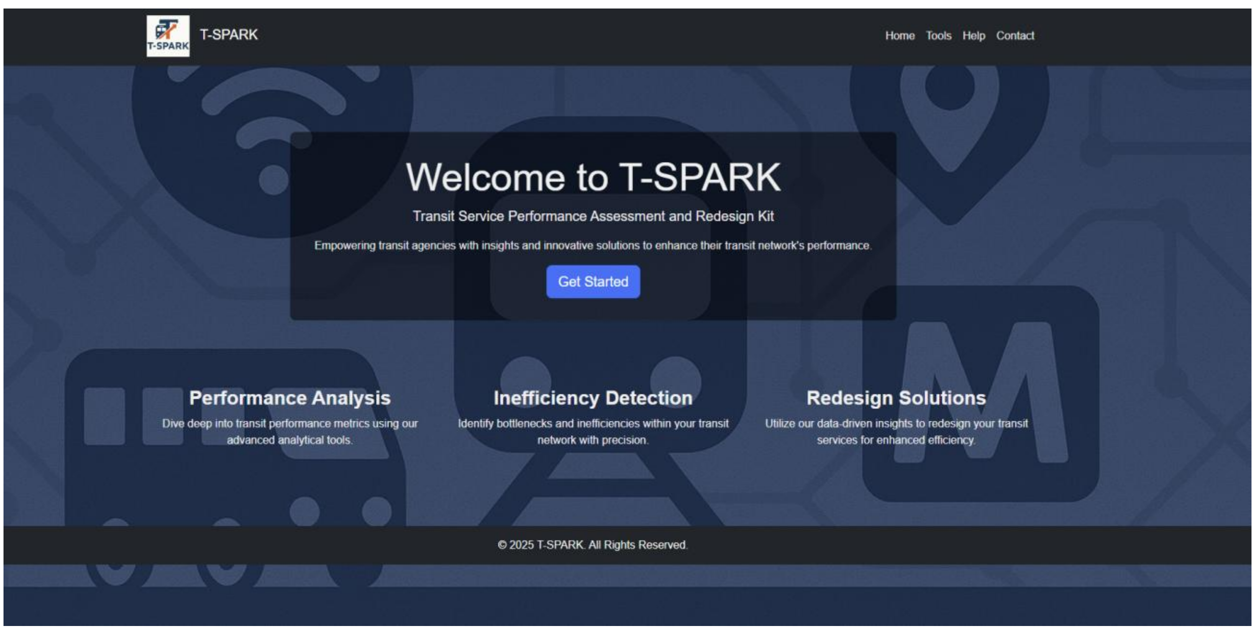
3. Landing Page

When you run the app.py code and click on the provided link you will navigate to the landing page of T-SPARK. In the landing page you will have multiple option, where you can access the tools by clicking on “Get Started” or “Tools” in the options bar. You can always navigate back to the landing page by clicking on “Home”. To access to the guide file, you just need to click on Help, while to see the contact information of the developers you must click on “Contact”.

After running app.py, a local URL will appear in your console. Click that link to open T-SPARK’s landing page. From here, you can:

- **Get Started or Tools**: access to any of the six analysis tools
- **Home**: return to this landing page at any time
- **Help**: open the user guide
- **Contact**: view the developers’ contact information

This menu bar appears on every page, so you can easily navigate between modules and resources.



4. Tools Page

Once you select Tools or Get Started, you will be taken to the Tools page. From there, simply choose one of the available modules to begin your analysis.

5. Access Analysis

Overview: This tool provides a sophisticated analysis of transit accessibility, a key measure of transportation equity and service effectiveness. It moves beyond simple distance-based measures to calculate realistic, schedule-aware travel times from residential zones to essential services like hospitals, schools, grocery stores, and jobs.

Required Inputs:

- **TAZ Shapefile (.zip):** A zipped shapefile of your Traffic Analysis Zones. Must contain a pd column for filtering and a unique ID. Although the current version of T-SPARK is based on TAZ, you can always use it to conduct your analysis in any zonal level as long as you provide the right shape file for your zones.
- **Boundary Shapefile (.zip):** A zipped shapefile of your study area boundary.
- **Service File:** A .geojson or zipped .shp file of your destination locations.
- **GTFS Data (.zip):** A valid GTFS feed for your transit system.
- **OSM Data (.pbf):** An OpenStreetMap data file in .pbf format for the street network.
- **Time Intervals:** At least one defined time interval (start and end date/time) for your analysis.

Methodology Deep Dive: Multimodal Accessibility Analysis

This tool is powered by **r5py**, a Python wrapper for the **R5 (Rapid Realistic Routing on Real-world Multimodal networks)** engine. This state-of-the-art routing engine, developed by Conveyal, is specifically designed for multimodal transport analysis.

Step 1: Building the Multimodal Network

Before any analysis can be done, the tool builds a comprehensive network graph. This is a critical step that combines two data sources:

- **GTFS (Transit Layer):** Provides all information about the public transit system, including stop locations, route paths, schedules, and service frequencies.

- **OpenStreetMap (Street Layer):** Provides a detailed representation of the underlying street and pedestrian network, including sidewalks, paths, and intersections.

R5 intelligently links these two layers. It knows, for example, how to get from a TAZ centroid to the nearest street, walk along the street network to a transit stop, wait for the bus, travel on the bus, and then walk from the destination stop to the final service location. This integrated approach ensures that the calculated travel times are realistic and account for all segments of a transit journey.

Step 2: Calculating Accessibility

The tool uses two distinct and well-established methods for measuring accessibility, depending on the chosen service type.

Method 1: Average Travel Time to the Nearest Facility (Used for Hospitals, Schools, Grocery) For services where a person typically only needs to reach one location (e.g., the closest hospital), the tool calculates the average travel time to the *nearest* facility. This is a robust, two-stage process:

1. **Identify the Nearest Facility:** The tool first runs a preliminary travel time calculation from every origin TAZ to *every* destination (e.g., all hospitals) at a single point in time (the start of your specified time window). For each TAZ, it identifies the single destination with the lowest travel time. This establishes the most likely or "nearest" destination for each origin zone.
2. **Calculate Average Travel Time:** Now that the most relevant origin-destination pair is known for each TAZ, the tool performs a more detailed analysis. It repeatedly calculates the travel time for these specific pairs at regular intervals (e.g., every 5 minutes – you can change this number in the backend – smaller number will lead to longer computation time) throughout your entire specified time window (e.g., 7 AM to 9 AM). By averaging the results of these repeated calculations, it produces a

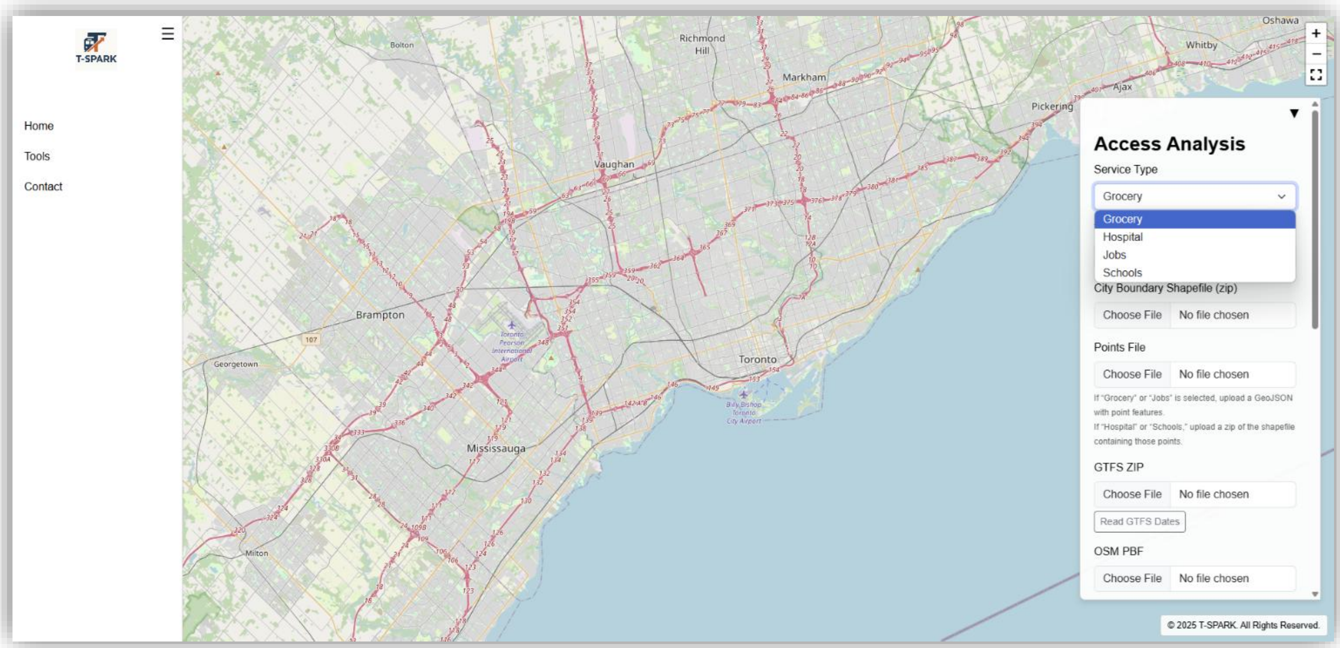
stable and reliable average travel time that accounts for variations in transit service and wait times during that period.

Method 2: Cumulative Opportunities (Used for Jobs) For destinations like jobs, a person's access is better measured by the *number* of opportunities they can reach, not just the time to the single closest one. For this, the tool uses the "cumulative opportunities" method.

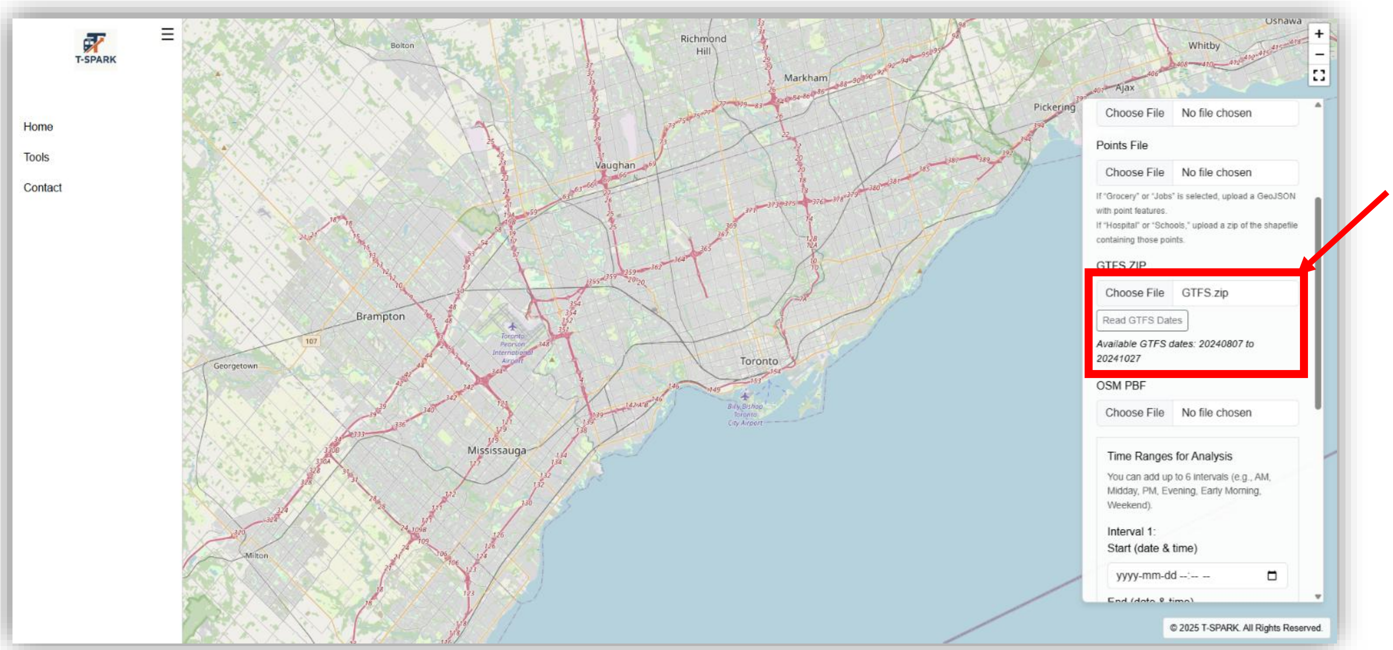
1. **Define a Threshold:** The analysis is based on a fixed travel time budget, which is set to **45 minutes** in the application. You can check this threshold in backend to do sensitivity analysis or match it with your standards.
2. **Count Reachable Jobs:** The tool calculates travel times from each origin TAZ to all possible job destinations (represented by TAZs, weighted by their business count). It then sums the total number of jobs located in all destination TAZs that can be reached *within* the 45-minute threshold.
3. **Average Over Time Window:** This counting process is repeated at every minute throughout your specified time window. The final result is the *average number of jobs* that can be reached from each TAZ within 45 minutes, providing a powerful measure of economic access that accounts for transit schedule variations.

Step-by-Step Guide

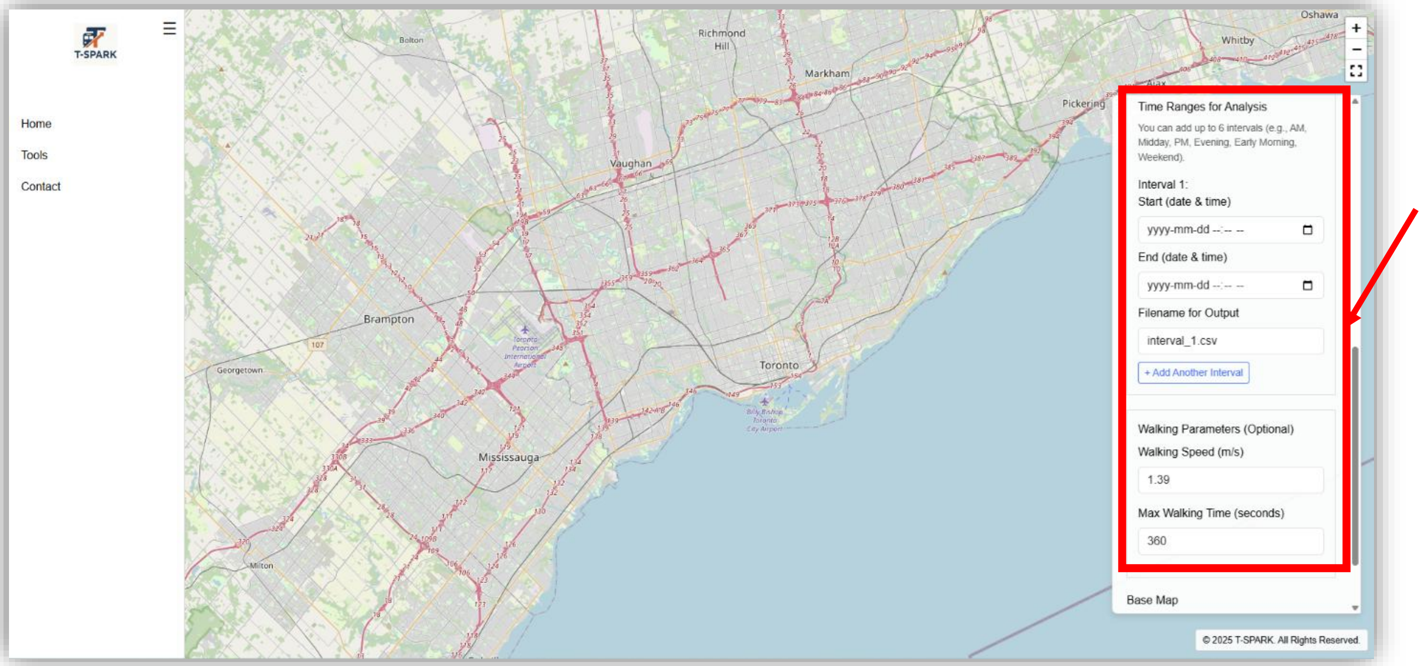
1. Navigate to the "Access Analysis" tool.
2. Choose the specific service type for which you want to perform the accessibility analysis.



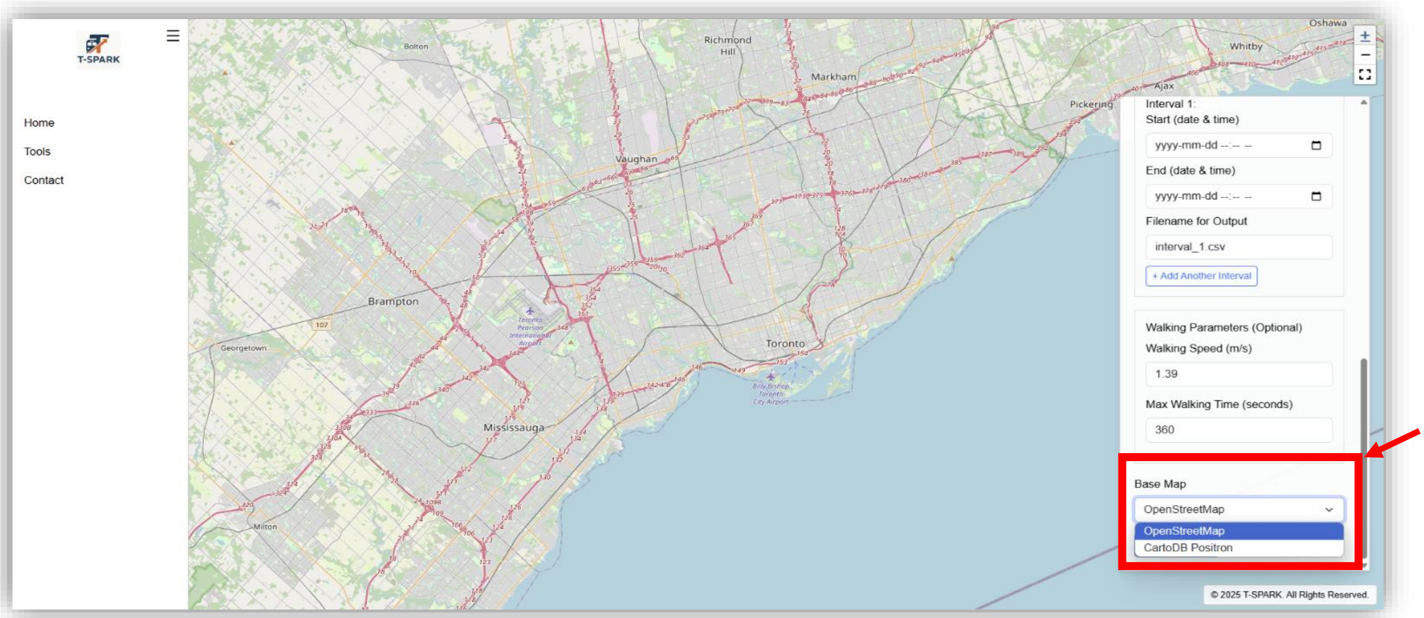
3. Upload the required files. Upload all necessary files for your analysis. You will then be prompted to specify a time range. If you're unsure which dates your GTFS feed covers, click "Read GTFS Dates" to display the exact date range included in your dataset.



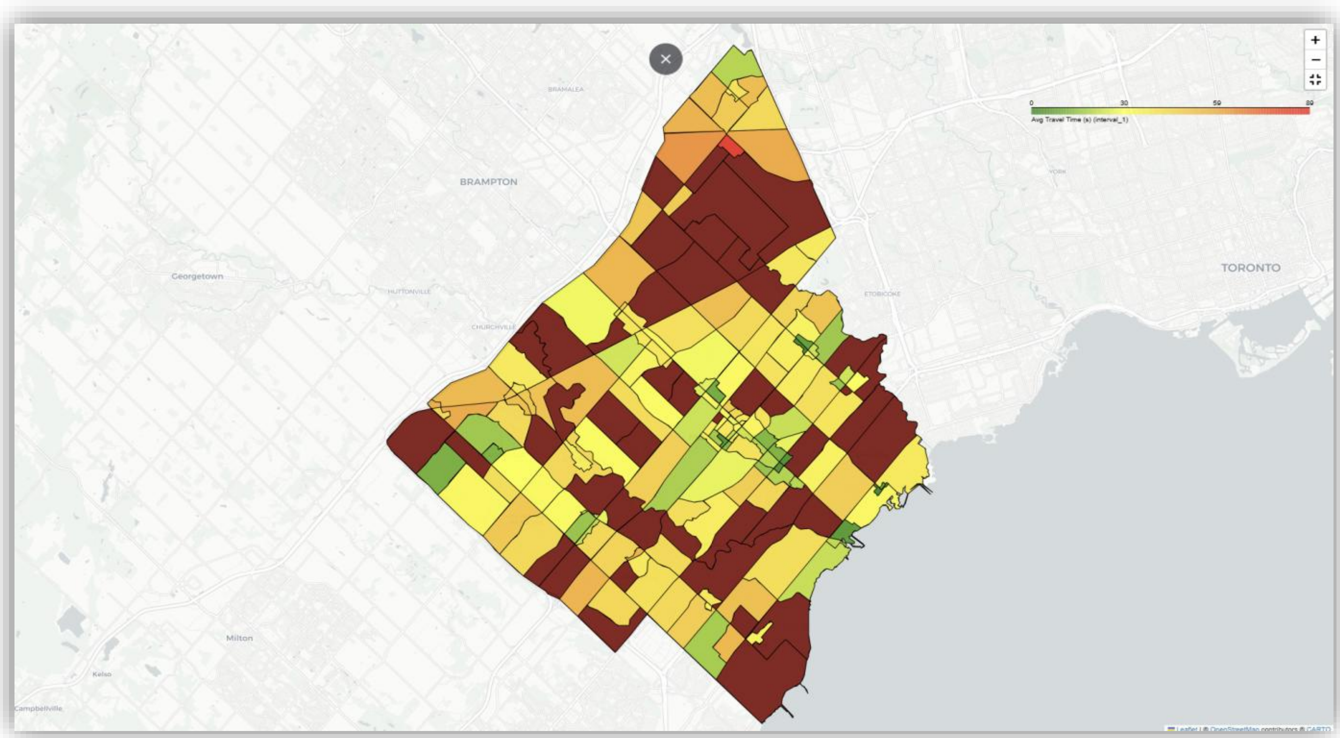
4. Specify the time range for your analysis and walking parameters.



5. You will always have two options for your base map. Select one and then "Run Analysis".



Here is a sample output for this analysis for grocery access within City of Mississauga based on user inserted parameters:



You can see details for each zone just by clicking on it.

6. Demographic Analysis

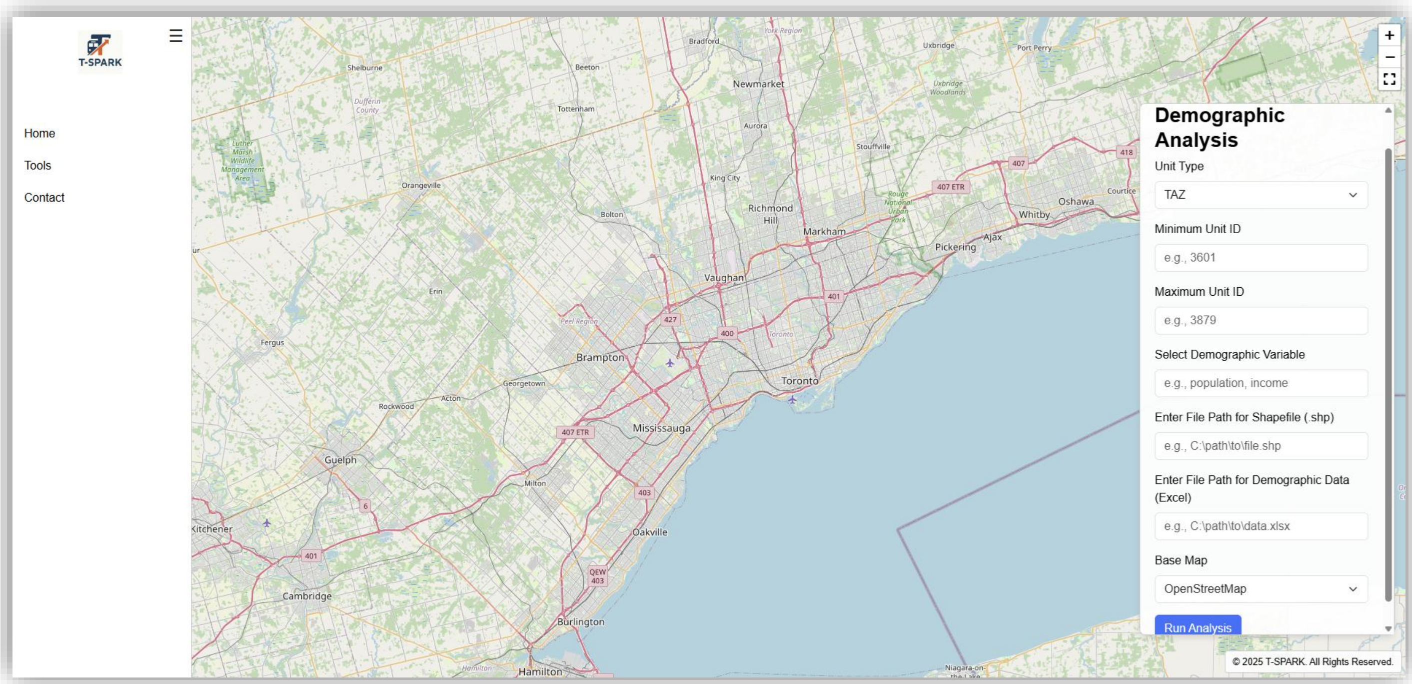
Overview: The Demographic Analysis tool allows you to visualize demographic or land-use data on a map. You can map data from your zone shapefile or merge it with an external Excel file to create rich, thematic maps.

Required Inputs:

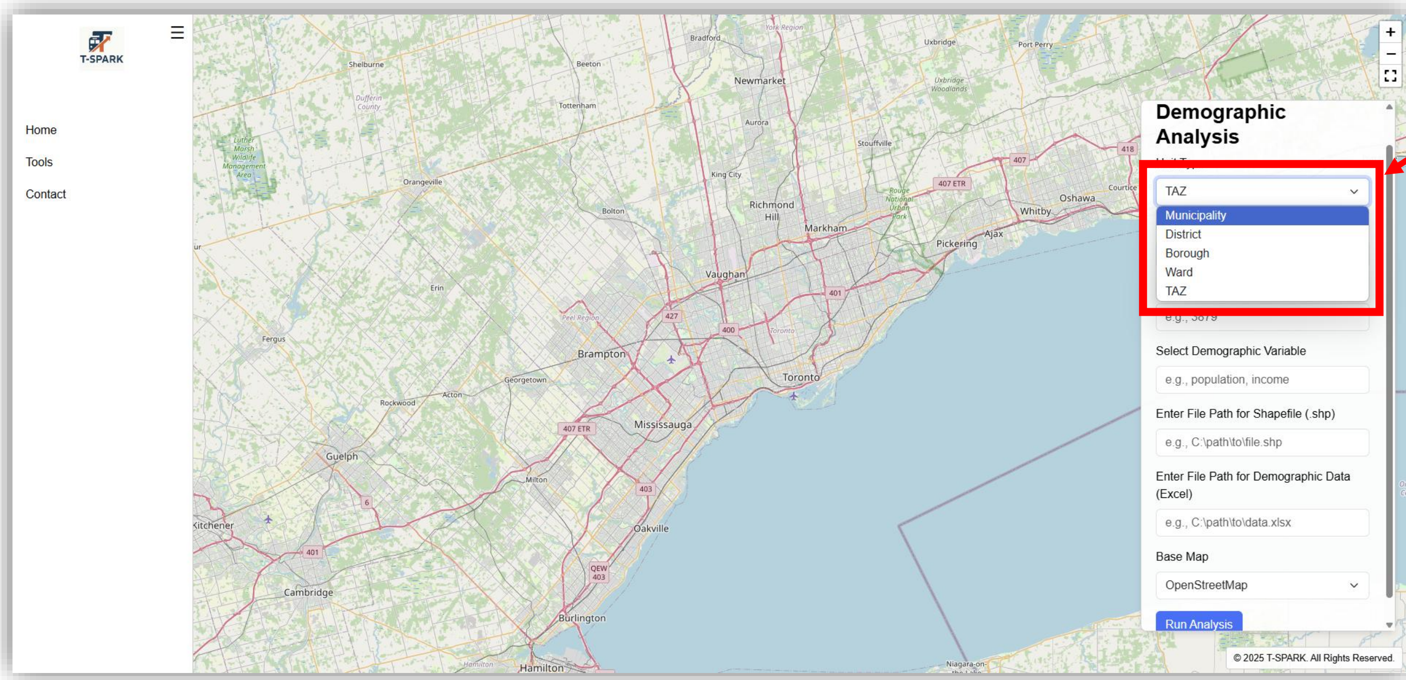
- **Shapefile (.shp):** An ESRI Shapefile of your analysis zones. **Crucially, it must contain an integer column named gta06** which will be used as the unique ID for each zone. You have different format. Still T-SPARK can handle it with minor modification in the backend based on your ID name. Change **gta06** to your label.
- **Excel File (.xlsx) (Optional):** An Excel file containing demographic data. It must have a gta06 column to be merged with the shapefile. The other columns should be your demographic criteria (e.g., Population, Median_Income).

Step-by-Step Guide:

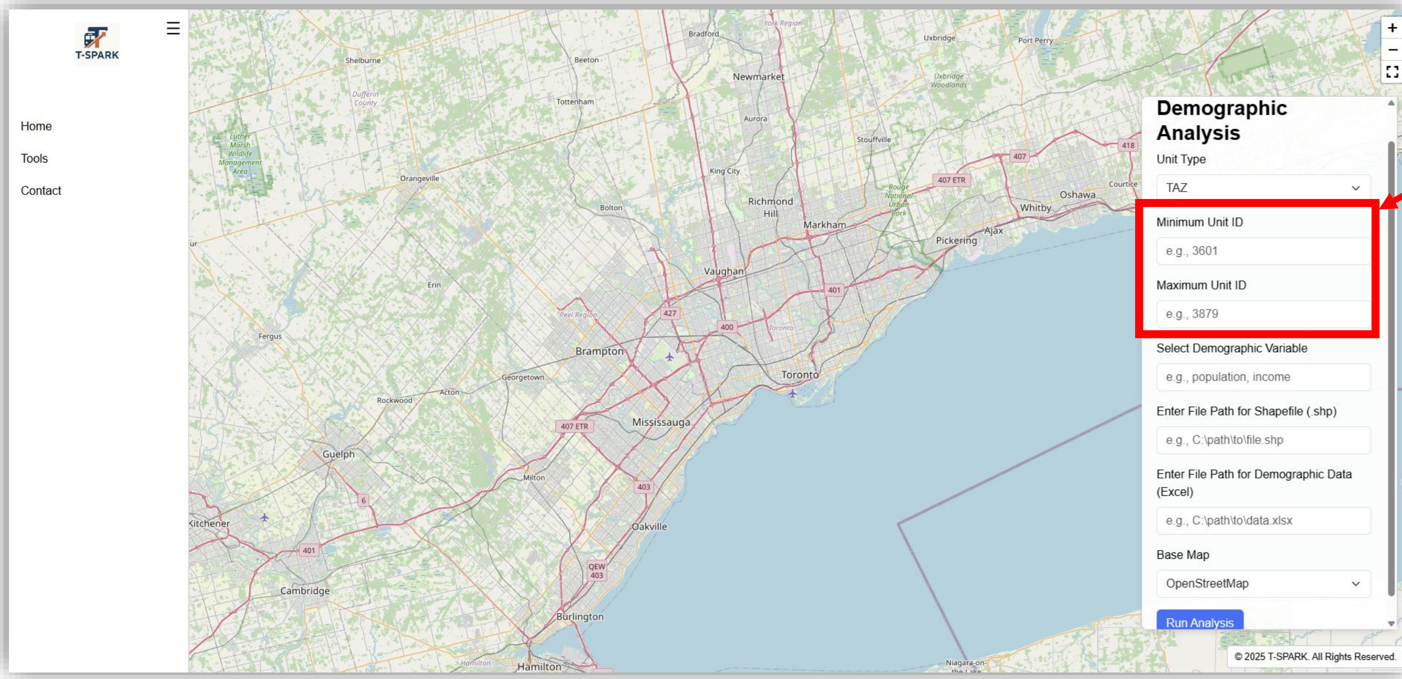
1. Navigate to the "Demographic" tool.



2. Select the zone level that you want to conduct your analysis.

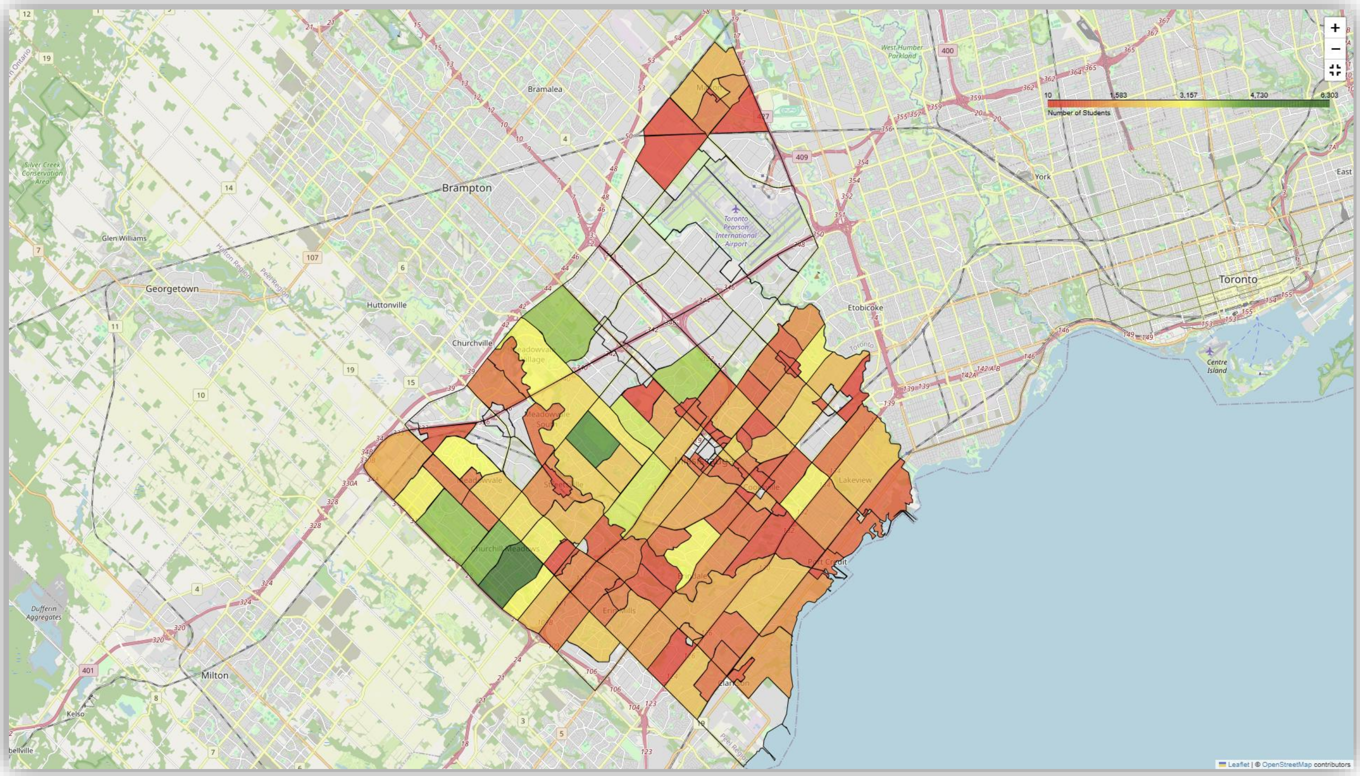


3. Filter the zones to be displayed by providing a min and max ID.



4. Provide the path to your shapefile.
5. Provide the path to your Excel file.
6. Select the Criterion you wish to map.
7. Choose a base map and click "Run Analysis".

d. Understanding the Outputs The tool produces an interactive map. If you select a criterion, the zones will be color-coded based on their value for that criterion, with a legend provided. You can hover over a zone to see its ID and data value. Below is a sample output illustrating the “Number of Students” analysis across various Traffic Analysis Zones (TAZs) in the City of Mississauga.



8. Coverage Analysis

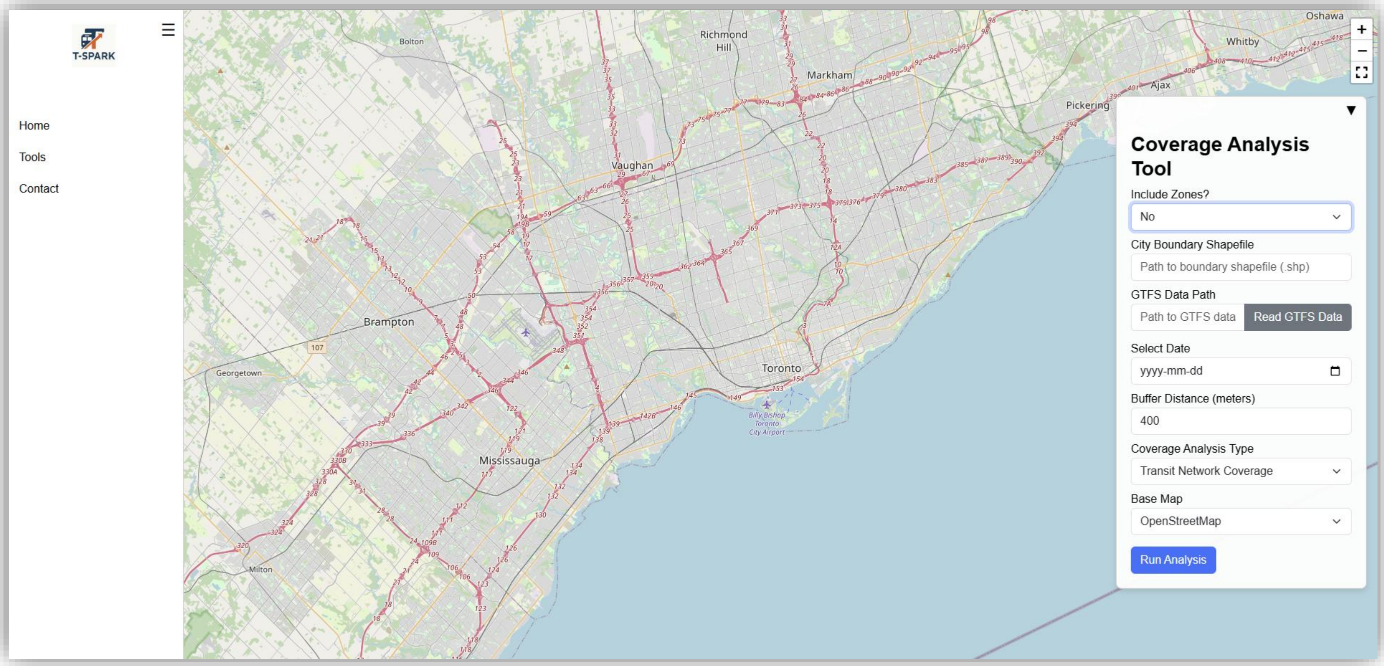
Overview: This tool helps you understand the spatial coverage of your transit service. You can either map the "catchment area" of your entire network (or a subset of routes) or identify "transit deserts" – areas within your city that are far from any transit stop.

Required Inputs:

- **GTFS Data Path:** The file path to your unzipped GTFS folder.
- **Boundary Shapefile Path:** The file path to your city boundary shapefile.
- **Buffer Distance:** The distance (in meters) to draw around each transit stop to define its service area (a common value is 400m, representing a 5-minute walk).
- **Date and Routes:** You will select a specific date for the analysis and can choose to analyze all routes or a specific subset of routes.

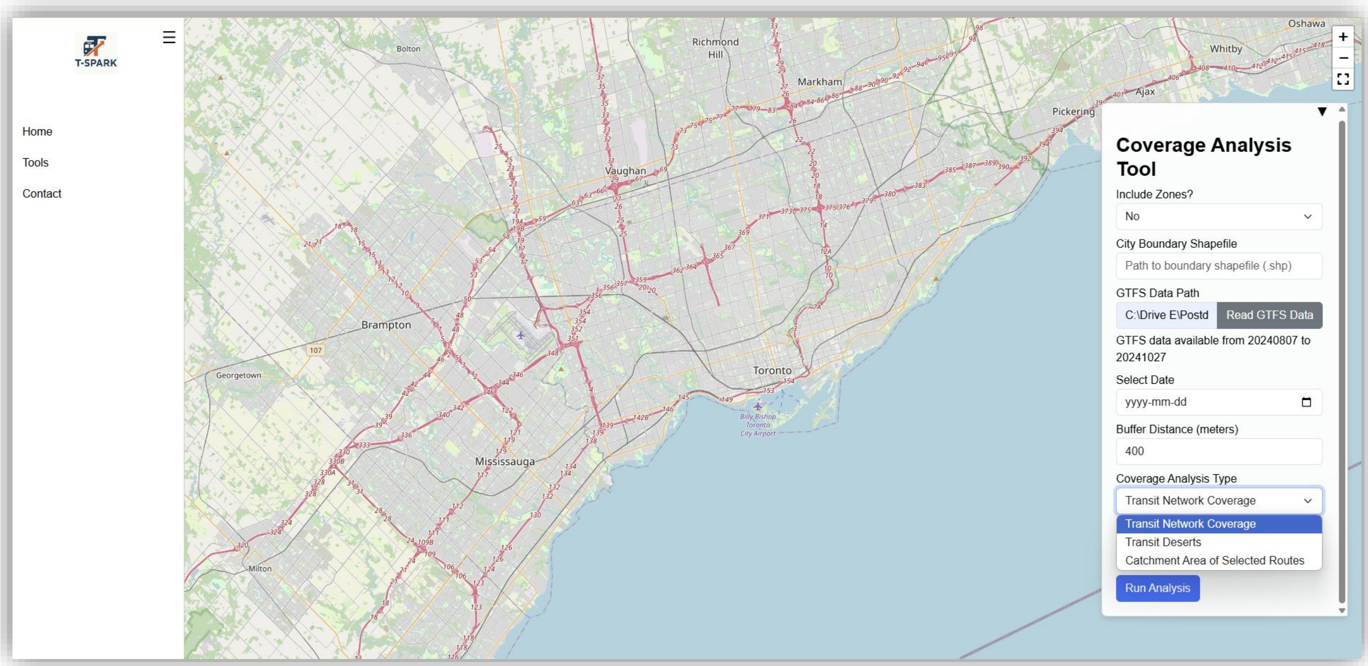
Step-by-Step Guide:

1. Navigate to the "Coverage Analysis" tool.

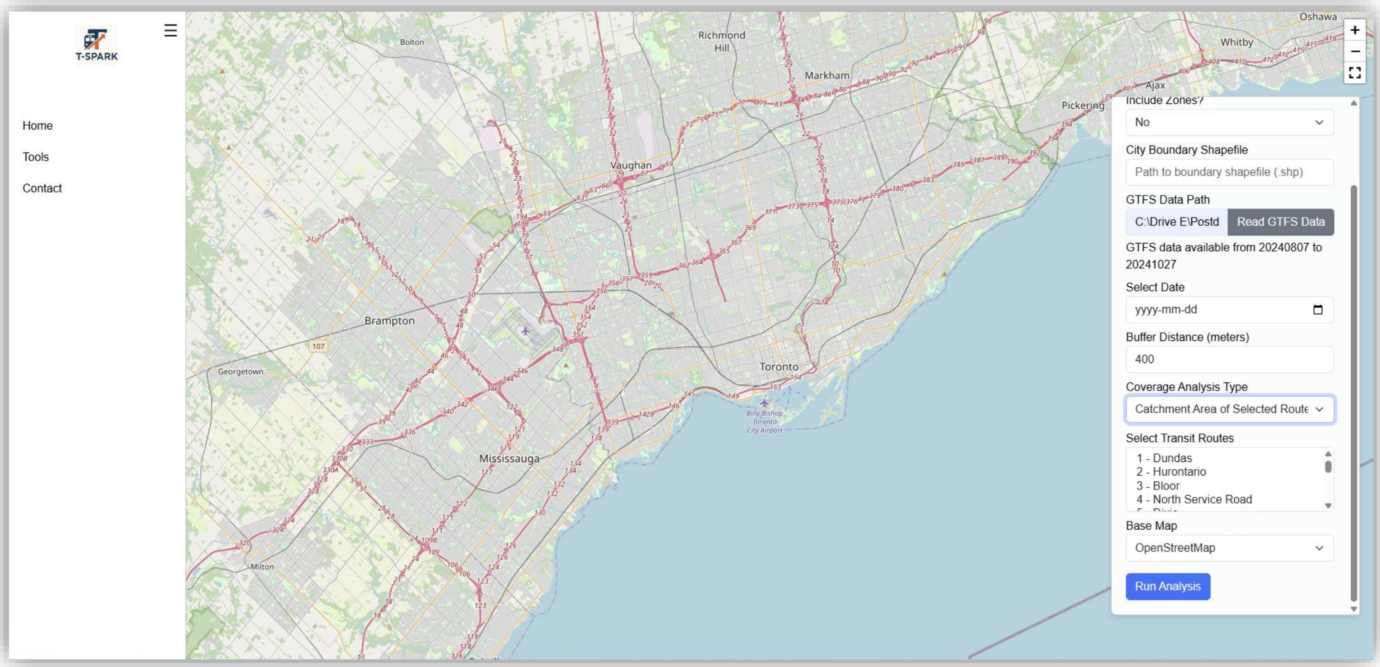


2. First indicate if you want to include zones on your coverage map. If you select “Yes” you need to provide details of target zone and path to the zones file.
3. Provide the paths to your GTFS data and boundary shapefile.
4. Select a date based on your GTFS data. “Read GTFS Data” will help you to see the covered dates in your GTFS data set. You will need to select the operation period as well. Your options will be “AM peak”, “PM peak”, “Mid-Day”, “After PM peak”, and “Overnight”.
5. Set the Buffer Distance.

6. Choose the Analysis Type ("Transit Network Coverage", "Transit Deserts" or “Catchment Area of Selected Routes”).



7. If you select “Catchment Area of Selected Routes”, you will be asked to select a set of routes. Not that to see the list of routes you need to click on “Read GTFS Data” in advance.

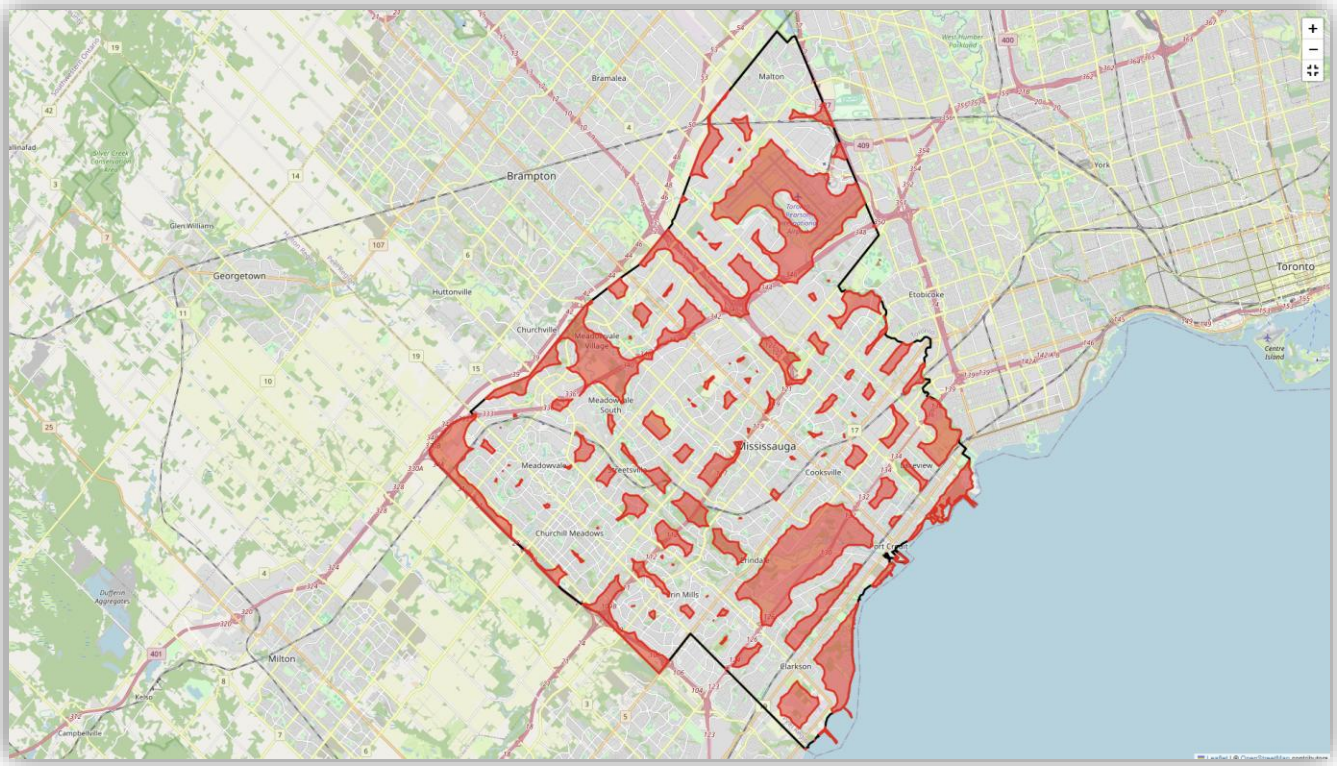


8. Click "Run Analysis".

Understanding the Outputs:

- **Catchment Area:** The map will show a blue-shaded area representing all locations within the specified buffer distance of an active transit stop (within the network or a set of transit routes).
- **Transit Deserts:** The map will show red-shaded areas. These are the parts of your city that are outside the transit catchment area, indicating poor transit access.

Below is a sample output illustrating the transit deserts within the City of Mississauga during AM peak in a weekday.



T-SPARK is developed at Transit Analytics Lab, University of Toronto

9. Adjacent Network Analysis

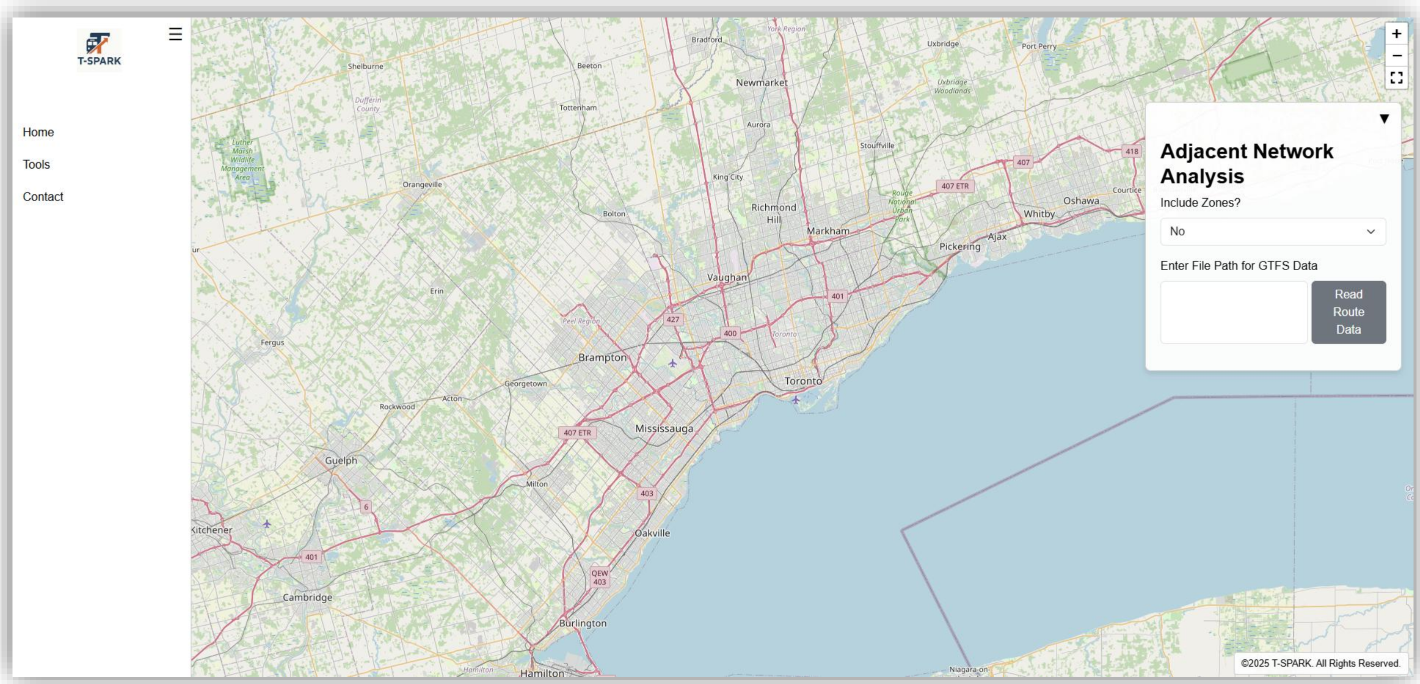
Overview: This is a visualization tool that allows you to map and inspect transit routes. Its primary purpose is to help you analyze routes that are adjacent to or overlap with each other. You can display two groups of routes in different colors for easy comparison.

Required Inputs:

- **GTFS Data Path**: The file path to your unzipped GTFS folder.
- **Zone Shapefile Path (Optional)**: The path to a shapefile of your zones, which can be displayed as a background layer.
- **Routes**: You can select routes to be displayed in two categories: "Under Evaluation" and "Adjacent".

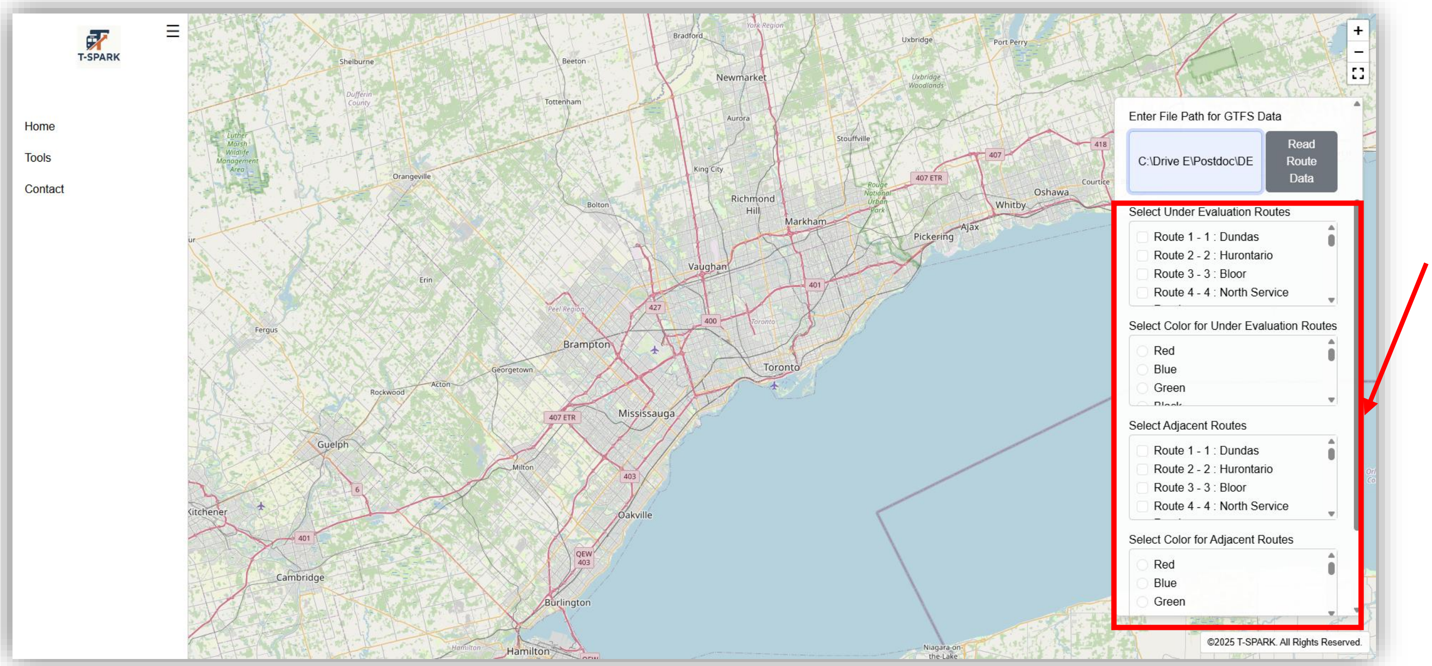
Step-by-Step Guide:

1. Navigate to the "Adjacent Network Analysis" tool.



2. If you decide to include zones' boundaries on your map, you will be asked to input path to your zones' shape file.
3. Provide the path to your GTFS data and click on "Read Route Data".

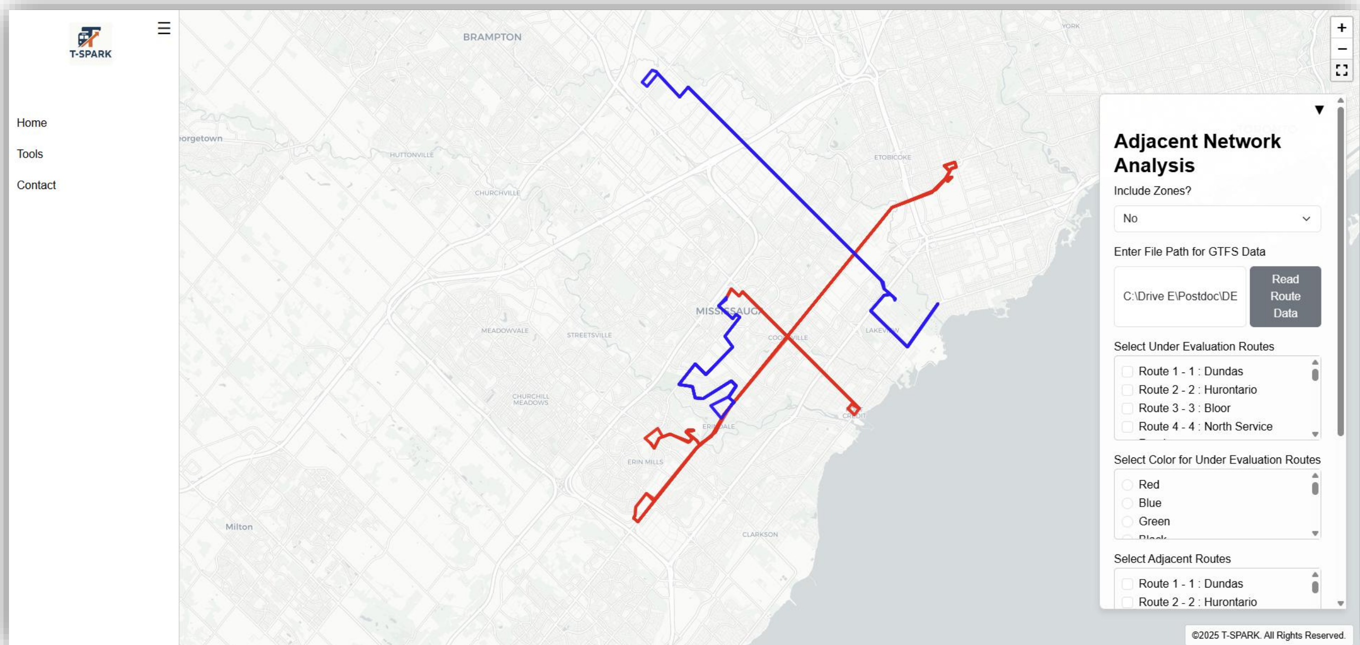
4. Select the routes you want to display and categorize them.



5. Choose colors for each category.
6. Click "Run Analysis" to generate the map.

Understanding the Outputs

The tool produces an interactive map showing the selected transit routes, colored according to the categories you defined. This is useful for visual analysis of your network's structure and investigates overlaps. Below is a sample output illustrating a set of selected under evaluation routes versus a set of selected adjacent routes within the transit network of the City of Mississauga.



7. DEA Engine

Overview: DEA Engine is the core of T-Spark's performance analysis capabilities. It uses sophisticated mathematical models to calculate the efficiency of transit routes (referred to as Decision Making Units or DMUs) in both route and segment level. This tool allows you to go beyond simple metrics and understand how effectively a route uses its inputs to produce its outputs. You can analyze efficiency for a single period or measure productivity changes over multiple periods using the Malmquist Productivity Index (MPI). Data Envelopment Analysis (DEA) is the core methodology behind the DEA Engine. It enables a relative performance evaluation by benchmarking each transit unit against every other unit across the network, thereby producing relative efficiency scores and highlighting opportunities for improvement. User can also choose to do analysis using traditional absolute performance analysis method. We will discuss these methods later.

Required Inputs:

- **Data Files (.xlsx):** You will need one or more Excel files containing your input and output data. The file(s) must have the following structure:
    - **Column A:** DMU ID (e.g., route_id)
    - **Column B:** DMU Name (e.g., route_short_name)
    - **Input Columns:** The next set of columns should be your input variables (e.g., vehicle_hours, fuel_cost, number_of_staff).
    - **Output Columns:** The final set of columns should be your output variables (e.g., ridership, vehicle_kilometers).
- **Number of Inputs/Outputs:** You must specify the number of input and output columns in your data file.

- **Calculation Type:**
  - **Absolute Efficiency:** Requires a list of weights for each input and output.
  - **Relative Efficiency (DEA):** No weights needed. Requires selecting a DEA model and orientation.
  - **MPI (Malmquist Productivity Index):** Requires data files for two time periods (t and t+1).

Methodology Deep Dive:

This tool is built on **Data Envelopment Analysis (DEA)**, a non-parametric method for measuring the relative efficiency of DMUs.

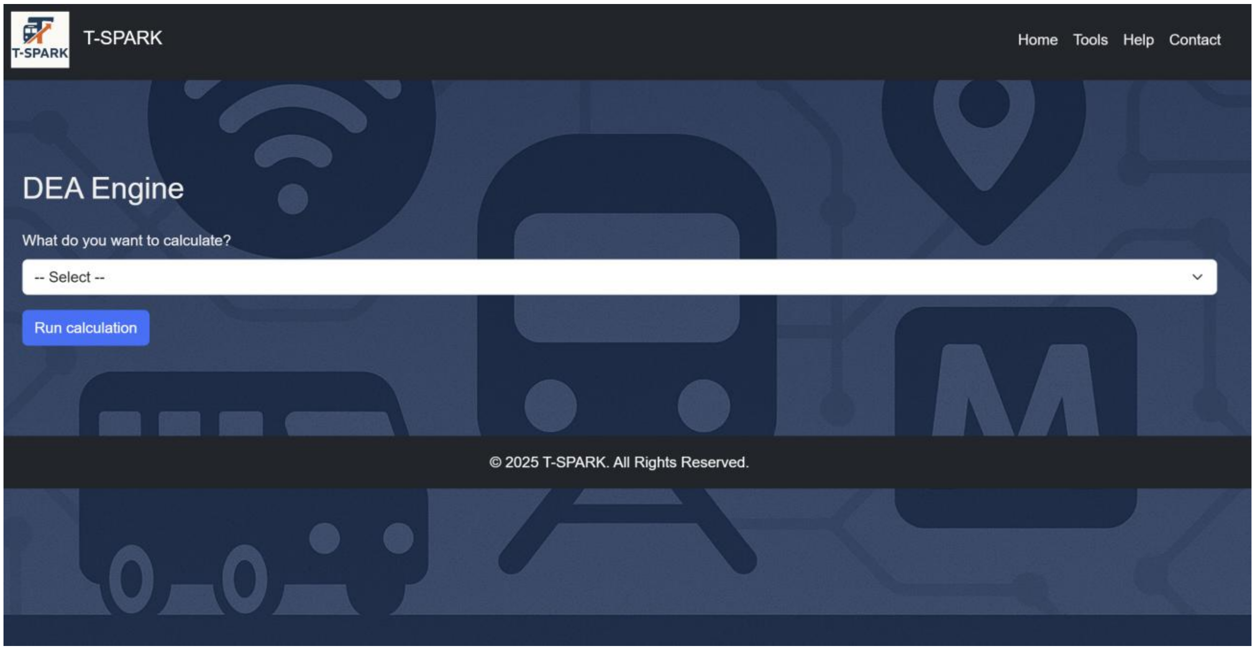
- **Efficiency Score:** A score of **1.0** means the route is efficient relative to its peers. A score **less than 1.0** indicates inefficiency. For an input-oriented model, a score of 0.8 means the route could theoretically produce the same outputs with only 80% of its inputs.
- **DEA Models:**
  - **CCR (Charnes, Cooper & Rhodes):** This is the original DEA model. It assumes **Constant Returns to Scale (CRS)**, meaning that a proportional increase in inputs will result in the same proportional increase in outputs.
  - **BCC (Banker, Charnes & Cooper):** This model is an extension of CCR that assumes **Variable Returns to Scale (VRS)**. It is useful when you believe that an increase in inputs may not result in a proportional increase in outputs (e.g., due to operational constraints).
  - **SBM (Slack-Based Measure):** This is a more advanced model that directly accounts for "slacks" (the amount by which inputs could be reduced or outputs increased). Unlike CCR and BCC, it is non-radial and can provide a more nuanced view of inefficiency.

- **Model Orientations:**
  - **Input-Oriented:** Aims to find the maximum possible reduction in inputs while keeping outputs constant. The question it answers is: "How much can I reduce my inputs without affecting my service levels?"
  - **Output-Oriented:** Aims to find the maximum possible increase in outputs with the current level of inputs. The question is: "How much more service can I provide with my current resources?"
  - **Non-Oriented:** Aims to do both simultaneously, maximizing output and minimizing input. This is the approach taken by the SBM model.
- **Malmquist Productivity Index (MPI):** When you have data for two time periods, the MPI measures the change in productivity. It is broken down into two components:
  - **Efficiency Change (Catch-up):** Did the route get better at using its resources (i.e., did it move closer to the efficiency frontier)?
  - **Technical Change (Frontier Shift):** Did the overall "best practice" technology of the whole system improve (i.e., did the efficiency frontier itself shift)?

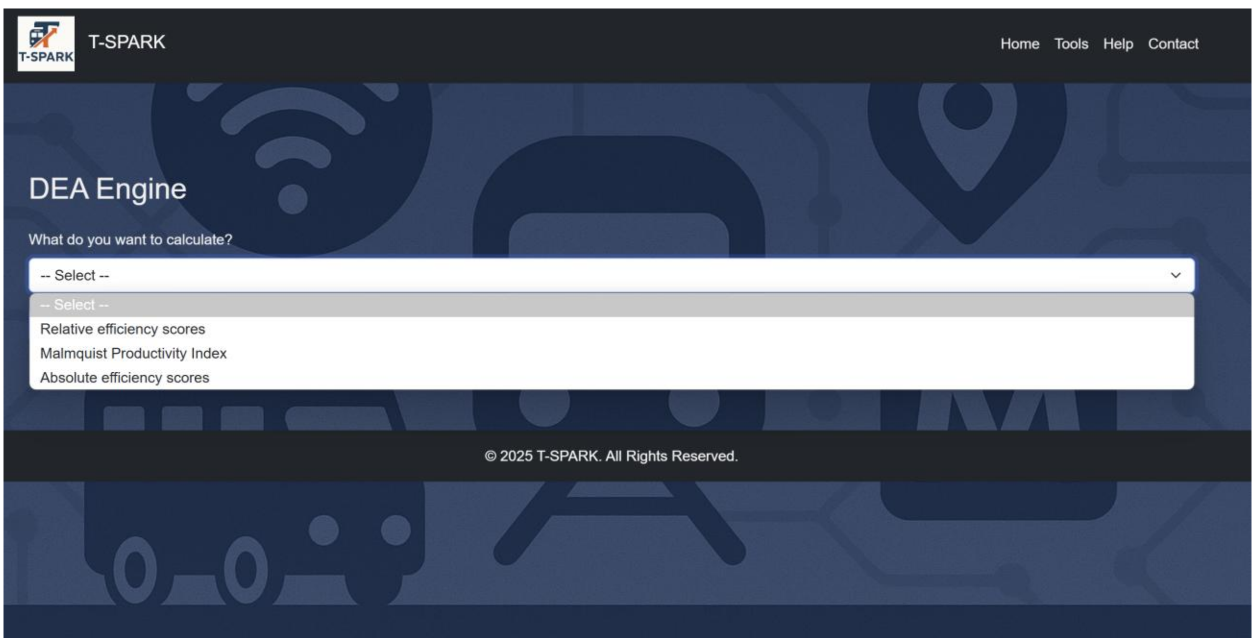
For details of the included models in DEA Engine see Appendix.

Step-by-Step Guide

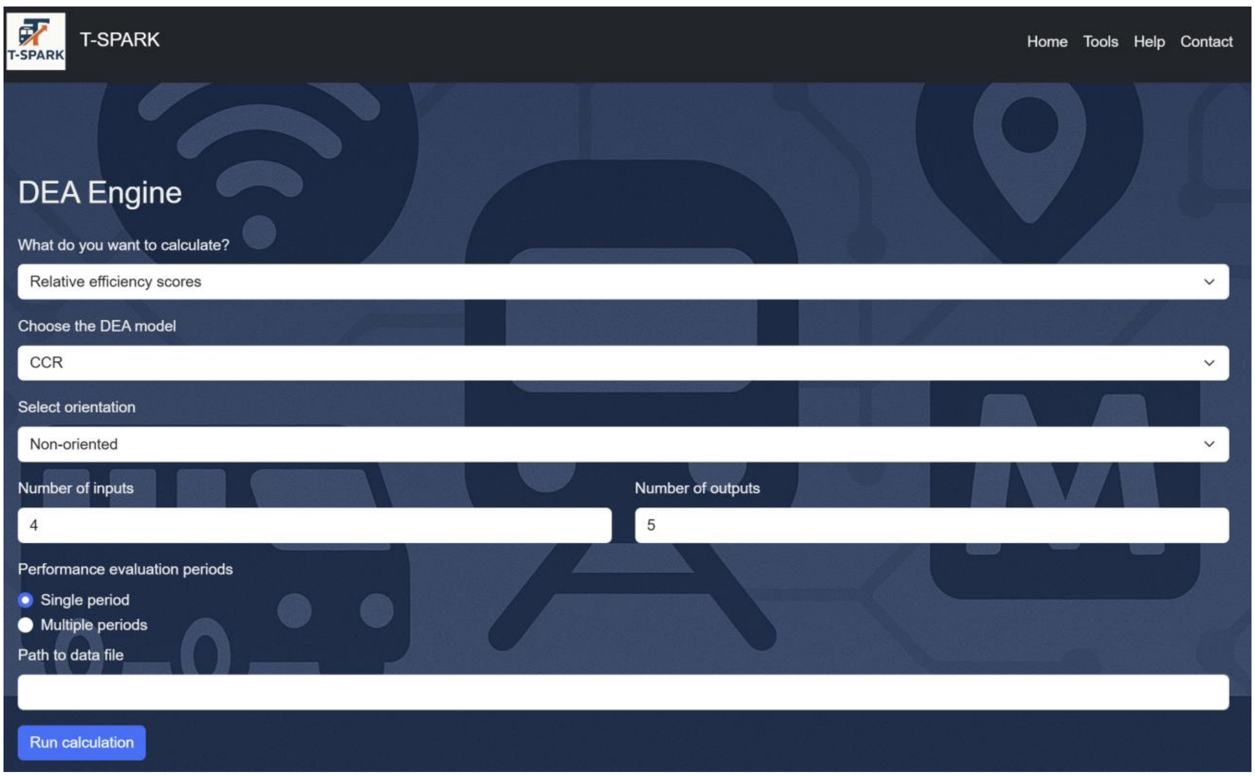
1. Navigate to the "DEA Engine" tool.



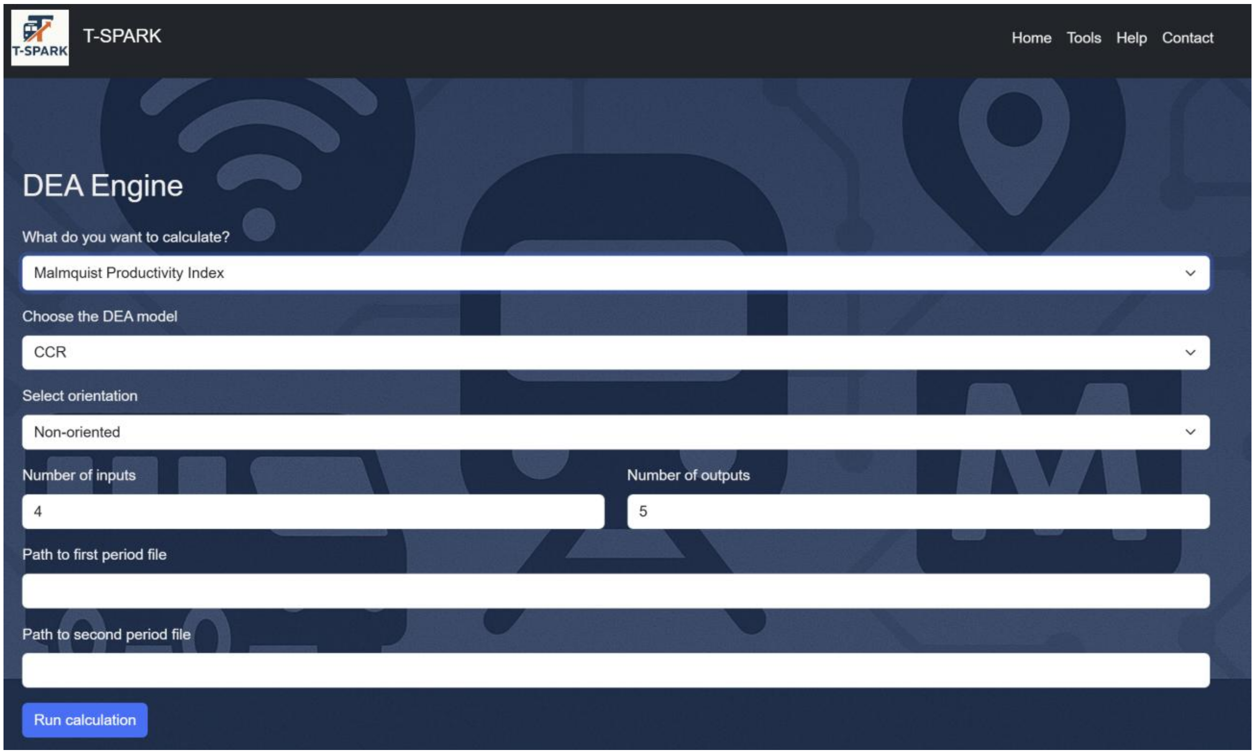
2. Select the Calculation Type (Absolute, Relative, or MPI).



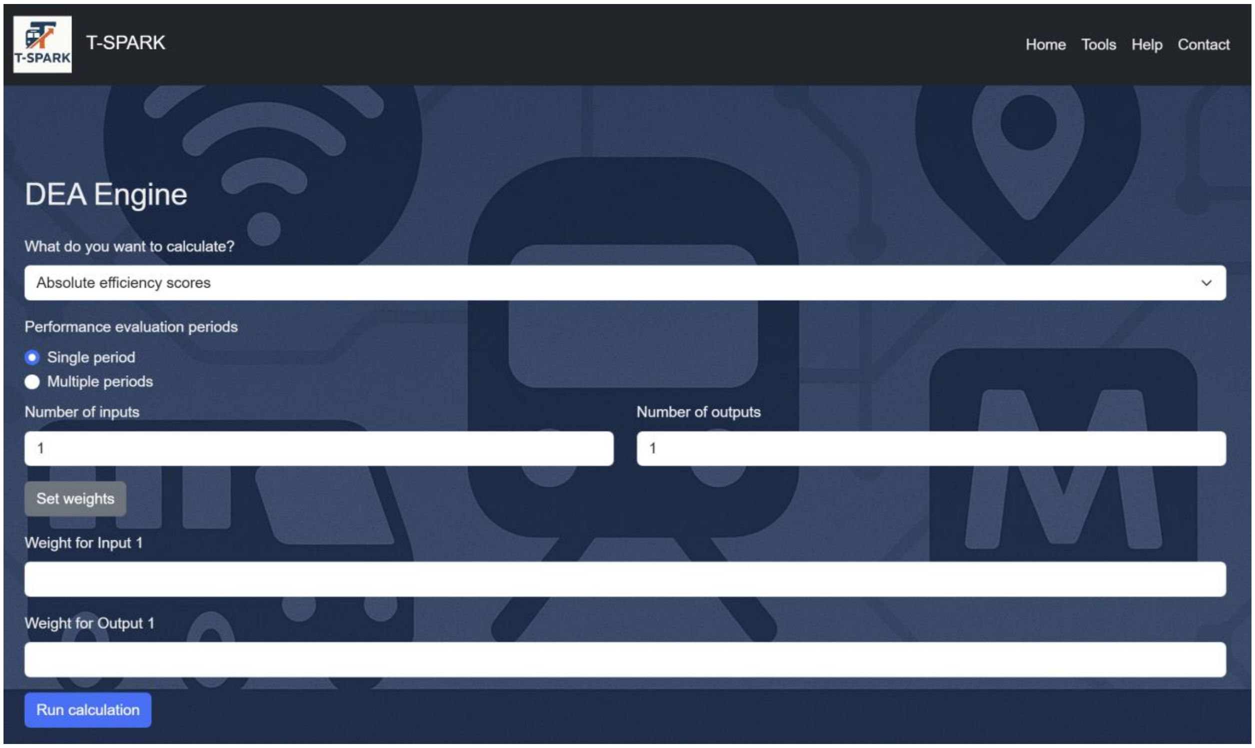
3. Based on your choice, fill in the required parameters (e.g., upload data files, specify the number of inputs/outputs, select a DEA model).
    a) Relative efficiency score: If you choose this option, you will need to choose the DEA model and the orientation of the selected DEA model. Insert the number of inputs and outputs that you have considered for your analysis and have included them in your Excel file. Next, you should decide if you want to conduct a single- or multi-period performance analysis. Depending on your decision, you will need to insert the path to the related Excel file for each evaluation period.



b) Malmquist Productivity Score: If you choose this option, everything is similar to the first option, unless in this case you will always need to insert the path to two different operations period.



c) Absolute Efficiency score: Selecting this option applies a traditional absolute performance metric, calculated as the ratio of the weighted sum of normalized outputs to the weighted sum of normalized inputs. You will be prompted to specify a weight for each input and each output variable.



4. Click "Compute".
5. The results will be saved as Excel files in a results folder on the server.

10. Performance Evaluation

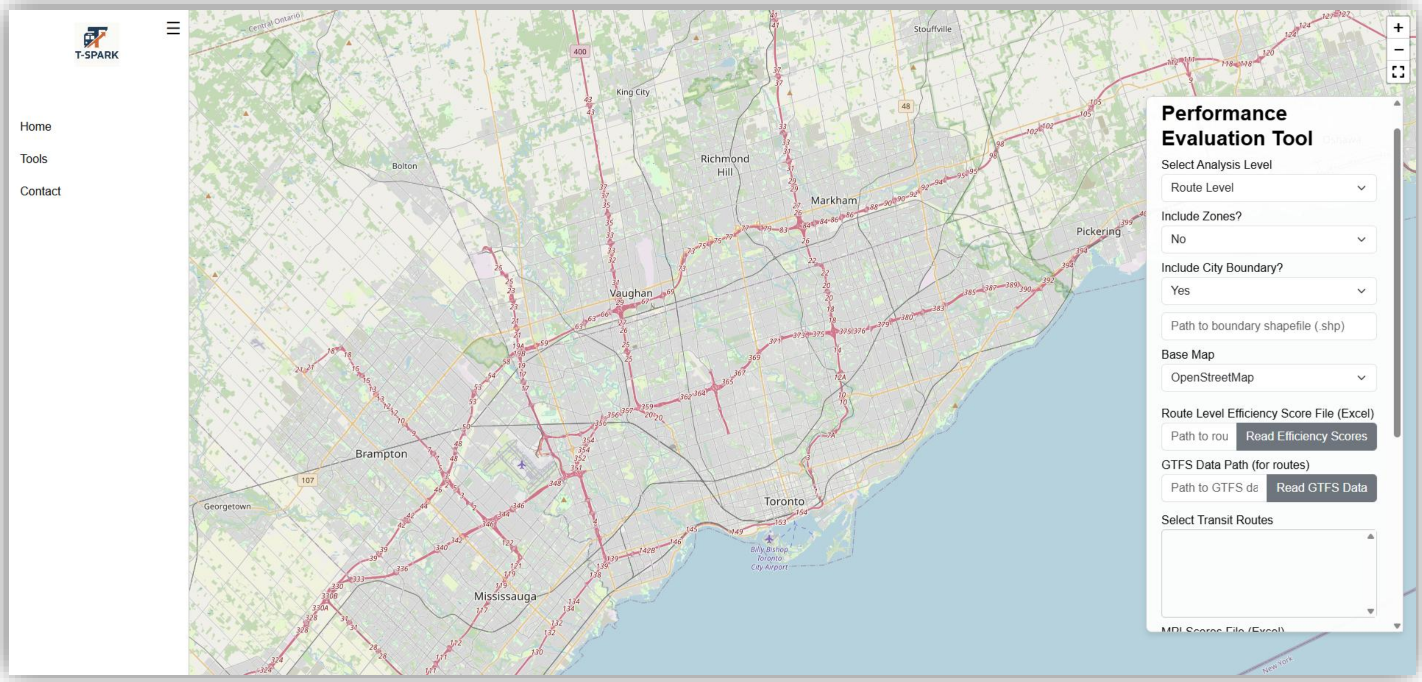
Overview: This tool is the visualization counterpart to the "DEA Engine". It takes the efficiency scores you calculated and maps them directly onto your transit network. This allows you to see the spatial patterns of performance, easily identifying high- and low-performing routes or route segments, and zones that the transit network has generally low performance. You can also generate plots to see performance trends over time based on the efficiency scores or different MPI values.

Required Inputs:

- **Analysis Level:**
  - Route Level: Requires a GTFS data path and an Excel file with route-level efficiency scores.
  - Segment Level: Requires an Excel file with segment-level efficiency scores.
- **Efficiency Files (.xlsx):**
  - Route Level: An Excel file with a route_id column and other columns representing efficiency scores for different time periods (e.g., Period_1, Period_2).
  - Segment Level: An Excel file with a route_id column, a geometry column (containing the line geometry of the segment in WKT format), and columns for efficiency scores.
- **Other Files (.xlsx):** You can also upload files for MPI, Efficiency Change, and Technical Change to create interactive charts.

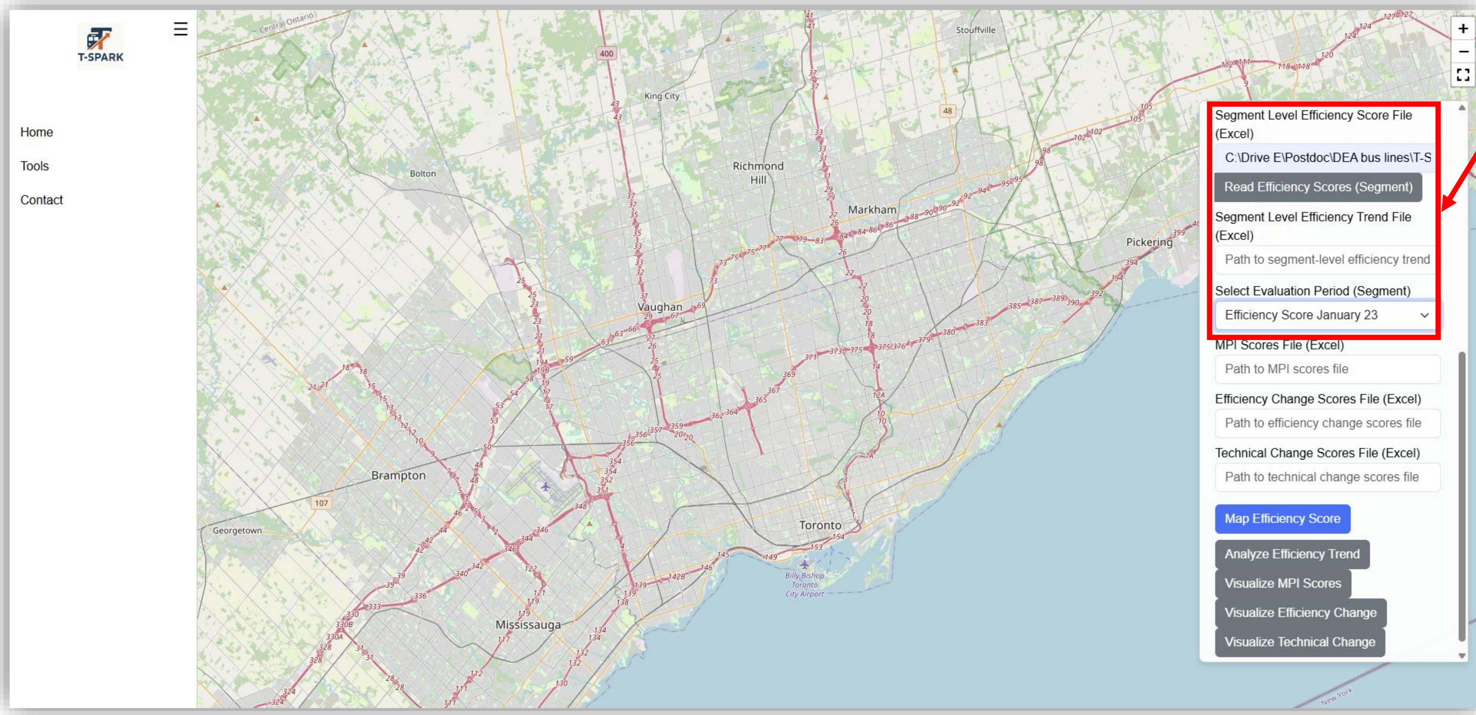
Step-by-Step Guide

1. Navigate to the "Performance Evaluation" tool.

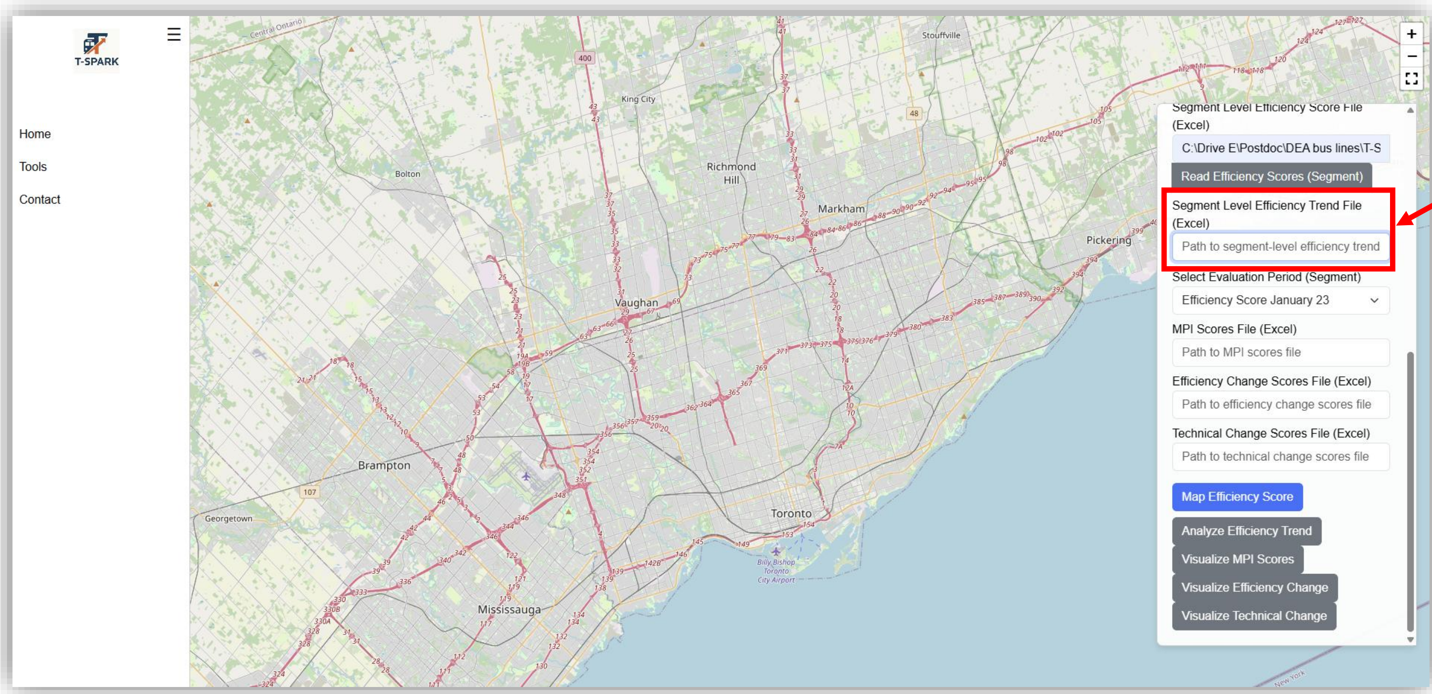


2. Select the Analysis Level. You have two options: Route Level or Segment Level. Segment Level will provide more detailed analysis of transit networks performance. However, the line geometry of the segments in WKT format is an input for T-SPARK and the user must obtain it in advance.

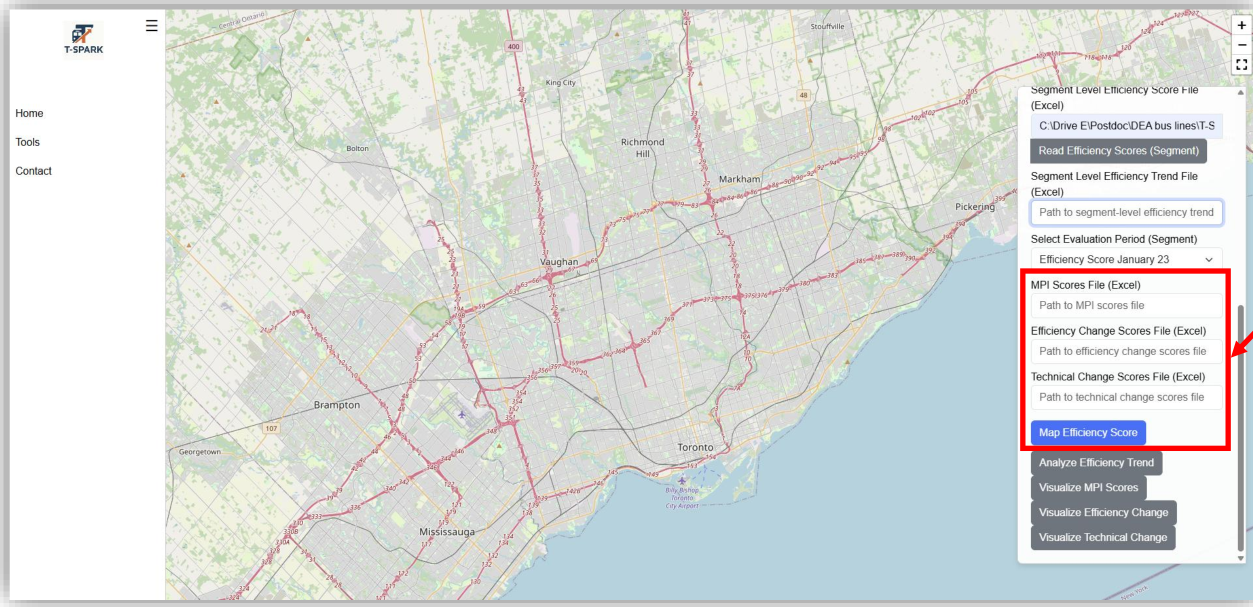
3. Upload the required GTFS data and/or efficiency files.
4. If you decide to include zones on the map you will need to enter the path to the shape files of the zones.
5. If you decide to include the city boundary on the map you will need to enter the path to the shape files of the boundary.
6. Select the format of the base map.
7. Based on your chosen analysis level, enter the file path to the corresponding efficiency score Excel file. Then click “Read Efficiency Score (selected level of analysis)”. The application will parse the workbook and display a dropdown listing all operation periods contained in the file; simply select one to visualize its results.



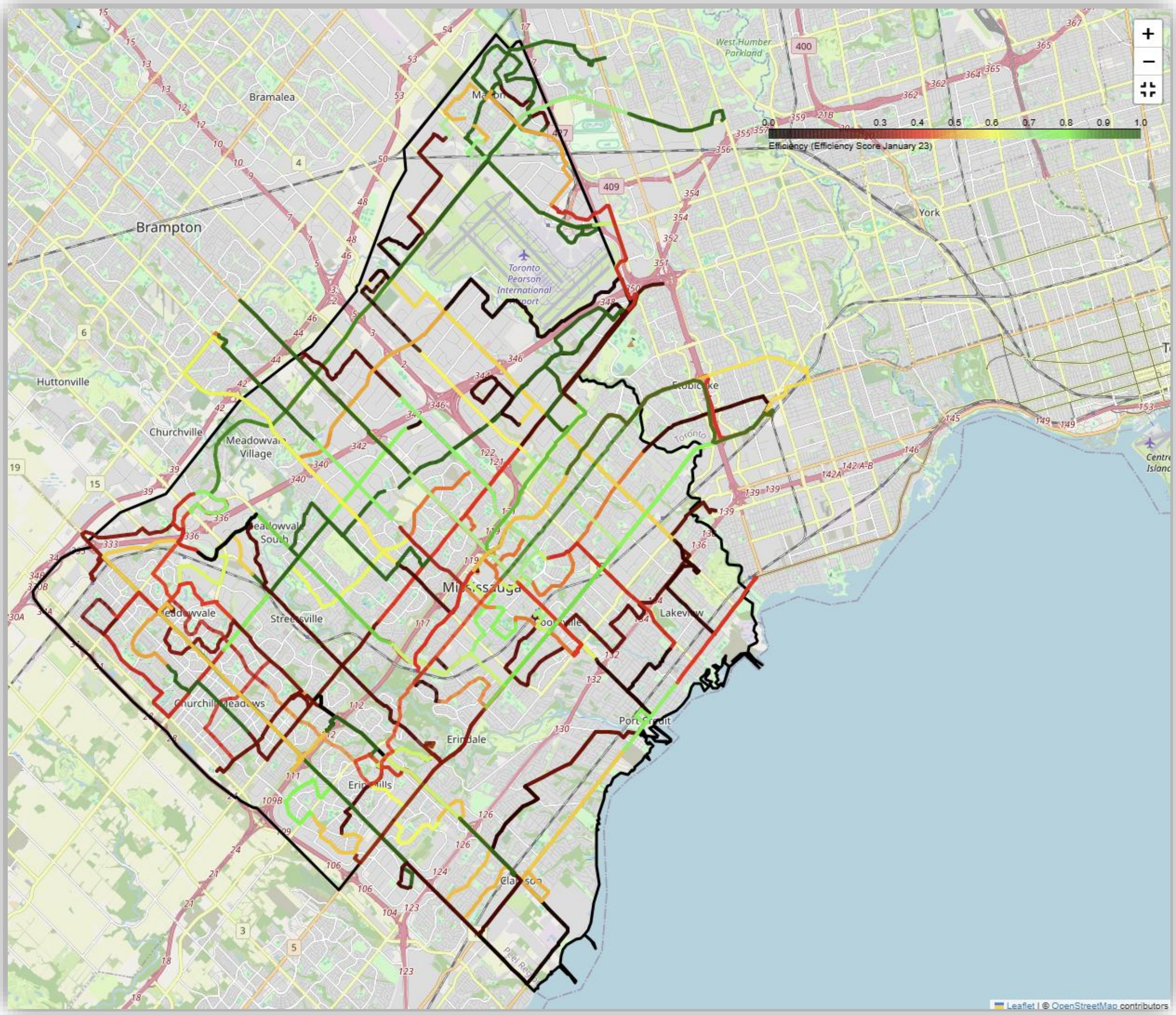
8. If you wish to analyze the trend of the efficiency scores of transit routes or segment during the evaluation period, enter the path to an excel file that includes efficiency score of all units during the evaluation period.



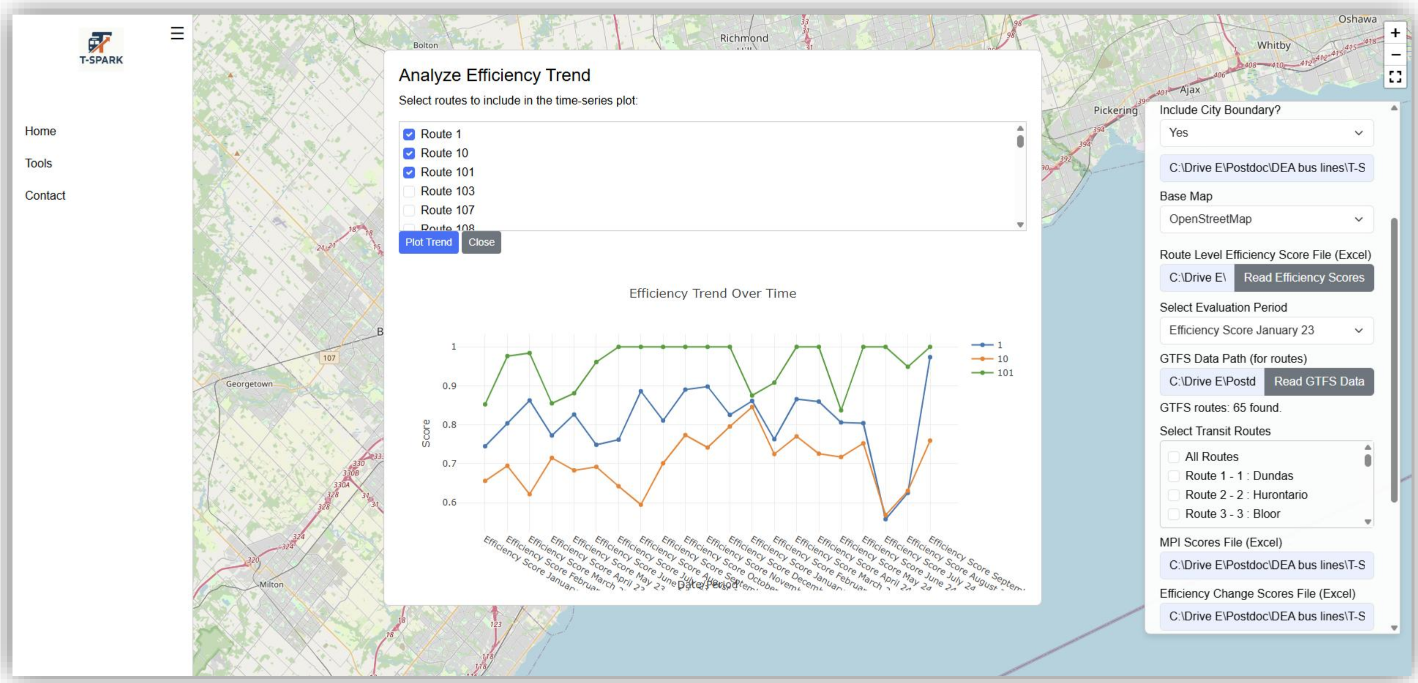
9. Enter path to MPI score, efficiency change score and technical change scores.



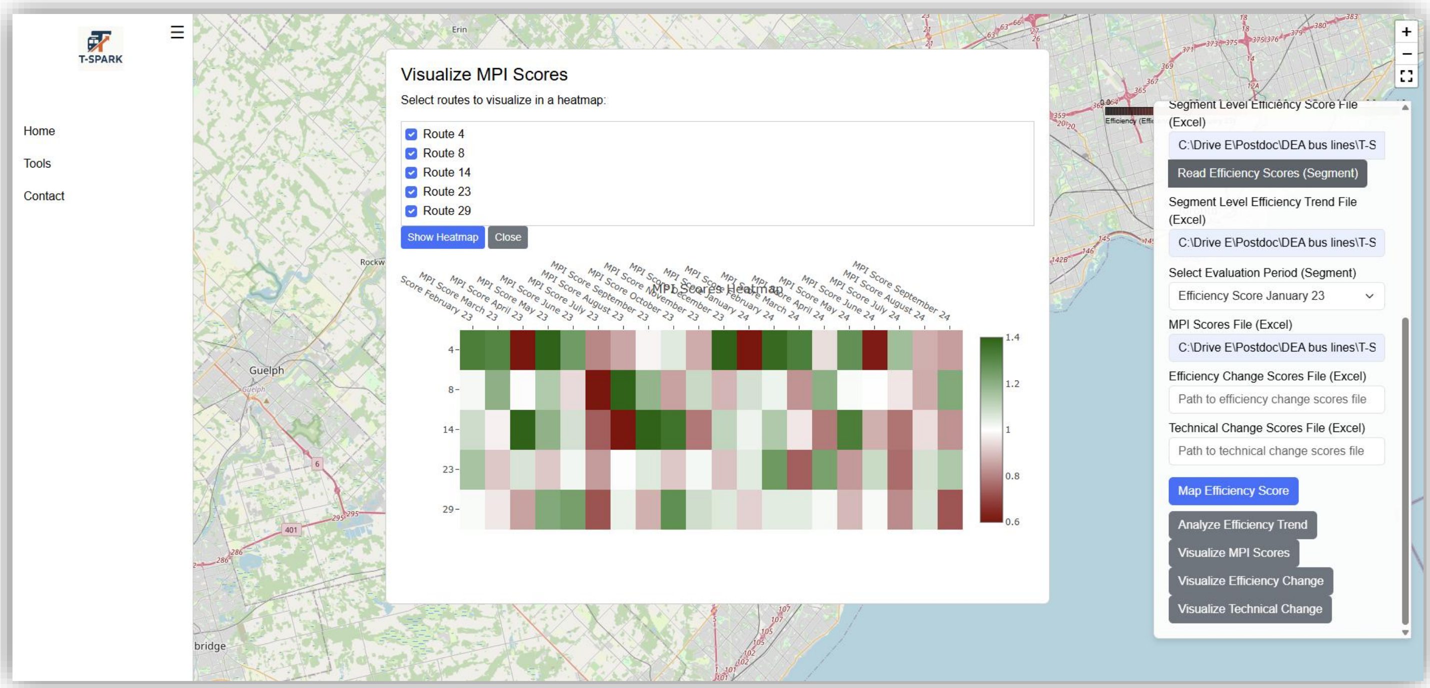
10. Click Map Efficiency Scores to display the transit network overlaid with efficiency scores for each route or segment. The example below shows the City of Mississauga network mapped by transit route segments efficiency, using data from January 2023.



11. Click “Analyze Efficiency Trend” to select one or more routes or segments and generate charts illustrating their performance over time. You can download these plots or customize them to your preferred format. The example below shows how the performance of routes 1, 10, and 101 in the City of Mississauga changed from January 2023 through September 2024.



12. Click “Visualize MPI scores” to select one or more routes or segments and generate heatmaps illustrating their MPI scores change over time. You can download these heatmaps or customize them to your preferred format. The example below shows how the MPI scores of of routes 4, 8, 14, 23, and 28 in the City of Mississauga have been changed from January 2023 through September 2024.



13. “Visualize Efficiency Change” and “Visualize Technical Change” function exactly like Visualize MPI Scores, allowing you to generate time-series charts that track changes in overall efficiency or technical performance.

Understanding the Outputs:

- **Map:** An interactive map where transit routes or segments are color-coded by their efficiency score. A legend is provided to help you interpret the colors (typically green for high efficiency and red/black for low efficiency).
- **Charts:**
    - Efficiency Trend: A line chart showing how the efficiency of selected routes has changed over multiple periods.
    - MPI, Efficiency Change, Technical Change: Interactive heatmaps that visualize these values for different routes over different time periods.

11. Troubleshooting

“File not found” errors: verify correct path and permissions.

Shapefile missing gta06: ensure your TAZ layer includes that field.

GTFS parsing issues: check that routes.txt, trips.txt, shapes.txt are present.

Timeouts on large networks: increase server timeout or simplify area.

12. FAQ

Q: Can I use T-SPARK for any city?

A: Yes, just upload the appropriate shapefiles & GTFS for any region.

Q: How do I adjust input/output weights?

A: In the **DEA Engine** tool, specify your weight vectors in the form for traditional evaluation approach.

Q: Should I use a specific version of r5py?

A: Yes, the current version of T-SPARK (i.e., v 1.0.0) is designed to work with the latest available version of r5py to date (i.e., v 1.0.5).

13. Support & Contact

T-SPARK is developed by Reza Mahmoudi (principal developer), Amer Shalaby (supervisor), Diego Da silva (contributor), and Shahrin Islam (contributor), all from Transit Analytics Lab, University of Toronto. All rights are reserved by the developers.

For bug reports, feature requests, collaboration requests, or questions, please email the development team.

Reza Mahmoudi: reza.mahmoudi@utoronto.ca | j.mahmoudi.reza@gmail.com

Amer Shalaby: amer.shalaby@utoronto.ca

Diego Da Silva: dluiz.silva@utoronto.ca

Shahrin Islam: shahrin.islam@mail.utoronto.ca

Phone: +1 587 438 5322

Address: 35 St George St, Toronto, ON M5S 1A4

Appendix

Some Mathematical Formulations (linear formulations) embedded in DEA Engine:

- **CCR Models:**
  - **Input-Oriented:**

$$
\begin{aligned}
\min_{\theta,\lambda} \quad & \theta \\
\text{subject to} \quad & \sum_{j=1}^{n} \lambda_j x_{ij} \leq \theta x_{io} \quad \text{for } i = 1, \dots, m \\
& \sum_{j=1}^{n} \lambda_j y_{rj} \geq y_{ro} \quad \text{for } r = 1, \dots, s \\
& \lambda_j \geq 0 \quad \text{for } j = 1, \dots, n
\end{aligned}
$$

Where:

- θ is the efficiency score of *DMUo* ($0<\theta\leq 1$).
- λj are the weights or intensity vectors for each *DMUj* in the reference set.
- The constraints ensure that a virtual "composite DMU" (a linear combination of other *DMUs*) is created that is no less efficient than *DMUo*.

  - **Output-Oriented:**

$$
\begin{aligned}
\max_{\phi,\lambda} \quad & \phi \\
\text{subject to} \quad & \sum_{j=1}^{n} \lambda_j x_{ij} \leq x_{io} \quad \text{for } i = 1, \dots, m \\
& \sum_{j=1}^{n} \lambda_j y_{rj} \geq \phi y_{ro} \quad \text{for } r = 1, \dots, s \\
& \lambda_j \geq 0 \quad \text{for } j = 1, \dots, n
\end{aligned}
$$

Where $\phi = 1/\theta$.

- **Non-Oriented:**

$$\min_{\theta,\lambda,s^-,s^+} \theta - \epsilon\Big(\sum_i s_i^- + \sum_r s_r^+\Big)$$

$$\text{s.t.} \quad \sum_j \lambda_j x_{ji} + s_i^- = \theta\, x_{oi}, \quad i = 1..m,$$

$$\sum_j \lambda_j y_{jr} - s_r^+ = y_{or}, \quad r = 1..s,$$

$$\lambda_j, s_i^-, s_r^+ \geq 0.$$

Where s^- and s^+ capture input excesses and output shortfalls.

- **BCC Models:** The BCC model relaxes the CRS assumption to allow for **Variable Returns to Scale (VRS)**. This is more flexible and accounts for situations where efficiency might change with the scale of operations. The BCC model measures *pure technical efficiency*. The mathematical formulation is nearly identical to the CCR model, with the addition of a single "convexity" constraint:

$$\sum_{j=1}^{n} \lambda_j = 1$$

This constraint ensures that the DMU under evaluation is only compared to other DMUs of a similar scale. The efficiency scores from a BCC model are always greater than or equal to those from a CCR model.

- **SBM Models (Slack-Based Measure):** The CCR and BCC models are "radial," meaning they reduce inputs or expand outputs proportionally. The SBM model is non-radial and directly measures inefficiency through input/output "slacks." A slack is the amount by which an input can still be decreased (or an output increased) even after the proportional reduction.

- **Input-Oriented:**

$$\min_{\theta,\lambda,s^-,s^+} \quad \theta - \frac{1}{m}\sum_{i=1}^{m}\frac{s_i^-}{x_{oi}}$$

$$\text{s.t.} \begin{cases} \sum_{j=1}^{n}\lambda_j x_{ji} + s_i^- = \theta\, x_{oi}, \quad i = 1..m, \\ \sum_{j=1}^{n}\lambda_j y_{jr} - s_r^+ = \theta\, y_{or}, \quad r = 1..s, \\ \theta + \frac{1}{s}\sum_{r=1}^{s}\frac{s_r^+}{y_{or}} = 1, \\ \sum_{j=1}^{n}\lambda_j = 1, \\ \lambda_j, s_i^-, s_r^+ \geq 0. \end{cases}$$

- **Out-Oriented:**

$$\max_{\theta,\lambda,s^-,s^+} \quad \theta + \frac{1}{s}\sum_{r=1}^{s}\frac{s_r^+}{y_{or}}$$

$$\text{s.t.} \begin{cases} \sum_{j=1}^{n}\lambda_j x_{ji} + s_i^- = \theta\, x_{oi}, \quad i = 1..m, \\ \sum_{j=1}^{n}\lambda_j y_{jr} - s_r^+ = \theta\, y_{or}, \quad r = 1..s, \\ \theta - \frac{1}{m}\sum_{i=1}^{m}\frac{s_i^-}{x_{oi}} = 1, \\ \sum_{j=1}^{n}\lambda_j = 1, \\ \lambda_j, s_i^-, s_r^+ \geq 0. \end{cases}$$

- **Non-Oriented:**

$$\min_{\theta,\lambda,s^-,s^+} \quad \theta - \frac{1}{m}\sum_{i=1}^{m}\frac{s_i^-}{x_{oi}}$$

$$\text{s.t.} \begin{cases} \theta\, x_{oi} - s_i^- = \sum_{j=1}^{n}\lambda_j x_{ji}, \quad i = 1..m, \\ \theta\, y_{or} + s_r^+ = \sum_{j=1}^{n}\lambda_j y_{jr}, \quad r = 1..s, \\ \theta + \frac{1}{s}\sum_{r=1}^{s}\frac{s_r^+}{y_{or}} = 1, \\ \sum_{j=1}^{n}\lambda_j = 1, \\ \lambda_j, s_i^-, s_r^+ \geq 0. \end{cases}$$

- **Malmquist Productivity Index (MPI):** When you have panel data (data for the same DMUs over multiple time periods, t and t+1), the MPI measures the change in productivity. It is calculated as:

$$MPI = \left[\frac{E^t(x_{t+1}, y_{t+1})}{E^t(x_t, y_t)} \times \frac{E^{t+1}(x_{t+1}, y_{t+1})}{E^{t+1}(x_t, y_t)} \right]^{1/2}$$

  Where $E^t(x_t, y_t)$ is the efficiency of a DMU from period t measured against the frontier from period t. The "cross-period" term $E^t(x_{t+1}, y_{t+1})$ measures the efficiency of a period t+1 DMU against the technology of period t.

  The MPI can be decomposed into two components:

  - **Efficiency Change (Catch-up):** Did the DMU get closer to its own period's frontier?

$$EC = \frac{E^{t+1}(x_{t+1}, y_{t+1})}{E^t(x_t, y_t)}$$

  - **Technical Change (Frontier Shift):** Did the frontier itself move?

$$TC = \left[\frac{E^t(x_t, y_t)}{E^{t+1}(x_t, y_t)} \times \frac{E^t(x_{t+1}, y_{t+1})}{E^{t+1}(x_{t+1}, y_{t+1})} \right]^{1/2}$$

    An MPI, EC, or TC value greater than 1 indicates improvement, while a value less than 1 indicates a decline.

- **Absolute Efficiency score**
  - **Normalization:**

$$X_{ij}^n = \frac{x_{ij}}{\max_j x_{ij}}, \quad Y_{ir}^n = \frac{y_{ir}}{\max_i y_{ir}}$$

  - **Score for DMU:**

$$\theta_i = \frac{\sum_{r=1}^{s} w_r^o \, Y_{ir}^n}{\sum_{j=1}^{m} w_j^i \, X_{ij}^n}$$

where w^o_r and w^i_j are user-provided input/output weights.