Work realized by : Yasmine DALY M2 SIA

Université Gustave Eiffel

Management in Intelligent Systems
Fall 2023
Mme BHOURI Neila & M. FRANÇOIS Olivier

Project: Networks Reliability

Reliability is the quality of being trustworthy or performing consistently well. This performance may concern different aspects of a system's operation. In the case of transportation networks, two main reliability performances are studied: travel time reliability and connectivity reliability.

Connectivity reliability concerns the accessibility performances in the network, while travel time reliability studies the variability in travel time. It argues that traffic congestion is common in many cities, and in these cities, drivers are accustomed to congestion. They expect and plan for some delay, particularly during peak driving times. They need to adjust their schedules or budget extra time to allow for traffic delays.

The objective of this project is to study different ways to measure travel time reliability and to analyze the best way to take into account to guarantee being on time in most instances, despite the congestion. In paper [1], authors studied the impact of a management strategy, namely on-ramp metering, on travel time reliability. They used different reliability indicators to measure travel time reliability before and after the application of the ramp metering strategy. In your project, you will apply these indicators to a public transportation network and analyze their objective.

Work realized by : Yasmine DALY M2 SIA

Part 1: Main indicators

1. Read the paper and analyze the different indicators.

The paper "Managing Highways for Better Reliability - Assessing Reliability Benefits of Ramp Metering" by Neila Bhouri and Jari Kauppila analyzes various indicators to measure travel time reliability in the context of traffic management, particularly focusing on the use of ramp metering. The indicators are categorized into four main groups: Statistical range methods, Buffer time methods, Tardy trip measures, and Probabilistic measures. These indicators include

The paper provides detailed definitions and applications of various indicators:

1. COV (Coefficient of Variation): This is the ratio of the standard deviation to the mean travel time, reflecting travel time variability.

2. Percentiles (TT10, TT50, TT80, TT90, TT95): Percentiles represent travel times under which a certain percentage of trips are completed. For example, TT80 means 80% of trips are completed within this travel time.

3. λvar and λdev: These are statistical measures. λvar is related to the variance of travel times, while λdev refers to the deviation in travel times.

4. UIr (Undesirable Index of Reliability): This index measures the frequency and severity of undesirable travel times, indicating less reliable trips.

5. BI (Buffer Index): It measures the additional time required above the average travel time to ensure a high probability of on-time arrival.

6. PTI (Planning Time Index): This is the ratio of the time that must be planned for a trip to ensure on-time arrival under the worst usual conditions to the free-flow travel time.

7. MI (Misery Index): This index focuses on the most delayed trips, comparing the mean travel time of the most delayed travelers to the overall mean travel time.

8. Pr(TTi > B + TT50): This probability calculation measures the likelihood of a travel time exceeding a certain threshold, typically a buffer added to the median travel time.

Each indicator offers a unique perspective on travel time reliability, providing a comprehensive assessment of highway performance and the effectiveness of measures like ramp metering.

2. Load the matrix containing 50 **T**ravel **T**imes of the bus network in "TT_link.mat" *(based on "Route.mat" used in tutorials).* This matrix provides the travel time (in minutes) between two consecutive bus stops on a given bus route when a bus route is linking these stops.

```
load('C:\\Users\\Yasmine\\Downloads\\TT_link.mat');
travel_times = TT_link;
travel_times
disp(TT_link(1:10, 1:10))

```

```
ans(:,:,25) =

 Columns 1 through 11:

        0   2.7000        0        0        0        0   2.0600        0        0        0        0
   4.1600        0   3.3300        0        0        0        0   2.3700        0        0        0
        0   3.3300        0   2.6000        0        0        0   2.9300        0        0        0
        0        0   2.3400        0  10.4900        0        0        0        0        0        0
        0        0        0  10.3500        0   5.8500        0  10.4500   9.8300   9.9300        0
        0        0        0        0   5.6200        0        0        0        0        0        0
   2.5500        0        0        0        0        0        0        0        0        0   8.9100
        0   2.4300   2.4300        0   8.9800        0        0        0        0        0        0
        0        0        0        0  10.9600        0        0        0        0   2.2500        0
        0        0        0        0  14.0000        0        0        0   2.2400        0        0
        0        0        0        0        0        0   7.7400        0        0        0        0
        0   4.5300        0        0        0        0   4.1200        0        0        0        0
        0        0        0        0        0   9.5600        0        0        0        0        0
        0        0   8.2800        0        0        0        0        0        0        0        0
        0        0        0        0        0        0        0        0        0        0        0

 Columns 12 through 15:

        0        0        0        0
   3.9800        0        0        0
        0        0  10.9300        0
        0        0        0        0
        0        0        0        0
        0  12.9300        0        0
   3.1400        0        0        0
        0        0        0        0
        0        0        0        0
        0        0        0        0
        0        0        0        0
        0        0        0   3.6800
        0        0        0        0
        0        0        0        0
   5.3700        0        0        0
```

```
ans(:,:,50) =

 Columns 1 through 11:

        0   3.7700        0        0        0        0   2.9000        0        0        0        0
   3.9200        0   3.2600        0        0        0        0   2.0600        0        0        0
        0   3.9900        0   1.8400        0        0        0   2.1100        0        0        0
        0        0   2.9800        0   9.6000        0        0        0        0        0        0
        0        0        0   8.0800        0   5.0900        0   8.2400   9.6700  10.8400        0
        0        0        0        0   7.4000        0        0        0        0        0        0
   2.5300        0        0        0        0        0        0        0        0        0   9.7100
        0   2.0200   1.9800        0   9.4700        0        0        0        0        0        0
        0        0        0        0   7.8300        0        0        0        0   2.2900        0
        0        0        0        0  11.5700        0        0        0   2.1300        0        0
        0        0        0        0        0        0   7.5700        0        0        0        0
        0   5.7400        0        0        0        0   3.6600        0        0        0        0
        0        0        0        0        0  13.4800        0        0        0        0        0
        0        0   7.3900        0        0        0        0        0        0        0        0
        0        0        0        0        0        0        0        0        0        0        0

 Columns 12 through 15:

        0        0        0        0
   5.6200        0        0        0
        0        0  11.6200        0
        0        0        0        0
        0        0        0        0
        0  10.8300        0        0
   3.5000        0        0        0
        0        0        0        0
        0        0        0        0
        0        0        0        0
        0        0        0        0
        0        0        0   5.1400
        0        0        0        0
        0        0        0        0
   4.1300        0        0        0

        0   3.4600        0        0        0        0   1.9600        0        0        0
   3.7800        0   3.5800        0        0        0        0   1.8800        0        0
        0   3.2500        0   2.8700        0        0        0   2.3000        0        0
        0        0   2.4000        0   9.0500        0        0        0        0        0
        0        0        0   8.1700        0   6.5700        0   9.6600  10.8600   9.5800
        0        0        0        0   5.5700        0        0        0        0        0
   2.2200        0        0        0        0        0        0        0        0        0
        0   2.4000   2.8600        0   9.5500        0        0        0        0        0
```

3. Compute the shortest travel time using the Dijkstra algorithm from any origin to any destination of the network to build a "TT_OD" matrix for the 50 samples of the TT_link 3D-Matrix.

```
% Get the size of the matrix
[matrix_size, ~, num_samples] = size(TT_link);

% The number of nodes is the number of rows (or columns) in one of the 2D slices
num_nodes = matrix_size;

% Display the number of nodes
disp(['The number of nodes (bus stops) in the network is: ', num2str(num_nodes)]);
```

```
The number of nodes (bus stops) in the network is: 15
```

Work realized by : Yasmine DALY M2 SIA

```
function [shortestDist, routePath] = Dijkstra(netGraph, startPoint)
    totalNodes = size(netGraph, 1);
    shortestDist = inf(1, totalNodes);
    routePath = zeros(1, totalNodes);

    nodeChecked = false(1, totalNodes);
    shortestDist(startPoint) = 0;

    for idx = 1:totalNodes
        minNode = -1;
        for nodeIter = 1:totalNodes
            if ~nodeChecked(nodeIter) && (minNode == -1 || shortestDist(nodeIter) < shortestDist(minNode))
                minNode = nodeIter;
            end
        end

        nodeChecked(minNode) = true;

        for adjNode = 1:totalNodes
            if netGraph(minNode, adjNode) > 0
                potentialDist = shortestDist(minNode) + netGraph(minNode, adjNode);
                if potentialDist < shortestDist(adjNode)
                    shortestDist(adjNode) = potentialDist;
                    routePath(adjNode) = minNode;
                end
            end
        end
    end
end
```

Calculate TT_OD matrix: (ques2.m)

```
function TT_OD = ques2(TT_network)
    node_count = size(TT_link, 1);

    TT_OD = zeros(node_count, node_count, size(TT_link, 3));

    for sample_idx = 1:size(TT_link, 3)
        current_sample_net = squeeze(TT_link(:, :, sample_idx));

        for start_node = 1:node_count
            [shortestTimes, ~] = dijkstra(current_sample_net, start_node);
            TT_OD(start_node, :, sample_idx) = shortestTimes;
        end
    end
end
```

octaveProject.m

```
% Display a portion of the results for manual verification
origin_node = 1; % for example, node 1
sample_number = 1; % for example, sample 1

% Retrieve the shortest travel times from the origin to all destinations for the chosen sample
travel_times_from_origin = TT_OD(origin_node, :, sample_number);

% Display the results
fprintf('Shortest travel times from node %d in sample %d:\n', origin_node, sample_number);
for destination_node = 1:numel(travel_times_from_origin)
    if destination_node ~= origin_node % To avoid displaying the travel time from the node to itself
        fprintf('To node %d: %6.2f minutes\n', destination_node, travel_times_from_origin(destination_node));
    end
end
```

Work realized by : Yasmine DALY M2 SIA

```
>> octaveProject

Shortest travel times from node 1 in sample 1:
To node 2:    3.46 minutes
To node 3:    7.04 minutes
To node 4:    9.91 minutes
To node 5:   14.89 minutes
To node 6:   21.46 minutes
To node 7:    1.96 minutes
To node 8:    5.34 minutes
To node 9:   25.75 minutes
To node 10:   24.47 minutes
To node 11:    9.91 minutes
To node 12:    4.75 minutes
To node 13:   35.41 minutes
To node 14:   18.03 minutes
To node 15:    9.70 minutes
>> |
```

Save the TT_OD matrix to a file:

```
save('TT_OD.mat', 'TT_OD');
```

4. Compute all the key performance indicators given in the paper *(named COV, TT10, TT50, TT80, TT90, TT95, λ^{var}, λ^{dev} , UI_r, BI, PTI, MI, Pr(TTi>B+TT50) in the paper)*. Where TT80/ TT90, TTk… is the 80th / 90th / kth ... percentile, also known as percentile score or centile. A 5% percentile could be chosen to define the *freeflow Travel time*. To define the route length L_r let consider a 15km/h mean speed together with the free flow travel time.

- **COV**

```
% COV
averageTT_OD = mean(TT_OD,3);
COV = std(TT_OD, 0, 3) ./ averageTT_OD;
COV(isnan(COV)) = 0;
```

COV [15x15 double]

	1	2	3	4	5	6	7	
1	0	0.15331	0.11317	0.093339	0.10107	0.080935	0.13373	0.1
2	0.14286	0	0.13968	0.10555	0.12169	0.091318	0.093117	0.1
3	0.093294	0.13473	0	0.1396	0.10246	0.07701	0.072826	0.1
4	0.076826	0.099767	0.14402	0	0.15214	0.10382	0.063317	0.0
5	0.088609	0.10579	0.10848	0.12592	0	0.1398	0.075146	0.1
6	0.075148	0.083862	0.091325	0.10255	0.15827	0	0.06683	0.1
7	0.13291	0.11616	0.099027	0.084022	0.092826	0.075239	0	0.0
8	0.098406	0.14651	0.13599	0.092689	0.14909	0.10575	0.07202	0
9	0.089601	0.098739	0.099997	0.11006	0.13592	0.11289	0.081415	0.1
10	0.078778	0.086379	0.086507	0.093526	0.11916	0.096186	0.071385	0.0
11	0.10885	0.09098	0.079636	0.076742	0.081195	0.070342	0.12736	0.0
12	0.11172	0.14028	0.089274	0.075801	0.094605	0.079897	0.14758	0.0
13	0.076458	0.083799	0.084541	0.092065	0.11887	0.15554	0.070909	0.0
14	0.095605	0.11571	0.15176	0.12447	0.080072	0.065961	0.084832	0.1
15	0.090315	0.093481	0.072612	0.069284	0.077574	0.066499	0.1012	0.0

Work realized by : Yasmine DALY M2 SIA

- **Percentiles**

```
function [tt10,tt50, tt80, tt90, tt95] = percentiles(TT_OD)
    tt10 = prctile(TT_OD, 10, 3);
    tt50 = prctile(TT_OD, 50, 3);
    tt80 = prctile(TT_OD, 80, 3);
    tt90 = prctile(TT_OD, 90, 3);
    tt95 = prctile(TT_OD, 95, 3);

end
```

```
% Percentiles
[tt10, tt50, tt80, tt90, tt95] = percentiles(TT_OD);
```

Pr [15x15 double]

	1	2	3	4	5	6	7	
1	0	0.44	0.48	0.5	0.48	0.48	0.34	0.4
2	0.44	0	0.44	0.44	0.46	0.48	0.44	0.3
3	0.44	0.4	0	0.38	0.46	0.48	0.42	0.4
4	0.38	0.46	0.42	0	0.48	0.48	0.44	0.4
5	0.44	0.48	0.44	0.42	0	0.48	0.48	0.4
6	0.44	0.46	0.48	0.44	0.48	0	0.48	0.4
7	0.36	0.5	0.48	0.48	0.48	0.46	0	0.4
8	0.42	0.44	0.4	0.46	0.48	0.48	0.42	0
9	0.48	0.48	0.44	0.48	0.48	0.48	0.48	0.4
10	0.46	0.48	0.48	0.46	0.48	0.48	0.48	0.4
11	0.48	0.48	0.46	0.48	0.46	0.5	0.46	0.4
12	0.44	0.46	0.48	0.4	0.46	0.48	0.44	0.4
13	0.46	0.5	0.5	0.48	0.5	0.5	0.48	0.5
14	0.44	0.48	0.46	0.46	0.46	0.46	0.44	0.4
15	0.48	0.42	0.46	0.46	0.48	0.48	0.4	0.4

- **λvar, λdev and Undesirable Index of Reliability**

```
function [var_coeff, routeDist, skewness_coeff, ReliabilityIndex] = computeReliabilityMetrics(tt10, tt50, tt90

    average_speed = 15; % Average speed in km/h
    free_flow_time = tt95; % Using 95th percentile as free flow time
    routeDist = average_speed * mean(free_flow_time / 60); % Route distance calculation

    % Compute variability coefficient (lambda_var)
    var_coeff = (tt90 - tt10) / tt50;

    % Compute skewness coefficient (lambda_skew)
    skewness_coeff = (tt90 - tt50) / (tt50 - tt10);

    % Compute Unreliability Index (UIr)
    ReliabilityIndex = var_coeff * log(skewness_coeff) / routeDist;
end
```

Work realized by : Yasmine DALY M2 SIA

skewness_coeff [15x15 double]

	1	2	3	4	5	6	7	
1	1.2266	0.19412	0.69671	0.16027	-0.44398	-0.20604	-0.29972	-0.
2	0.056347	0.77236	0.20009	-0.014131	-0.16371	-0.13061	0.14579	-0.
3	-0.35353	0.70257	0.82349	0.05149	-0.32914	-0.16061	0.4922	-0.
4	-0.10416	0.80392	-0.50286	1.1612	0.034159	-0.11773	0.32602	-0.
5	-0.036097	0.2471	-1.2136	0.067874	0.94776	0.50622	0.26397	0.9
6	-0.56461	-0.58321	-0.18401	-0.014987	-0.86021	1.6035	0.056543	1.1
7	-0.00010028	0.6456	-0.37913	0.77978	0.18501	-0.31073	0.88155	-0.
8	0.37832	-0.24652	0.8823	-0.018365	-0.47866	-0.18971	0.0058692	0.2
9	1.0785	-0.23898	0.19977	-0.44515	0.18981	-0.41117	-0.070581	-0.
10	0.47766	-0.74805	0.55225	-0.36572	-0.1022	-0.16014	-0.066707	-0.
11	0.036545	-0.090439	-0.0027278	-0.30411	-0.30222	0.18437	-0.47982	0.6
12	0.52948	-0.22517	0.21112	-0.086787	-0.14189	0.33343	-0.98578	0.2
13	0.63364	0.082979	1.4762	-1.8203	-0.80837	0.19777	-1.0503	0.0
14	-0.28479	-0.32094	-0.081105	0.21605	-0.14964	0.30556	0.099947	0.4
15	-0.28936	0.23264	0.2889	-0.21897	-0.041551	-0.2617	0.58293	-0.

var_coeff [15x15 double]

	1	2	3	4	5	6	7	
1	0.48536	-0.20129	-0.145	0.12352	-0.15412	-0.019946	-0.17007	0.0
2	-0.39102	0.60135	-0.15404	0.11631	-0.15171	-0.051507	0.43838	0.0
3	-0.26106	0.096903	0.4699	-0.10776	-0.025545	-0.052806	0.37496	-0.
4	-0.087111	-0.074594	-0.1039	0.40201	-0.20323	-0.015065	0.23809	0.1
5	-0.10848	0.15591	0.12618	-0.13455	0.48103	-0.12229	0.093831	-0.
6	-0.10093	0.22479	-0.011623	0.021301	-0.10458	0.38973	0.073866	-0.
7	-0.11946	-0.024369	-0.18526	0.11088	-0.10902	-0.022074	0.52077	0.1
8	-0.097506	-0.053322	-0.13459	0.19068	-0.24428	-0.020591	0.22862	0.4
9	0.14395	-0.24912	0.31887	-0.16731	0.012173	-0.07666	-0.12641	-0.
10	0.054287	-0.23416	0.19037	-0.053454	-0.13057	0.05355	-0.10156	-0.
11	0.070942	-0.0843	0.086221	-0.038194	-0.04147	-0.039592	-0.26286	0.0
12	0.29195	-0.32871	-0.038064	0.11222	-0.13183	-0.067066	-0.2866	0.1
13	-0.29332	0.16231	0.023508	-0.088799	-0.01567	-0.17131	0.27298	-0.
14	0.11325	-0.039932	-0.24253	-0.0092063	-0.046874	0.044463	-0.12727	0.0
15	0.005676	-0.048667	-0.033302	0.075797	-0.15523	0.033642	0.043688	0.0

ReliabilityIndex [15x1 double]

	1
1	0.0019279 + 0…
2	-0.14911 - 0.…
3	-0.082261 + 0…
4	-0.043849 + 0…
5	-0.088241 + 0…
6	-0.048549 + 0…
7	-0.074445 + 0…
8	-0.07635 + 0.…
9	0.0024054 + 0…
10	0.021804 + 0.…
11	-0.016927 + 0…
12	0.0060454 + 0…
13	-0.069514 - 0…
14	-0.012878 + 0…
15	-0.043868 + 0…

- **Buffer Index**

```
% Buffer Index
BI = (tt95 - averageTT_OD) ./ averageTT_OD;
```

- **Planning Time Index**

```
% Planning Time Index
tt_freeflow = tt95;
PTI = tt95 / tt_freeflow;
```

- **Misery Index**

```
% Misery Index
TT_OD_TT80 = TT_OD > tt80;
TT_OD_filtered = TT_OD .* TT_OD_TT80;
MI = (mean(TT_OD_filtered, 3) - averageTT_OD) / averageTT_OD;
```

- **Probability**

```
% Probability
Pr = sum(TT_OD > (0.1 + tt50), 3) / size(TT_OD, 3);
```

5. Comment key performance indicators results. Which index will you use if you were a regular user compared to an occasional user.

When analyzing the key performance indicators for travel time reliability, such as the Coefficient of

Variation (COV), Buffer Index (BI), Planning Time Index (PTI), Misery Index (MI), and Probability (Pr), along with percentiles like TT10, TT50, TT80, TT90, and TT95, it's crucial to consider the perspective of different types of users. As a regular user of the transport network, compared to an occasional user, my focus on specific indices would vary based on my needs.

1. Percentiles: The range of travel times under various conditions is given by these values. TT50 is particularly relevant to me as a regular user since it indicates the median travel time I would typically expect. However, as an occasional user, I would look at TT90 or TT95 to understand the longer travel times for worst-case scenarios.

2. Coefficient of Variation (COV): This index shows the level of variability in travel times. As a regular commuter, I prefer a lower COV for consistent travel times. The occasional user might not prioritize COV unless they are on a tight schedule.

3. Buffer Index (BI): BI tells me how much extra time I should add to the median travel time to ensure on-time arrival for most of my trips. This is incredibly useful for me, especially when I have important commitments.

4. Planning Time Index (PTI): PTI compares the worst-case travel time to the normal travel time, indicating how much extra time I should plan for my journey to ensure on-time arrival. As a regular user, I find PTI useful on days when my schedule is not flexible. For an occasional user, PTI is valuable for understanding how much buffer time to add for critical trips.

5. Misery Index (MI): This index indicates the severity of the worst-case scenarios compared to typical travel times. While it might be less relevant for my daily commutes, as a regular user, I consider it for special occasions. For occasional users, it's helpful to understand the extreme cases, particularly if traveling for important events.

6. Probability (Pr): Pr gives the likelihood of experiencing travel times longer than a certain threshold. For me, it serves as a tool for risk assessment over time. Occasional users can use it to gauge the chances of delays for their specific travel plans.

In conclusion, as a regular user, my focus is on TT50, COV, and BI for consistent and predictable travel time planning, with PTI also being a consideration on critical days. In contrast, as an occasional user, I would prioritize understanding the range of possible delays

through PTI, TT90/TT95, and MI, which provide insights into the worst-case scenarios. Additionally, BI and Pr would be important for determining the necessary buffer time. The choice between these indices depends on whether my priority is consistent travel times or being prepared for extremes and my familiarity with the usual traffic conditions. Regular users benefit from indices that offer consistency and predictability, while occasional users might focus more on understanding the likelihood and extent of potential delays.

Part 2: Value of time

1. Considering values of time (VOT) given on Table 1 and using the VOT for the case *"without a specific purpose"* in the Île-de-France area, calculate the maximum and minimum time-savings benefits in the worst and best situation in the samples of the TT_OD matrix compared to the initial conditions.

Work realized by : Yasmine DALY M2 SIA

```
% Value of Time
vot = 11.4 / 60;

% Best Benefits Calculation
best_benefits = (tt50 - min(TT_OD, [], 3)) * vot;

% Worst Benefits Calculation
worst_benefits = (tt50 - max(TT_OD, [], 3)) * vot;

% Display the results
disp('Best Time-Savings Benefits (in euros):');
disp(best_benefits);

disp('Worst Time-Savings Benefits (in euros):');
disp(worst_benefits);
```

```
Best Time-Savings Benefits (in euros):
 Columns 1 through 12:

        0   0.1425   0.2964   0.3486   0.6242   0.6688   0.0941   0.1929   0.8427   0.9215   0.4703   0.2432
   0.1653        0   0.1567   0.2185   0.4826   0.5747   0.2593   0.1396   0.7913   0.8930   0.5995   0.2175
   0.2641   0.1615        0   0.0884   0.4209   0.5016   0.3800   0.1169   0.8303   0.5909   0.5425   0.3031
   0.3790   0.2185   0.1168        0   0.4712   0.6508   0.4712   0.1700   0.8788   0.7467   0.4950   0.4019
   0.5434   0.4266   0.3306   0.3344        0   0.3154   0.4636   0.4703   0.5671   0.4132   0.6004   0.5320
   0.6460   0.5045   0.4741   0.6004   0.2024        0   0.6412   0.6251   0.7125   0.5035   0.9110   0.7239
   0.1017   0.2185   0.3496   0.3467   0.7087   0.6764        0   0.2423   0.7524   0.8208   0.4266   0.1558
   0.2708   0.0921   0.1264   0.1919   0.4351   0.6318   0.3116        0   0.7581   0.8987   0.5804   0.2584
   0.9111   0.7334   0.6498   0.6679   0.4978   0.6887   0.8265   0.8417        0   0.0921   0.9301   0.8664
   0.7381   0.6318   0.4531   0.4522   0.3467   0.5719   0.7163   0.6669   0.1045        0   0.9196   0.7819
   0.5216   0.6717   0.7011   0.7562   1.1980   1.0098   0.4883   0.7420   1.1048   1.2255        0   0.5206
   0.2527   0.2394   0.3382   0.3838   0.6565   0.7457   0.1282   0.2195   1.0042   1.0802   0.5064        0
   1.0982   1.1219   0.8778   0.7771   0.6812   0.5814   1.1457   1.1580   0.7895   1.0127   0.8274   1.2302
   0.5652   0.4370   0.4740   0.5263   0.6745   0.6013   0.6422   0.4835   0.6194   0.8531   0.8046   0.5462
   0.4085   0.3914   0.4465   0.5159   0.5235   0.6754   0.3696   0.4246   0.9225   1.0051   0.6137   0.2898

 Columns 13 through 15:

   0.8607   0.7533   0.4057
   0.8455   0.6498   0.3781
   0.8502   0.5263   0.4617
   0.9224   0.5501   0.5719
   0.7971   0.5900   0.5862
   0.4655   0.8863   0.8066
   0.8445   0.8398   0.3477
   0.8892   0.6071   0.4066
   0.7999   0.8246   0.8911
   0.7752   0.7866   0.9272
   1.0450   1.0212   0.6964
   0.7296   0.7049   0.1919
        0   1.1239   1.3348
```

```
Worst Time-Savings Benefits (in euros):
 Columns 1 through 12:

        0  -0.1919  -0.2793  -0.2689  -0.6355  -0.6403  -0.1093  -0.2442  -0.7229  -0.8721  -0.5329  -0.2413
  -0.1558        0  -0.1625  -0.2375  -0.4997  -0.5729  -0.1682  -0.0846  -0.7704  -0.9044  -0.5558  -0.2195
  -0.2546  -0.1406        0  -0.1264  -0.4664  -0.4940  -0.2128  -0.0921  -0.4503  -0.6916  -0.6280  -0.3050
  -0.2860  -0.1919  -0.1073        0  -0.3078  -0.5482  -0.3211  -0.1776  -0.5690  -0.6707  -0.6716  -0.3600
  -0.6403  -0.4588  -0.5396  -0.4275        0  -0.2508  -0.6365  -0.3923  -0.3144  -0.5463  -0.7410  -0.5890
  -0.6175  -0.6640  -0.5614  -0.5130  -0.3410        0  -0.5862  -0.6194  -0.5928  -0.8892  -0.8008  -0.8227
  -0.1187  -0.2546  -0.3971  -0.3410  -0.6764  -0.7049        0  -0.3486  -0.8550  -1.0222  -0.4323  -0.1748
  -0.2137  -0.1264  -0.0903  -0.1862  -0.4636  -0.6146  -0.2812        0  -0.7676  -0.7904  -0.4969  -0.3154
  -0.7571  -0.7011  -0.8702  -0.7628  -0.4104  -0.6850  -0.7657  -0.7467        0  -0.1245  -0.8331  -0.8550
  -0.8275  -0.8446  -1.0859  -0.9728  -0.6032  -0.8531  -0.8227  -0.8474  -0.1121        0  -0.8968  -0.8901
  -0.4285  -0.4760  -0.4180  -0.4674  -0.8521  -0.8028  -0.3686  -0.4398  -0.9566  -1.1932        0  -0.4351
  -0.2736  -0.1938  -0.3344  -0.2584  -0.6450  -0.6698  -0.2080  -0.2574  -0.7248  -0.8749  -0.6013        0
  -0.9880  -0.9035  -0.8265  -0.8398  -0.7723  -0.5472  -1.0146  -1.0022  -0.8085  -0.9557  -1.3215  -1.1828
  -0.6184  -0.5491  -0.4304  -0.5187  -0.6840  -0.7476  -0.6004  -0.5064  -0.8664  -0.9063  -0.7267  -0.5957
  -0.4465  -0.3439  -0.4180  -0.4703  -0.9054  -0.8521  -0.3334  -0.4284  -0.9338  -1.0450  -0.6384  -0.1586

 Columns 13 through 15:

  -0.6783  -0.5900  -0.4322
  -0.6251  -0.5168  -0.3097
  -0.6204  -0.3610  -0.3781
  -0.8237  -0.4304  -0.4332
  -0.6375  -0.9434  -0.6337
  -0.5643  -0.8977  -0.7457
  -0.6717  -0.6118  -0.3933
  -0.7315  -0.4474  -0.2907
  -1.0735  -1.2559  -0.8037
  -1.0906  -1.4193  -0.9766
  -0.6213  -0.7382  -0.5196
  -0.8303  -0.5890  -0.2603
        0  -1.1789  -1.0117
  -1.1524        0  -0.6441
```

2. Analyze and comment results linked to the monetary waste of time due to reliability of travel time.

Analyzing the key performance indicators (KPIs) from the data, we can observe various aspects of travel time reliability and variability in the network. These indicators offer insights into different scenarios a traveler might encounter, be it as a regular or an occasional user.

Best Time-Savings Benefits: This indicator shows the potential time saved during the best-case scenario travel times compared to the median travel times. For a regular user, who might be more familiar with the usual travel times and more sensitive to time savings, this indicator is particularly relevant.

Worst Time-Savings Benefits: Contrarily, this indicator reflects the potential time lost during the worst-case scenario travel times compared to the median travel times. Occasional users, who might not have a clear understanding of typical travel times and could be more affected by unexpected delays, would find this indicator more relevant.

Selection of Indicators Based on User Type:

Regular User: They would likely be more interested in consistent travel times and minor variations. Hence, indicators like the Coefficient of Variation (COV), which measures relative variability, and the Buffer Index (BI), which quantifies how much extra time should be planned to ensure on-time arrival, would be highly pertinent. Regular users might also consider the Misery Index (MI) to understand the frequency of unusually long travel times.

Occasional User: These users might be more concerned about extreme cases and overall reliability. Indicators like the worst time-savings benefits and Probability (Pr) indicator, which shows the likelihood of travel times exceeding a certain threshold, would be crucial. Additionally, the Planning Time Index (PTI) and the Undesirable Index of Reliability (UIr) would also be useful to understand the extent of potential delays.

In summary, while a regular user might focus more on consistency and minor variations, an occasional user would be more concerned with extreme cases and overall network reliability. Understanding these indicators can significantly enhance travel planning and time management for different types of users.

Table 1: Value of Time in €/hour in France in 2015
(extracted from: Chiffres clés des transports - Édition 2021, Ministères Écologie Énergie Territoires)

Trip purpose	Average in France	Ile-de-France only *(Paris and suburbs)*
Professional	18.6	23.7
Home-to- work/school/university	10.6	13.4
Others *(shopping, care, visit, leisure…)*	7.2	9.3
Without details of the trip purpose	8.4	11.4

Bibliography
[1] *Bhouri N. and Kauppila, J. "Managing Highways for Better Reliability – Assessing Reliability Benefits of Ramp Metering". Transportation Research Record, Journal of the Transp. Research Board N°2229. Pp1-7. DOI : 10.3141 /2229-01, 2011.*
Available at ***https://hal.science/hal-01483591/document***

Note:

I would like to extend my heartfelt gratitude to my esteemed professors, François Olivier and Neila Bhouri, for their invaluable guidance and support throughout the course of this project. Their expertise and insights have been instrumental in shaping this work.

For those interested in exploring the methodologies and analyses further, the source code used in this study can be accessed at https://github.com/yasminedaly/Networks-Reliability This repository contains all the scripts and data used, allowing for a deeper dive into the research and findings presented in this report.

Once again, my sincerest thanks to Professors Olivier and Bhouri for their mentorship and to all who have contributed to this endeavor.