WHAT'S NEW

Wasatch Front Travel Demand Model

Version 9.1.0

WFRC / MAG

October 31, 2024

Table of Contents

List of Figures

List of Tables

No table of figures entries found.

1 Input Updates

1.1 Highway Network

The highway network was updated minorly to accommodate projects from WFRC's adopted Amendment #2 and MAG's draft Amendment #2 of the 2023-2050 RTP. Figure 1.1 shows the areas that were updated for Amendment #2.

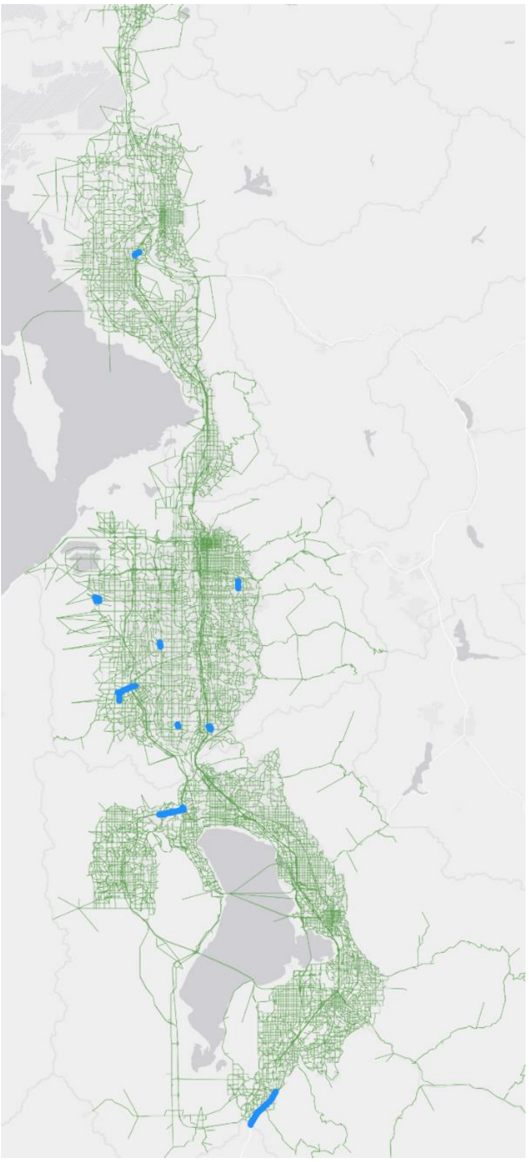


Figure 1.1 Updated Lane and Operational Lane Fields (blue)

The **SEGID** field along with other fields were updated as part of regular network maintenance (See Figure 1.2).

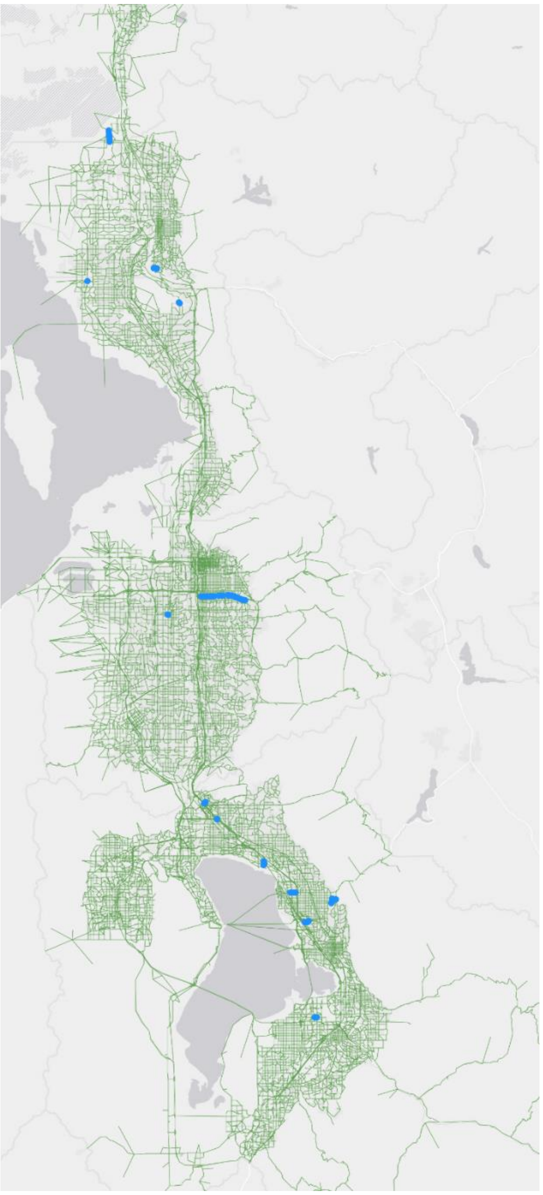


Figure 1.2 Updated SEGID Field (blue)

1.2Transit Networks

The transit networks were updated to reflect changes in the highway network.

OGX and MVX routes were updated to better align with current conditions and assumptions.

1.3 Segment Shapefile

A **DIRECTION** field was added onto the segment shapefile where each segment was assigned NB/SB for a segment in the northbound or southbound direction, and EB/WB for a segment in the eastbound or westbound direction to support the segment-based link direction calculation (See Section 4.2.1). The Jupyter Notebooks within the *"1_Inputs/6_Segment/_Segment Processing Tools"* folder were updated to account for the new **DIRECTION** field.

A new folder called "*Corridors*" was added to the "*1_Inputs/6_Segment*" directory. This folder includes various shapefiles and geojsons files describing corridors (highway, transit, freight, etc.) to support data aggregation and visualize in the vizTool.

1.4 TAZ Shapefile

The "*TAZ.shp*" was updated by renumbering the small districts as well as including two new fields: **PLANAREA** and **CITYGRP**. **PLANAREA** describes which modeling region the TAZ falls in and **CITYGRP** represents city groups (a new geographic definition to describe groupings of cities).

The renumbering of the small districts was also reflected in the "*Small_Districts.shp*". The small district geographies were also updated.

2 Python Updates

2.1 Python Environment

A complete Python environment is included within the model to better support the execution of Python scripts used by the model. The environment is self-contained, meaning it contains all necessary Python packages and dependencies, eliminating the need for external installations or configuration. The environment is named *"py-tdm-env"* and is located in the *"2_ModelScripts/_Python"* directory.

A list of the main Python libraries included in the environment is given below. This is not an exhaustive list.

- **dbfread**
- **filelock**
- **folium**
- **geopandas**
- **ipykernel**
- **ipywidgets**
- **json**
- **jupyter_server**
- **matplotlib**
- **numpy**
- **openpyxl**
- **pandas**
- **shapely**
- **yaml**

A full list of the Python libraries includes can be found in *"2_ModelScripts/_Python/py-tdm-env/Lib/site-packages"*.

The Python environment also includes packages for a Jupyter Kernel to help run Jupyter Notebooks within the model. For more direction on setting up the *"py-tdm-env"* to be used in Jupyter Notebooks, follow the instructions here.

2.2 "ip_UpdateNetwork_WalkBuffers.py"

In previous model versions, Python was called in the "*0_Update_TAZID_Distance.s*", "*1_NetProcessor.s*", and "*6_UpdateWalkBuffer.s*" scripts to perform geoprocessing tasks using the **arcpy** library. In version 9.1.0, the geoprocessing tasks were combined into one script *"ip_UpdateNetwork_WalkBuffers.py"* and the **arcpy** library was replaced with the **geopandas** library.

The *"ip_UpdateNetwork_WalkBuffers.py"* script creates files used to update the following fields on the highway network:

- **DISTANCE**

- **DIRECTION** (further details on the updates to the **DIRECTION** field can be found in Section 4.2.1)
- **TAZID** (links & nodes)
- **HOT_ZONEID** (links & nodes)

2.3 “ip_FolderSetup.py”

Scenario folders are now created with Python in the “*ip_FolderSetup.py*” script instead of with a DOS batch file in the “*0_FolderSetup.s*” script.

2.4 “py-vizTool” Folder

A new folder called “*py-vizTool*” was added to the “*2_ModelScripts/_Python*” directory. This folder houses the following four scripts that support the vizTool.

- “*vt_CompileJson.py*”: This script converts model outputs (.csv extension) to vizTool inputs (.json extension). It is called at the end of various scripts throughout the model. Section 3 outlines which scripts.
- “*_json_scripts.py*”: This script contains the functions for the “*vt_CompileJson.py*” script.
- “*vt_CreateGeoJsons.py*”: This script converts model input shapefiles to geojson format. It also contains logic that creates a project corridors geojson and transit stops geojson.
- “*vt_CreateScnJson.py”:* This script converts scenario attributes to a json file used to configure the vizTool data.

In addition, the “*py-vizTool*” folder contains a “*configs*” folder that contains multiple configuration json files that describe how to configure the vizTool inputs.

2.5 Other Python Updates

A new folder labeled “*_source - py script development*” exist within the “*2_ModelScripts/_Python*” directory for the purpose of double checking the development of Python scripts within a Jupyter Notebook environment.

3 Model Updates to Support vizTool

3.1vizTool Source Code and Setup

The vizTool is a tool designed to visualize travel demand model The vizTool source code is located in the "2_ModelScripts/_CopyToFolders/vizTool" folder. At the beginning of a model run, the "*vizTool*" folder is copied to the "*Scenarios/.vizTool*" directory (if it does not already exist in that location) via the "*2_vizToolSetup.s*" script. As the model runs, outputs are converted to vizTool inputs and placed in the "*Scenarios/.vizTool*" directory. This ensures that the "*.vizTool*" folder remains a standalone application allowing users to open it via localhost or host it on their own server.

A parameter called **Run_vizTool** was added to the "*0GeneralParameters.block*" file that when set to 0 will avoid running all Python scripts used to prepare data for the vizTool as well as prevent the tool from opening at the end of the model run. By default, this parameter is set to 1 to support the vizTool implementation.

Further information on the vizTool can be found at the following links:

- vizTool User Guide
- vizTool Data Dictionary
- vizTool Development Details
- vizTool Code Documentation
- vizTool source code GitHub repository

3.2 Expanded Socioeconomic File

An expanded socioeconomic file, *"expanded_SE_File_{RID}.csv"*, that includes total workers and number of household vehicles was added as a new model output. The "*2_HHDisaggregation.s*" and "*3_AutoOwnerhsip.s*" scripts in the "*2_ModelScripts/1_HHDisag_AutoOwn*" folder were adjusted accordingly.

3.3 Mode Choice Reports and Script Updates

The following scripts within "*2_ModelScripts/4_ModeChoice*" were either updated or created to support the vizTool:

"12_EstimateHBSchModeShare.s"* and *"13_vizTool_TripsByMode.s"

- Script *"11-13_MC_HBW_HBO_NHB_HBC.s"* was renamed to *"11_MC_HBW_HBO_NHB_HBC.s"*.
- *"12_EstimateHBSchModeShare.s"* was added to calculate two new detailed trip matrices that better summarize school trip data.
- The *"13_vizTool_TripsByMode.s"* script was added to summarize trip data by mode at the TAZ level. Data varies by period, purpose, and direction (production/attraction).

"17_BoardingsReport.s"

- This script was restructured and updated to calculate rider statistics in addition to boardings and alightings. Summaries are reported at the node, link, and segment levels by mode, hierarchical mode, and period. This new report takes the place of the previous mode choice boarding summary reports. The route and mode level summary for boardings and alightings is still output.

"18_SumToDistricts_FinalTripTables.s"

- This script was updated to create a new output summarized at the small district level. In addition, an error was fixed to include BRT records in all summaries.

3.4 Highway Assignment Reports and Script Updates

The following scripts within "*2_ModelScripts/5_AssignHwy*" were either updated or created to support the vizTool:

"01_Convert_PA_to_OD"* and *"02_Assign_AM_MD_PM_EV"

- The logic in these scripts (along with their supporting block files) were updated to output person trips instead of vehicle trips. The output matrices include more detail on school trips. The output matrices also specify direction in whether they go from the production zone to the attraction zone or vice versa to support TAZ-based metric calculations.

"06_SegmentSummary.s"

- The logic in this script was rewritten to better calculate highway statistics such as Vehicle Miles Traveled (VMT), Vehicle Hours Traveled (VHT), Vehicle Hours Delay (VHD), and Volume-Capacity (VC) Ratio at a wider breadth with data at the total, passenger car, medium truck, and heavy truck classifications. Minor logic errors were also corrected.

"09_TazBasedMetrics.s"

- This is a new script that summarizes final network skims and origin destination tables to calculate metrics at the TAZ level. Such metrics include Person Miles Traveled (PMT), Person Hours Traveled (PHT), Person Hours Delay (PHD), Vehicle Miles Traveled (VMT), Vehicle Hours Traveled (VHT), and Vehicle Hours Delay (VHD). These metrics are calculated with the congested network as well as the free flow.
- Note that the VMT, VHT, and VHD output from this script represent the metrics of those trips that are produced/attracted for a specific TAZ whereas the VMT, VHT, and VHD output from the *"06_SegmentSummary.s"* script represent the metrics of those trips that travel along the segments within a specific TAZ.

3.5 Log Files

Version 9.1.0 adds the following log reports to the *"_Log"* folder to support the vizTool:

- *"py_LogFile":* Files with this prefix store the logging statements associated with Python scripts that are run as part of the model.
- *"py_Variables"*: Files with this prefix store parameter type variables in a text file to be read in for Python scripts.
- *"py_Debug"*: Files with this prefix exist to show model appliers potential areas to debug within the model.

3.6 Opening the vizTool

After each model run, the vizTool application opens by default via localhost on the computer's default internet browser. In addition, the *"_OpenVizTool.bat"* batch script is created and placed in the specific scenario directory. The batch script provides an easy way to re-open the vizTool.

Users may also navigate to the *"_start-server-static.py"* script in the *"Scenarios/.vizTool"* directory and run that script to open the vizTool. (Note that users must have Python set up on their machine.)

4 Other Model Updates

4.1 Provo Airport

Version 9.1.0 inserts travel to/from the Provo Airport (PVU) by updating the *"2_ModelScripts/0_InputProcessing/d_TripTable/1_TripTable.s"* and *"2_ModelScripts/2_TripGen/1_TripGen.s"* scripts. The *"0GeneralParameters.block"* file was also updated to include the **PVU** zone number parameter.

The *"TripTableControlTotal.csv"* file, located in the *"1_Inputs/0_GlobalData/0_TripTables"* folder, was expanded to include trip totals by year for the Provo Airport.

Figure 4.1 shows the calculated control total for the number of trips attracted to the Provo Airport compared to those attracted to the Salt Lake City Airport (SLC). The number of trips attracted to the Provo Airport were estimated using a ratio calculated from the SLC control totals as well as data from the Federal Aviation Administration (FAA).

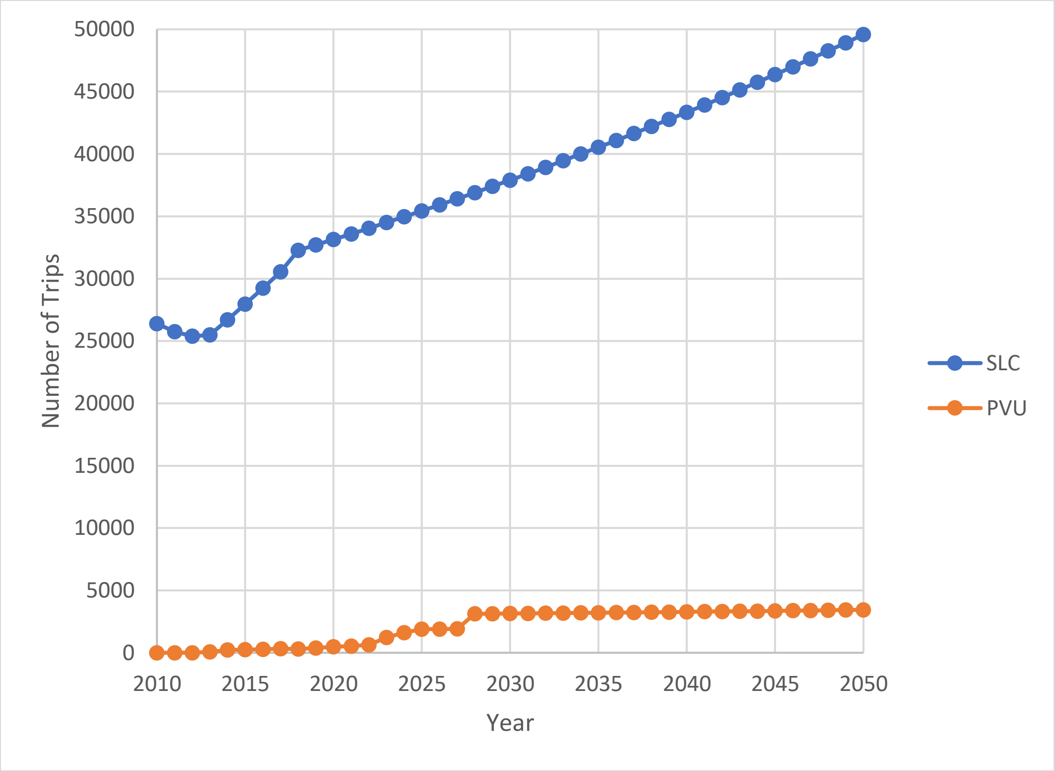


Figure 4.1 Number of Trips by Airport and Year

The *"BaseDistribution.csv"* file, located in the *"1_Inputs/0_GlobalData/0_TripTables"* folder, was also expanded to include trip totals produced by zone for the Provo Airport. Figure 4.2 shows the base distribution of trips going to/from the Provo airport. This distribution was calculated using trip data from StreetLight.

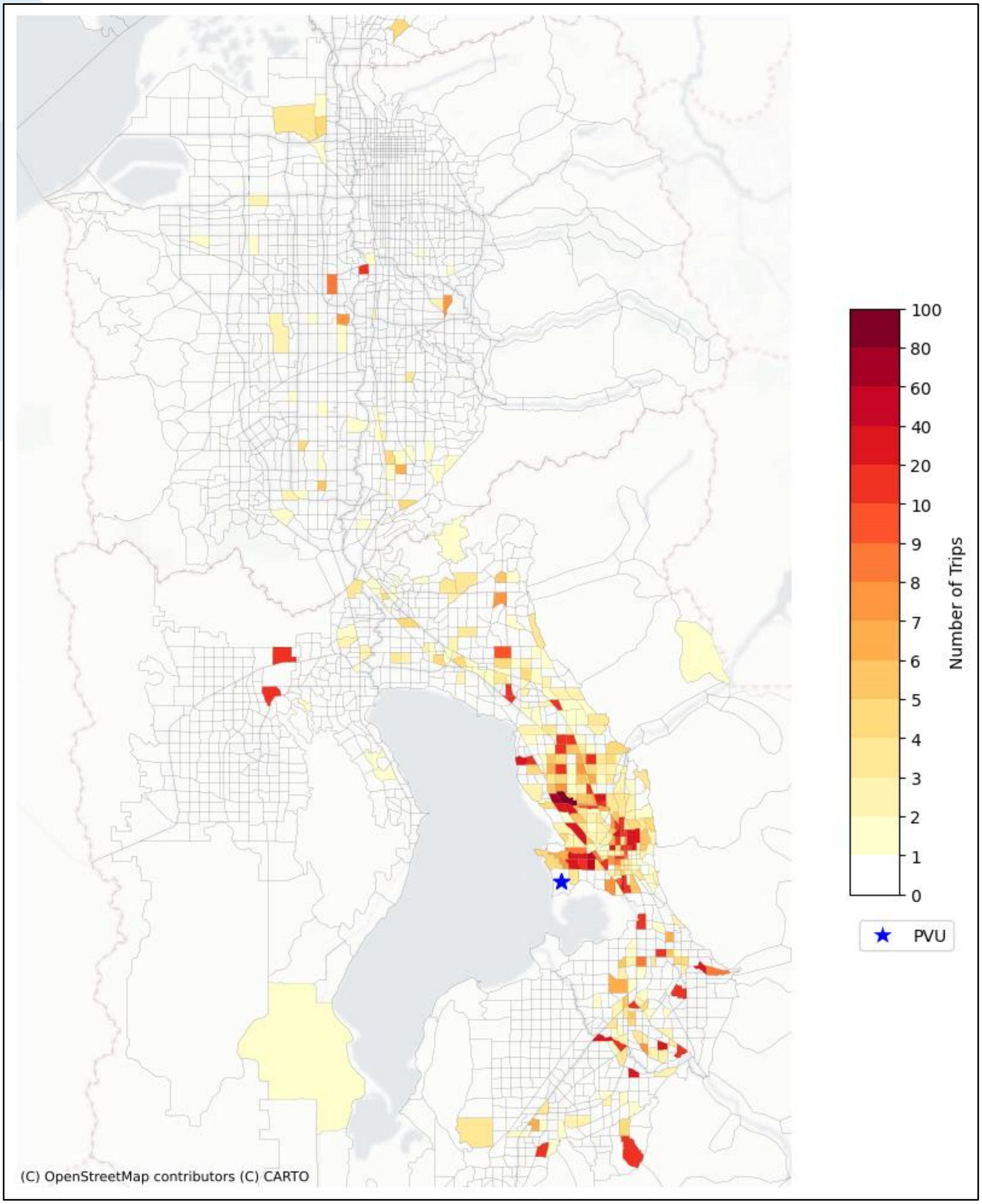


Figure 4.2 Base Trip Distribution by Zone for PVU

4.2 “1_NetProcessor” Script

The *“1_NetProcessor.s”* script was adjusted to accommodate new Python scripts, segment direction updates, reliability lane updates, and other bug fixes.

4.2.1 Segment Direction

The link direction logic was updated to align with segment direction. All links that fall within a segment adopt the cardinal direction of the overarching segment. For example, in Figure 4.3 the segment (orange) has a northbound/southbound direction. All model links (blue) associated with this segment will also have a northbound/southbound despite several links within the segment being oriented more east/west.

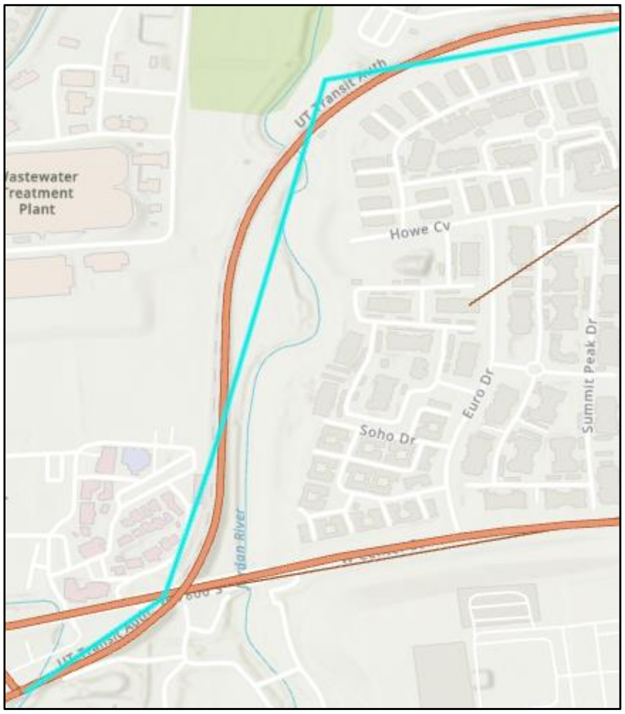


Figure 4.3 Segment and Link Direction Alignment

The segment direction code was added to the *“ip_UpdateNetwork_WalkBuffers.py”* script where it creates a temp scenario network with the **DIRECTION** field calculated. The temp scenario network is then read in for reference downstream.

4.2.2 Reliability Lanes

Reliability lane capacity calculations were moved to the *“1_NetProcessor.s”* script. The link level outputs were updated to include the **RelCap1Hr** field as well as fields for period specific capacities. A new parameter **Rel_LN_Toggle** was added to the *“0GeneralParameters.block”* file that acts as a switch to allow the user to determine if using reversible lanes adds lanes for the peak direction or repurposes off-peak direction lanes. Downstream scripts were updated to reflect these changes.

4.2.3 Bug Fixes

The "*1_NetProcessor*" script was updated to fix minor errors in the following calculations:

- Charge point code was updated to ensure that links were tagged with the correct charge point ID.
- Fixed a minor bug in the general purpose ramp ID classification.
- FT 39 was removed from the HOT speed calculation.

4.3 Other Minor Updates

Various inputs and outputs were renamed to comply with the new model file naming convention determined by the Interagency Modeling Technical Committee (IMTC).

5 Compare Model Results

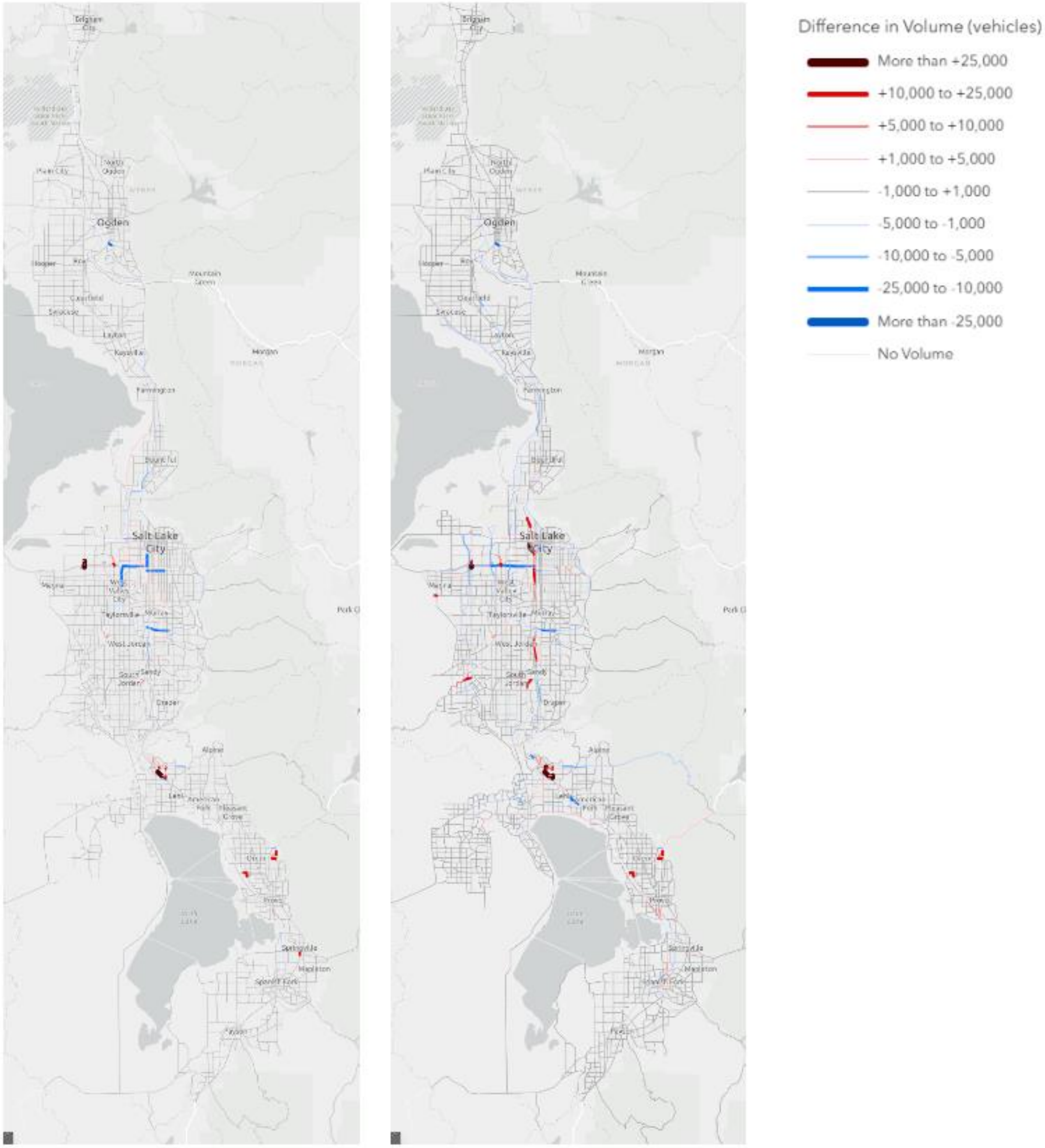
This section compares the model results between version 9.1.0 and version 9.0.2.

5.1Road Volume Comparisons

The comparison between daily volumes at the segment level can be found in Figure 5.1 for 2019 and 2050. Decreases in volume in version 9.1.0 compared to version 9.0.2 are shown in blue, while increases are shown in red. Figure 5.2 shows a similar comparison, displaying medium plus heavy truck volumes.

For 2019, there are a few volume differences for all vehicles and trucks due to segment fixes. Other differences are negligible.

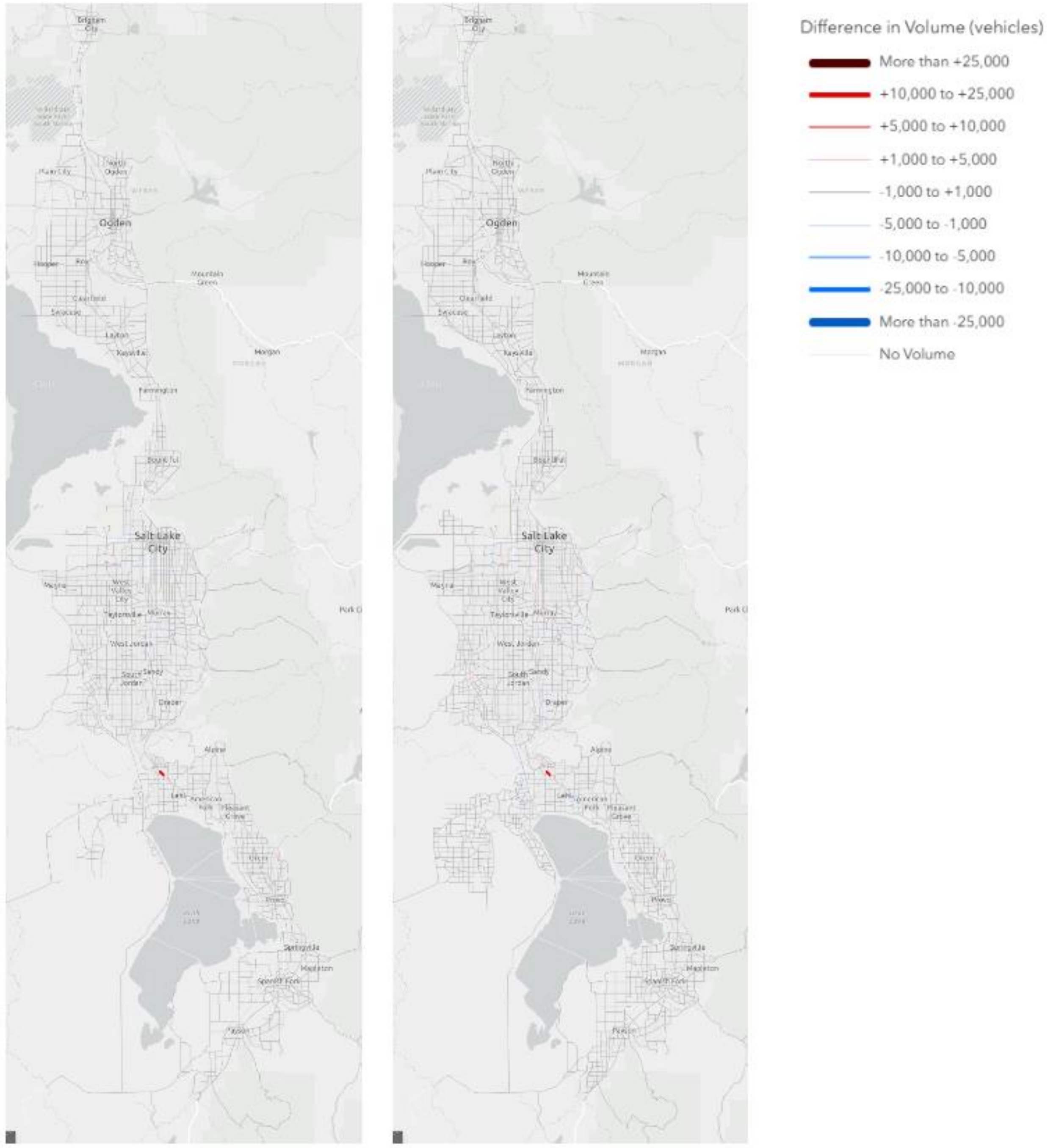
For 2050, there are a few volume differences for all vehicles and trucks due to segment fixes and project updates. Other differences are negligible.



(a) 2019

(b) 2050 Fiscally Constrained

Figure 5.1 Daily Volume Comparison – All Vehicles



(a) 2019

(b) 2050 Fiscally Constrained

Figure 5.2 Daily Volume Comparison – Medium+Heavy Truck

5.2 Transit Comparisons

Version 9.1.0 showed a slight increase in transit trips compared to version 9.0.2 in future years (see Figure 5.3 through Figure 5.9). The total transit trips in 2050 for version 9.1.0 is 342,000 daily trips compared to the version 9.0.2 model that showed 337,000 daily trips, which equates to 1.5% more trips.

Commuter Rail saw the greatest increase in trips, likely due to project updates. Express Bus saw a slight decrease in future trips, most likely due to the increase in trips to Commuter Rail since they compete for trips in similar markets. BRT, Core Route, and Local Bus trips remained relatively unchanged.

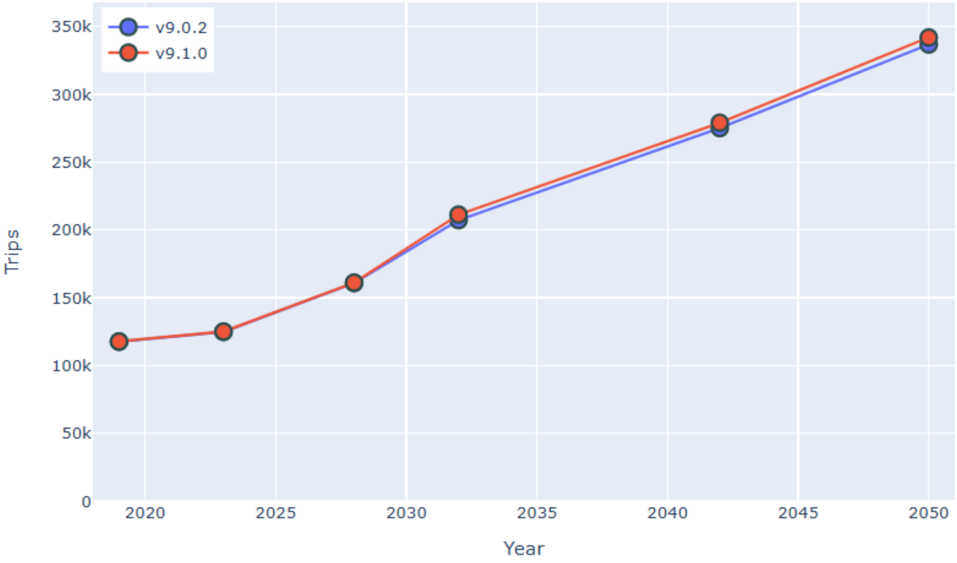


Figure 5.3 Daily Transit Trips - All Modes

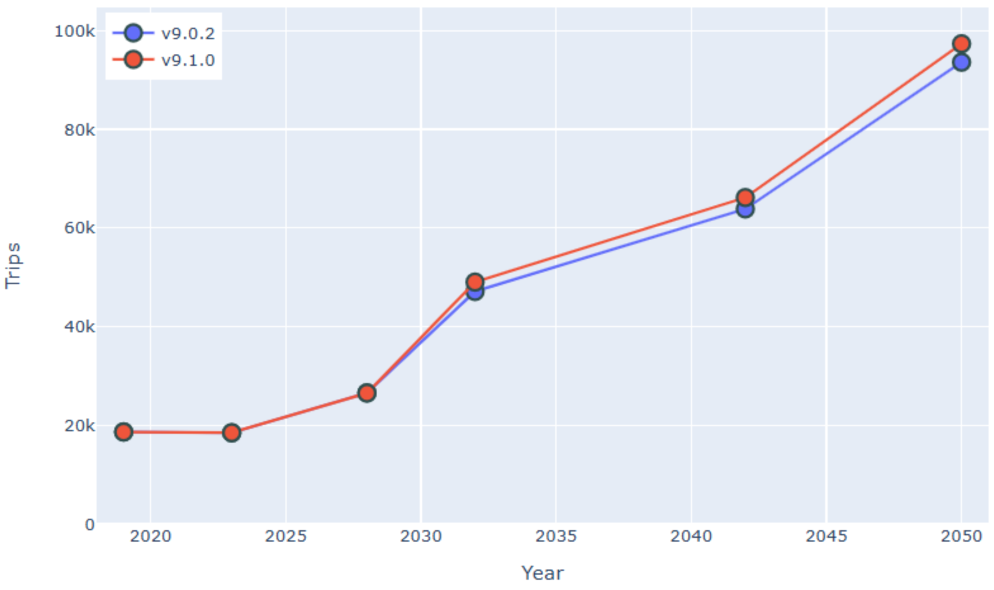


Figure 5.4 Daily Transit Trips – Commuter Rail

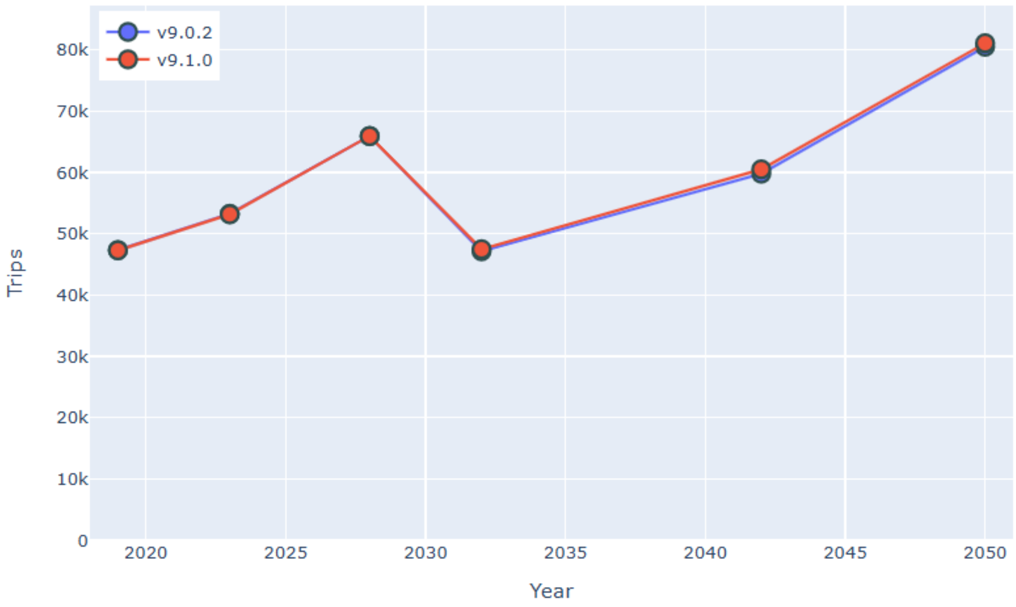


Figure 5.5 Daily Transit Trips – Light Rail

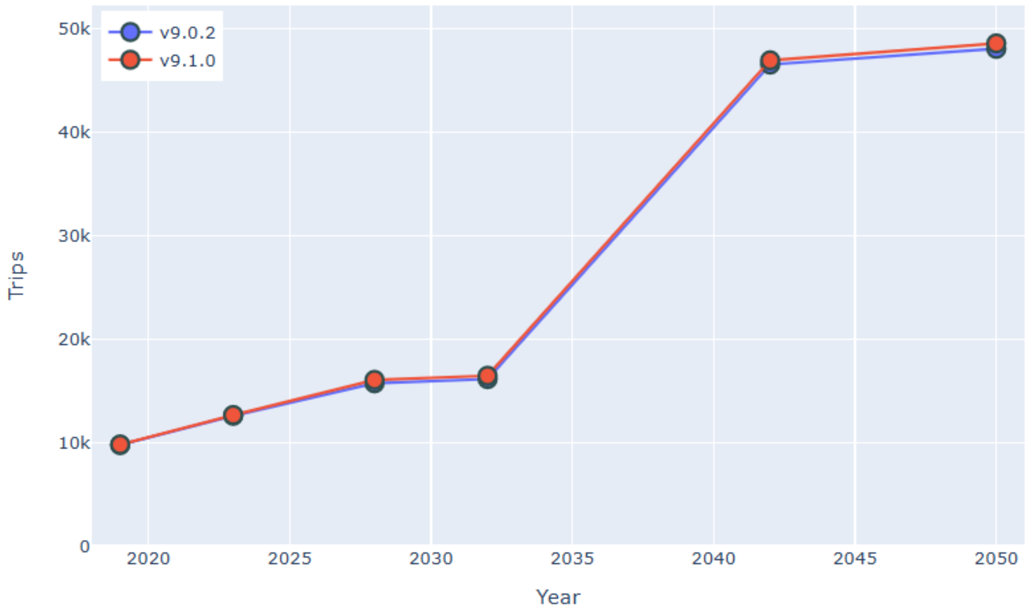


Figure 5.6 Daily Transit Trips - Bus Rapid Transit

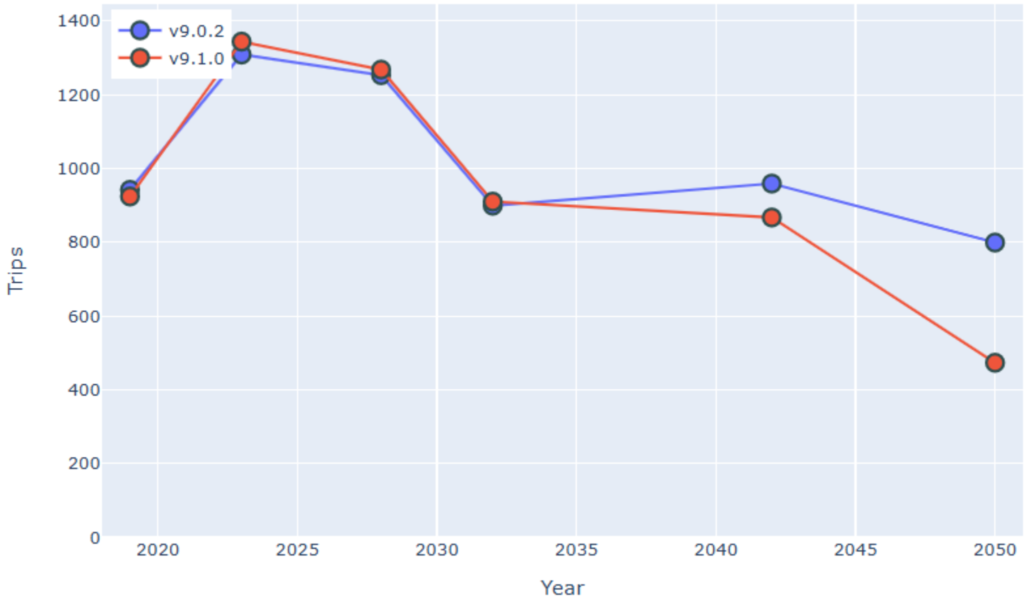


Figure 5.7 Daily Transit Trips - Express Bus

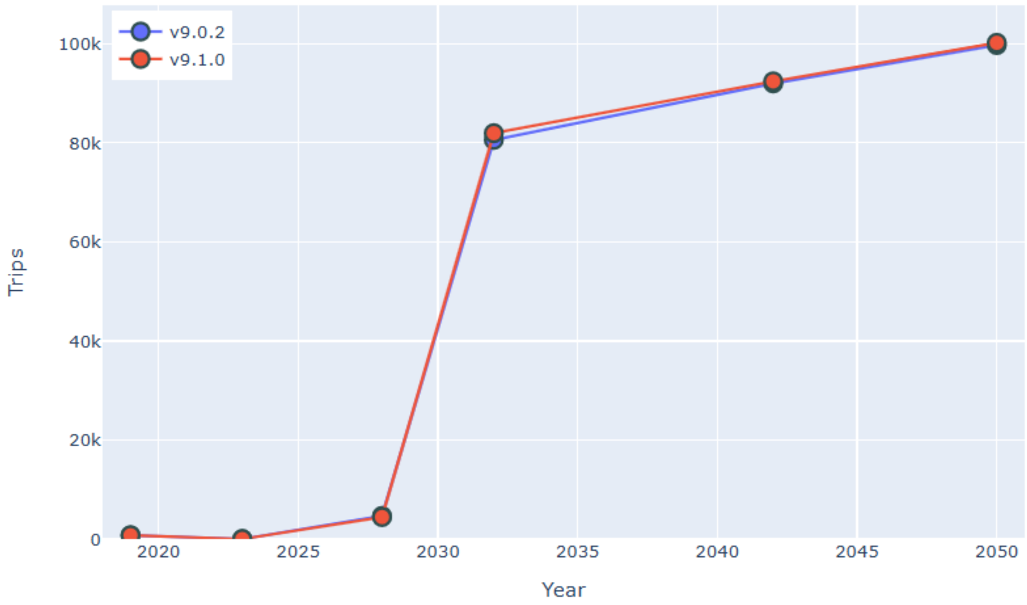


Figure 5.8 Daily Transit Trips - Core Bus

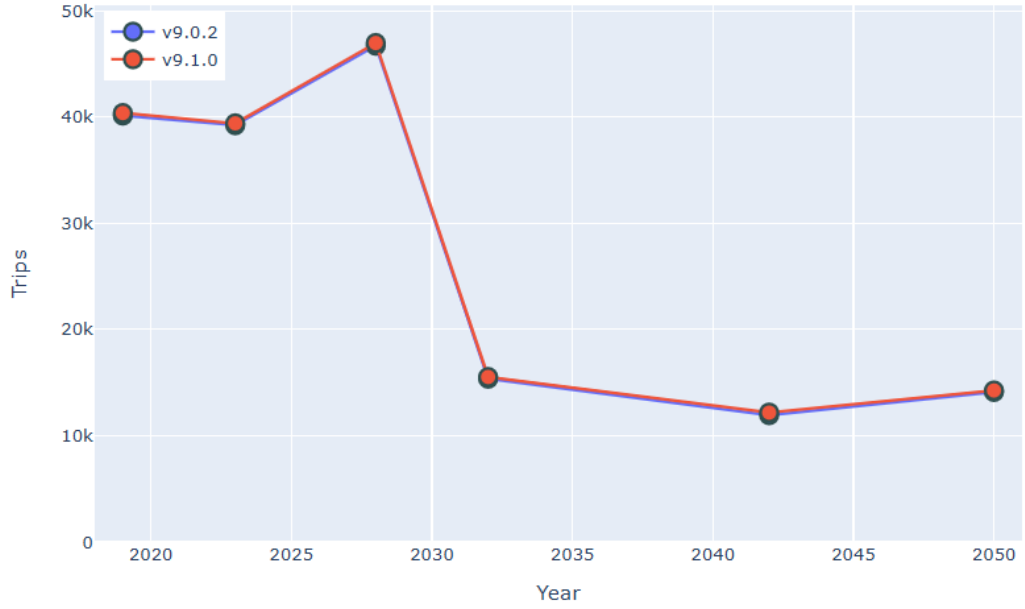


Figure 5.9 Daily Transit Trips - Local Bus