RSG
the science of insight

Metropolitan Washington Council of Governments

GEN3 MODEL USER GUIDE

January 31, 2024

V

PREPARED FOR:
METROPOLITAN WASHINGTON COUNCIL OF GOVERNMENTS

SUBMITTED BY:
RSG
WITH BASELINE MOBILITY GROUP AND METROPOLITAN WASHINGTON COUNCIL OF GOVERNMENTS

55 Railroad Row
White River Junction, VT 05001
802.295.4999
www.rsginc.com

LIST OF FIGURES

LIST OF TABLES

CREDITS

This document is the culmination of a collaborative effort among MWCOG, RSG, and Baseline Mobility Group (BMG).

The Introduction, System Architecture, Running the Model, and Working with ActivitySim, were written by RSG. Additionally, RSG wrote large parts of Data Inputs and Outputs, but there were a significant number of input tables that were copied from existing MWCOG documentation. The More Detailed Description of Model System was assembled by RSG using MWCOG's documentation (Highway Skimming, Toll Searching, Transit Skimming, Truck Model, Commercial Vehicle Model), RSG's PopulationSim and Auxiliary Model documentation, BMG's External and Visitor Transit Processing, and ActivitySim's documentation. The Appendix includes very detailed flow charts created by MWCOG and code examples written by RSG.

Specific roles include:
Andrew Rohne, RSG, Author
Joel Freedman, RSG, Author
Feng Xie, MWCOG, Author and Reviewer
Ray Ngo, MWCOG, Author and Reviewer
Mark Moran, MWCOG, Reviewer
Dusan Vuksan, MWCOG, Reviewer
Mushtaqur Rahman, BMG, Author

1.0 INTRODUCTION

The Metropolitan Washington Council of Governments (MWCOG or COG) is an independent, nonprofit association that brings area leaders together to address major regional issues in the District of Columbia, suburban Maryland, and Northern Virginia. COG's membership comprises 300 elected officials from 24 local governments, the Maryland and Virginia state legislatures, and U.S. Congress. The Board of Directors is COG's governing body and is responsible for its overall policies. The National Capital Region Transportation Planning Board (NCRTPB or TPB) is the federally designated metropolitan planning organization (MPO) for metropolitan Washington. TPB is responsible for developing and carrying out a continuing, cooperative, and comprehensive transportation planning process in the metropolitan area. COG is the administrative agent for the TPB, and the TPB is staffed by COG's Department of Transportation Planning (DTP). The TPB staff, with some consultant assistance, develops, maintains, applies, and improves the TPB's family of regional travel demand forecasting models, which are used for regional, long-range transportation planning in the metropolitan Washington region. These regional travel demand models are developed under the guidance of the Travel Forecasting Subcommittee (TFS).

This document is a user's guide for the COG/TPBGeneration-3, or Gen3, Travel Demand Forecasting Model, which covers the metropolitan Washington region, and which has been developed by RSG and Baseline Mobility Group (BMG) for COG. COG/TPB staff has also helped in the development of the Gen3 Model.

The Gen3 Model is a state-of-the-practice travel model that utilizes ActivitySim for modeling internal person demand and Cube Voyager for modeling auxiliary demand and transportation supply (e.g., trip assignment). ActivitySim is open-source software for developing and applying activity-based travel models (ABMs). The development of ActivitySim is guided by the ActivitySim Consortium, which currently consists of 11 public-sector agencies, including MWCOG. The Association of Metropolitan Planning Organizations Research Foundation (AMPORF) is the administrative agent for the ActivitySim Consortium.

Following the introduction, this document describes the system architecture, data inputs and outputs, how to run the model, how to work with ActivitySim, and finally detailed model/sub-model descriptions.

1.1 TPB TRAVEL MODELS AND TPB'S STRATEGIC PLAN FOR MODEL DEVELOPMENT

In 2018, COG/TPB set out to develop a next-generation travel demand model. The project team, consisting of RSG and Baseline Mobility Group, recommended that COG transition from its current aggregate, trip-based travel demand model (i.e., Generation 2, or Gen2, Model) to a simplified activity-based model (ABM) implemented in the open-source ActivitySim software platform. The new model is known as the Generation 3, or Gen3, Travel Model.

The Gen3 Model was implemented in two phases. Phase 1, which ended in February 2022, created a prototype model that was able to be tested by the COG/TPB staff. Phase 2, which is scheduled to finish in fall 2023, aimed to develop a production-use model that can be used for regional planning work, such as the air quality conformity analysis and scenario studies. The purpose of the phased approach to model development was to use the initial deployment and calibration efforts to inform the scope of final model development and calibration/validation tasks, rather than scope the entire model development project at the project initiation. This allowed the project team to learn from the initial deployment and prioritize resource allocation in Phase 2, and to ensure that the final delivered Gen3 Model meets the needs of MWCOG, partner agencies, and decision-makers. In Phase 2, additional models were estimated, a few new components (including a vehicle type choice model) were added, and the entire system was calibrated and validated to create a production-capable model system. Following delivery of the Gen3, Phase 2 Model by the consultant team to COG/TPB staff, staff plans to conduct suitability testing, to ensure that the model is ready for production use.

1.2 WHAT IS AN ACTIVITY-BASED TRAVEL MODEL (ABM) AND HOW IS IT DIFFERENT FROM A TRIP-BASED MODEL?

A trip is a one-way movement of a person or vehicle from an origin to a destination. A tour is a series of trips (a trip chain) beginning and ending at a home or work anchor location.[1] An activity-based travel model is a model that simulates the travel related choices made by individual travelers throughout a typical day. These choices include long-term choices, daily tour-level choices, and daily trip-level choices. Long-term choices include choices like a person’s work or school location (if applicable) and how many vehicles a household will own. Tour-level choices include whether or not a person will make mandatory tours, non-mandatory tours, or stay home during the day, as well as the tour mode, tour timing, and number of stops on the tour. Finally, trip-level choices determine specific modes, times, locations, and purposes for each segment of each tour. All of these choices are linked in the model system.

The main difference between trip-based and activity-based models is the level of detail and granularity in modeling travel behavior. Trip-based models focus on individual trips, while activity-based models represent travel as both trips and tours and consider the entire daily activity pattern of individuals and households. Activity-based models represent a 24-hour time window at a person level as a constraint on travel, recognizing that one person can be in only one place at a time. Trip-based models are aggregate and do not have an explicit accounting of a traveler’s time budget. In fact, most trip-based models represent travel by market segments rather than individual households and persons.

[1] Castiglione, Joe, Mark Bradley, and John Gliebe. “Activity-Based Travel Demand Models: A Primer.” SHRP 2 Capacity Project C46. Washington, D.C.: Transportation Research Board of the National Academies, 2015. http://www.trb.org/main/blurbs/170963.aspx.

Activity-based models are generally considered to be more realistic in terms of responses to inputs. They can consider many more variables including constraints and linkages between travel decisions than trip-based models. However, they are also more complex and computationally intensive than trip-based models and require additional and different skills to work effectively with model outputs.

Assuming that the model user has a basic knowledge of the activity-based modeling theory and model framework, this user's guide is focused on the setup and use of the Gen3 Model. For those who want to learn more about activity-based modeling, please refer to Activity-Based Travel Demand Models: A Primer[2] and the Transportation Model Improvement Program (TMIP) Activity Based Model Tutorial[3].

1.3 PURPOSE AND NEED FOR AN ACTIVITY-BASED MODEL FOR THE COG REGION

During the initial phase of development of the Gen3 Model, MWCOG outlined three objectives for the Gen3 Model:[4]

1. To ensure that the COG/TPB travel demand forecasting methods are either state of the practice or state of the art with respect to the modeling practices of peer MPOs.
2. To address current shortcomings with the TPB's adopted, production-use, trip-based travel demand model (currently the Gen2/Ver. 2.4 Model).
3. To ensure that the new model has the capability to address the most pressing regional transportation planning issues in the Washington, D.C. region.

An activity-based (AB) model, such as ActivitySim, is the current state of the practice. According to a survey of 23 peer MPOs conducted in 2015,[5] 70% have or were developing a production-use activity-based travel demand model. Since the survey was conducted, one of the surveyed MPOs that did not have an AB model in development now has an ActivitySim model under development (SEMCOG). It is therefore the conclusion of the RSG team that the state of the practice for peer MPOs is an activity-based model, and state of the art is an activity-based model that is "advanced" compared to peer MPO AB models in one or more ways - treatment of space, time, behavior, special travel markets, integration with dynamic traffic assignment, etc. RSG and COG believe that ActivitySim represents the state of the practice in terms of activity-based model form and function.

---

[2] National Academies of Sciences, Engineering, and Medicine. 2014. Activity-Based Travel Demand Models: A Primer. Washington, DC: The National Academies Press. https://doi.org/10.17226/22357.
[3] Transportation Model Improvement Program. 2012. TMIP Activity Based Model Tutorial. Online video series. https://tmip.org/content/tmip-activity-based-model-tutorial.
[4] RSG, Inc. and BMG. Gen3 Model Design Plan. July 2, 2020
[5] Cambridge Systematics, Inc., *Status of Activity-Based Models and Dynamic Traffic Assignment at Peer MPOs, Task Order 15.2, Report 2 of 3*, 10–11.

One of the key shortcomings of the Gen2/Ver. 2.4 Model is the aggregation bias[6] due to the trip-based model structure. By contrast, ActivitySim is a disaggregate activity-based model which eliminates some aggregation bias. The structure of the ActivitySim model also allows the use of any number of explanatory variables without significantly increasing the computational burden. In addition, the ActivitySim model effectively responds to some of the key shortcomings of the current trip-based model. For example, time-of-day choice is explicitly represented at the tour level. ABMs, such as ActivitySim, consider accessibility and therefore respond to changes in congestion. As peak-period travel gets more congested relative to off-peak periods, the utility and probability of travel in peak periods decreases, all else being equal. In addition to this, specific features of ActivitySim and the Gen3 Model that address shortcomings in the Gen2 Model include:

- Non-motorized modes
- Transportation Network Companies (TNCs) and other shared mobility modes, such as Uber and Lyft
- Teleworking, including periodic telecommute and work-from-home
- Special travel markets
    - Ability to model and report about travel made by low-income and zero-auto households.
    - Ability to model and report about travel made by senior citizens.
- Employer-based transit subsidies
- Connected/autonomous vehicles (CAVs)
- Improved sensitivity to time, pricing, and income
    - All households and people use disaggregated travel times and costs, and use individual household incomes and values of time from the representative population for the basis of all travel decisions.

Activity-based models, including ActivitySim, can be used to address the most pressing regional transportation issues in the Washington, DC region. The Coordinated Travel - Regional Activity-based Modeling Platform (CT-RAMP) model system upon which ActivitySim is developed has been used successfully in other MPO regions (e.g., MTC, ARC) for long-range regional transportation planning, transportation improvement program and air quality conformity analyses for approximately 10 years. The model system has been used to analyze the impacts of land use on transportation demand, the effects of highway capacity increases, planning for priced infrastructure including toll roads and managed lanes, demand for at-grade and grade-separated transit investments, and many other relevant projects and policies.

[6] See https://www.statology.org/aggregation-bias/ for an explanation of aggregation bias

1.4 ACTIVITYSIM AND THE ACTIVITYSIM CONSORTIUM

ActivitySim is an open platform for activity-based modeling. It is the product of a collaborative activity-based travel behavior modeling software development project that is supported both financially and managerially by a consortium of public-sector agencies. The travel model that is currently implemented in the ActivitySim platform is based on a fully functional activity-based model that was originally designed for the San Francisco Bay area's Metropolitan Transportation Commission (MTC)[7] The model currently implemented in the ActivitySim framework is a member of the CT-RAMP model family[8] The system relies on logit choice models to represent travel decisions (how frequently to travel, where to travel, by what mode, etc.) and was designed to achieve behavioral realism within a practical system of components. The existing model addresses many of the limitations noted above with respect to the MWCOG Gen2 trip-based model.

The original CT-RAMP model was developed jointly for both MTC and ARC and implemented in the Java programming language. In 2014, a consortium of Transportation Planning Agencies, including Metropolitan Planning Organizations (MPOs), state Departments of Transportation (DOTs), and municipal planning agencies created the ActivitySim project to "create and maintain advanced, open-source, activity-based travel behavior modeling software based on best software development practices for distribution at no charge to the public".[9] The consortium decided to adopt the MTC Travel Model One (TM1) activity-based model as the basis for the new software tool, and subsequently contracted for consultant services under the Association of MPOs (AMPO) to convert the model to Python, enhance, and maintain the software code. The new Python-based software is very flexible, configurable, and easier to administer.

There are currently 11 member agencies in the consortium and one administrative agency (AMPO). The member agencies are ARC, MTC, San Diego Association of Governments (SANDAG), San Francisco County Transportation Authority (SFCTA), Puget Sound Regional Council (PSRC), Southeast Michigan Council of Governments (SEMCOG), Oregon Department of Transportation (ODOT), Metropolitan Council (Met Council), MWCOG, Chicago Metropolitan Agency for Planning (CMAP), and The Ohio Department of Transportation. MTC Travel Model One has been fully implemented in ActivitySim. Model deployments are currently underway for ARC (there are minor differences between the MTC and ARC models that are being implemented), SEMCOG, SANDAG, MTC, Met Council, PSRC, MWCOG and several agencies

---

[7] Metropolitan Transportation Commission. Plan/Bay Area: Technical Summary of Predicted Traveler Responses to First Round Scenarios Technical Report. March 22, 2011, available https://mtcdrive.app.box.com/s/3qj8egg1esg01ac68qtnlq8e0c4l4h6s.

[8] Davidson, Vovsha, Freedman, and Donnelly. CT-RAMP Family of Activity-Based Models. Australasian Transport Research Forum 2010 Proceedings. 29 September – 1 October 2010, Canberra, Australia. Publication website: http://www.patrec.org/atrf.aspx

[9] "ActivitySim: An Open Platform for Activity-Based Travel Modeling," 2020. https://activitysim.GitHub.io/, accessed April 23, 2020.

that are not members of the consortium, including[10] Transport for New South Wales (Sidney, Australia), TransLink (Vancouver, Canada), Southeast Regional Planning Model (SERPM, Southeast Florida), Perth Australia, Melbourne Australia, and the Lawrence Livermore National Laboratory in their Behavior, Energy, Autonomy, and Mobility (BEAM) model. As can be gleaned from the above information, ActivitySim has a robust and active user community.

The ActivitySim model framework has the following characteristics:

- Utilizes tours (sequences of trips beginning and ending at an anchor location such as home or work) as an organizing principle for the generation of travel and to ensure consistency across trips within a tour.
- Utilizes micro-simulation for modeling travel choices, in which a representative population[11] is generated, and explicit mobility and travel choices are made for each decision maker in the population according to contextual probability distributions.
- Addresses both household-level and person-level travel choices including limited intra-household interactions between household members.
- Schedules tours into time-windows to ensure there are no overlapping travel episodes.
- Reflects and responds to detailed demographic information including household structure, aging, changes in wealth, and other key attributes.

1.5 DESCRIPTION OF GEN3 MODEL

The Gen3 Model runs in three phases – a preparation phase, a feedback loop phase, and a final phase. The preparation phase prepares pricing conversion data, highway network data, highway skim data, and transit fares. The feedback loop phase prepares transit network and skim data, runs ActivitySim to model the internal person transportation demand, prepares auxiliary travel demand data for special transportation markets (airports, visitors, trucks, external-internal, internal-external, and external/through traffic), assigns highway trips, and then prepares new highway skim data. This feedback loop phase repeats four times with increasing sampling proportions of the region’s households (250k, 500k, 1m, and finally all households) and iterations 2-4 use the highway skim data from the prior iteration as inputs. After the final feedback loop iteration (i.e., Iteration 4), the model assigns transit trips (including internal transit trips from ActivitySim and external transit trips prepared in an exogenous process), prepares a transit summary file, and prepares a summary of the ActivitySim output in a visualizer.

In the Phase 1 development, a representative population (a.k.a. synthetic population) for the modeled region was created, and the ActivitySim model system was transferred from the Southeast Michigan Council of Governments (SEMCOG) region (Detroit, Michigan) to MWCOG. Under Phase 1 deployment, tour mode choice and tour destination choice models were

[10] Membership can change over time and is not limited to US-based transportation agencies. Examples are current as of June 2023.
[11] In the travel modeling literature, a representative population is usually referred to as “synthetic population”. In this document, the two terms are used interchangeably.

estimated.[12] After implementation of the estimated tour mode choice and tour destination choice models, several model components such as auto ownership, tour mode choice, trip mode choice, individual non-mandatory tour frequency and intermediate stop frequency models were calibrated to the observed distributions from the 2017-18 MWCOG Regional Travel Survey (RTS) and the 2018-19 Maryland Statewide Household Travel Survey (MTS) data. The processing of RTS/MTS data was documented in the Phase 1 data development report.[13] The 2018 traffic counts and transit boarding counts that are used for Phase 1 model validation were provided by MWCOG staff and were documented separately.[14] Thus, the preparation of the count data is not covered in this document. Some important documentation for the Phase 1 development can be found at the COG website.[15]

Figure 1 shows the general structure of the ActivitySim portion of the Gen3 Travel Model. As noted in the figure legend, boxes with the red borderline indicate models that were estimated as part of the Phase 1 development and boxes with the yellow borderline were estimated as part of the Phase 2 development. Thus, seven models were estimated in Phase 1 and seven models were estimated in Phase 2.

---

[12] RSG, “Tour Mode Choice and Destination Choice Model Estimation”, Technical Report Prepared for Metroplitan Washington Council of Governments, January 19, 2022.
[13] RSG, “Gen3 Data Development.” Metropolitan Washington Council of Governments, National Capital Region Transportation Planning Board. December 29, 2021 (https://app.box.com/s/xe5vb28daox1aqtw895iy2r5ocy584w8)
[14] See, for example, “Metrorail Average Weekday Passenger Boardings (Station Level): 1977 to 2018,” Summary Table (Washington, D.C.: Washington Metropolitan Area Transit Authority, September 2018), https://www.wmata.com/about/records/public_docs/upload/2018_historical_rail-ridership_May-weekday-avg.pdf; Meseret Seifu to Feng Xie, “2018 Daily and Hourly Traffic Counts,” Memorandum, June 22, 2021.
[15] https://www.mwcog.org/transportation/data-and-tools/modeling/developmental-travel-model/

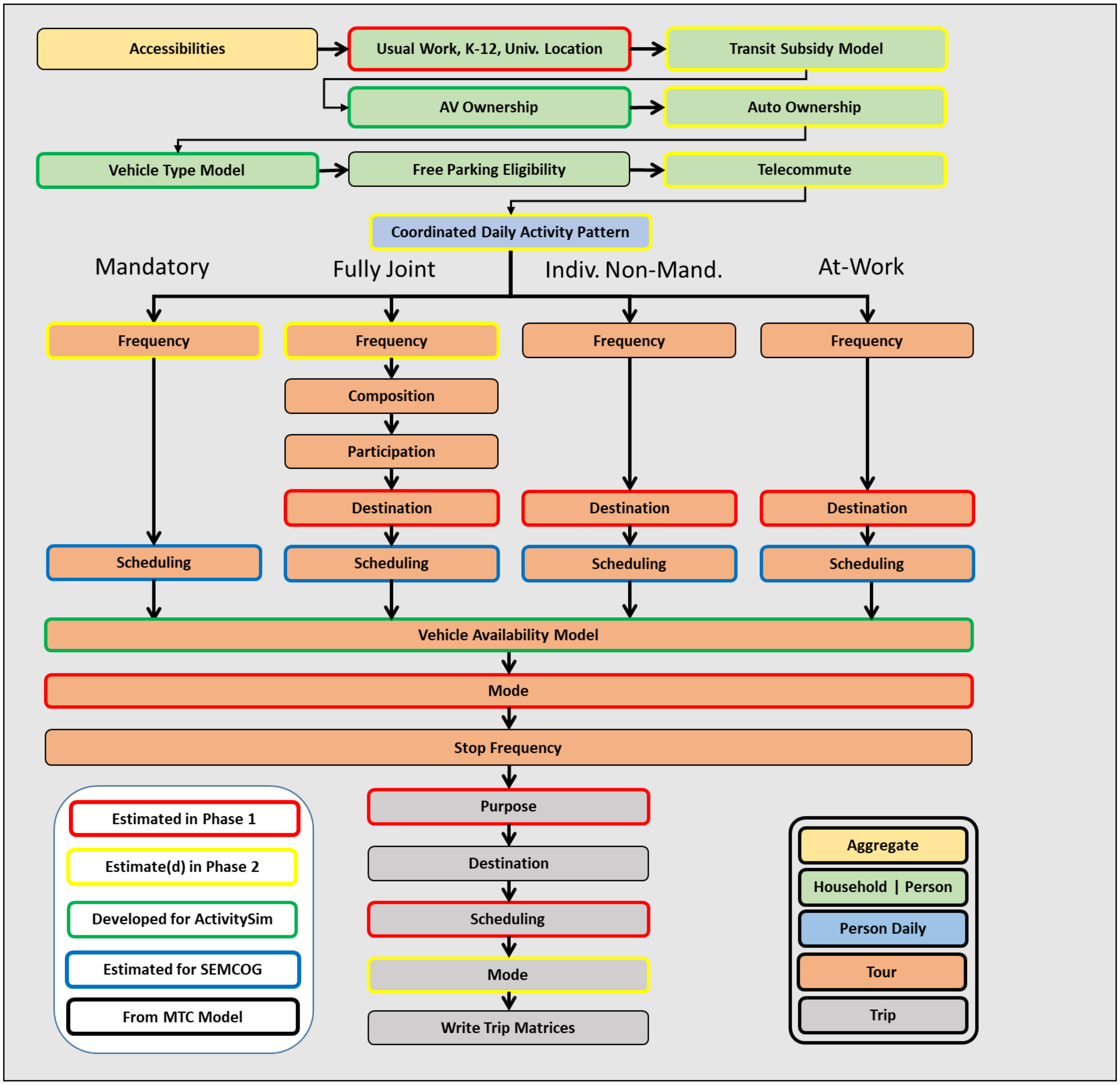


FIGURE 1: MWCOG GEN3 ACTIVITYSIM FLOWCHART

Treatment of Time

ActivitySim in the Gen3 Model is set up with a 30-minute temporal resolution that begins at 3 A.M. and ends at 2:59 A.M. the following day.

Temporal integrity is ensured so that no activities are scheduled with conflicting time windows, except for short activities/tours that are completed within an hour increment. For example, a person may have a short tour that begins and ends within the 8am-9am period, as well as a second longer tour that begins within this time period but ends later in the day.

A critical aspect of the model system is the relationship between the temporal resolution used for scheduling activities, and the temporal resolution of the network simulation (skimming and assignment) periods. Although each activity generated by the model system is identified with a start time and end time in 30-minute increments, level-of-service matrices will be created for four aggregate time-of-day periods consistent with the current Ver. 2.3 (trip-based) Model: A.M. Peak, Midday, P.M. Peak, and Night-Time (Evening/Early A.M). The trips occurring in each time period reference the appropriate transport network depending on their trip mode and the mid-point trip time. The definition of time periods for level-of-service matrices is given in Table 1, below.

TABLE 1: TIME PERIODS FOR LEVEL-OF-SERVICE SKIMS AND ASSIGNMENT

NUMBER	DESCRIPTION	BEGIN TIME	END TIME
1	A.M. Peak	6:00 A.M.	8:59 A.M.
2	Midday	9:00 A.M.	2:59 P.M.
3	P.M. Peak	3:00 P.M.	6:59 P.M.
4	Night Time (Evening/Early AM)	7:00 P.M.	5:59 A.M.

Treatment of Space

The Gen3 Model is set up to utilize Transportation Analysis Zones (TAZs) as the unit of space in the model. The TAZs are the current Ver. 2.3 Model system TAZs. There are 3,669 TAZs internally in the Gen3 Model, numbered 1 through 3,675 (TAZs 2555, 2629, 3103, 3478, 3482, and 3495 are skipped) and 47 external stations (3676 to 3722). The TAZs are shown in Figure 2, with stars representing the external stations. Although ActivitySim can also operate at a finer geography, typically referred to as Micro-Analysis Zones (MAZs), it was the project team's decision to keep TAZ as the smallest geography in the Gen3 Model, mainly because COG's Cooperative Land Use Forecasts data are currently generated at the TAZ level.

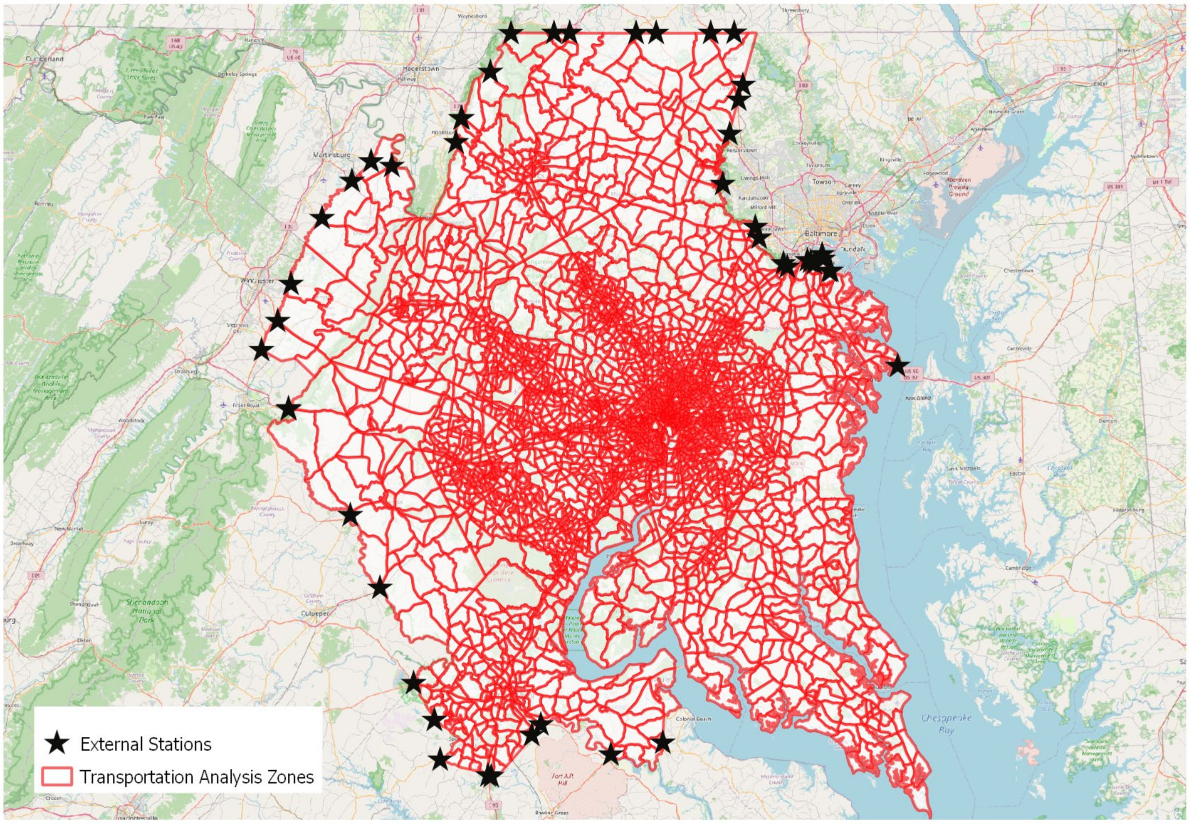


FIGURE 2: GEN3 MODEL TRANSPORTATION ANALYSIS ZONES

1.6 DIFFERENCES BETWEEN GEN2 MODEL AND GEN3 MODEL

The TPB Model can have four types of updates: a bug fix, a new feature, a feature enhancement, and updated documentation. The primary difference between the TPB's current production-use Gen2/Ver. 2.4 Model and the Gen3 Model is that the Gen2/Ver. 2.4 model is a trip-based model that forecasts the trips from each zone based on household characteristics and aggregates trips for the purposes of trip distribution and mode choice. By contrast, the Gen3 Model is an activity-based model that micro-simulates travel at the person level and has more detailed processes that affect traveling. This detail includes decisions made with specific person-level data instead of averages. This specific data can include household income, ages of people in the household, person type (I.e., full-time worker, part-time worker, university student, retired person, nonworking adult, driving-age child, school-age pre-driving-age child, and preschool-age child), work or school locations, if a worker is telecommuting that day, and even interactions among household members. All of the major updates are shown in Table 2.

TABLE 2: MAJOR GEN3 MODEL UPDATES RELATIVE TO GEN2 MODEL

#	DESCRIPTION	TYPE OF UPDATE	FURTHER DETAILS AND BENEFIT(S)	CHANGES MODEL RESULTS?
1	Replacement of trip-based demand model for residents with ActivitySim	New Feature	New model sensitivities in terms of household and person variables, sensitivity to changes in network level of service by time-of-day, representation of telecommuting, transit subsidies, park-and-ride versus kiss-and-ride, ride-hail modes and autonomous vehicles.	Yes
2	Representation of external-internal transit travel	New Feature	Used transit on-board survey data to generate trip tables representing non-resident travel into and out of modeled area.	Yes
3	Updated transit skimming procedures	New Feature	Replaced Cube TRNBUILD with Cube Public Transport (PT) module to generate multi-path transit skims with post-processing to ensure transit paths are more consistent with mode choice model alternatives.	Yes
4	Representation of external-internal highway travel	Feature Enhancement	Uses Cube Voyager and ActivitySim outputs to estimate internal-external and external-internal trips.	Yes
5	Addition of Activity-Based Model Visualizer	New Feature	Model creates a dashboard showing model results and compares them to the expanded household survey or a base model run.	No

2.0 SYSTEM ARCHITECTURE

This section discusses the hardware and software requirements for running the model. In general, a powerful server is required to run the model. Specific specifications are discussed in section 2.1. The main software required for the model includes Cube Voyager, Python, and R. Cube Voyager is a commercial transportation modeling platform that must be purchased separately and requires a computer with a Windows operating system. Python is an open-source cross-platform programming language that is currently one of the most popular programming languages. Python is the core language of ActivitySim. R is an open-source cross-platform statistical analysis and scripting language. In the Gen3 Model, R is used for the model visualizer. The required software is discussed further in section 2.2.

2.1 HARDWARE REQUIREMENTS

The MWCOG Gen3 Model runs on a Microsoft Windows workstation or server, with the minimum and recommended system specification as follows:

- Minimum specification:
  - Operating System: 64-bit Windows 7, 64-bit Windows 8 (8.1), 64-bit Windows 10, 64-bit Windows Server 2019
  - Processor: 16-core CPU processor
  - Memory: 200 GB RAM
  - Disk space: 600 GB
- Recommended specification:
  - Operating System: 64-bit Windows 10
  - Processor: Intel CPU Xeon Gold / AMD CPU Threadripper Pro (24+ cores)
  - Memory: 256 GB RAM
  - Disk space: 1200 GB

In general, a higher CPU core count and RAM will result in faster run times as long as ActivitySim is configured to utilize the additional processors and RAM.

Note that the model is unlikely to run on servers that have less than 200 GB of RAM.

2.2 SOFTWARE REQUIREMENTS

Two software applications, Cube and a Python Package Manager (e.g., MambaForge or Anaconda) should be installed on the computer that will be used to run the model.

The Gen3 Model system is an integrated model that is controlled by and primarily runs in the Cube transportation planning software platform. Bentley Systems, Inc. produces and markets Cube. Cube is a proprietary transportation planning software platform that includes Cube Base and Cube Voyager. Cube Base allows for viewing and editing highway and transit network files and viewing of matrix files. These files are critical for the transportation-supply-side of the model. The Gen3 Model uses Cube Connect Edition, version 6.5.[16] Note that, with Cube Connect versions of the Cube software (i.e., versions of Cube downloaded from Bentley's website after Bentley purchased Cube), the Bentley Connect Edition software will need to be logged in and activated after any server reboots. A Python Package Manager is software that creates an environment for an instance of Python. Instance refers to the libraries used by the Python processor in the environment. ActivitySim and related Python processes in the Gen3 Model are executed in an environment that is setup with a specific version of Python and specific library versions. This ensures that changes outside of the Python environment will not cause errors or change model results, and additionally ensure that the specific version of Python and specific libraries needed by the Gen3 Model and ActivitySim do not cause errors or changes to other Python software installations on the server. The libraries needed by ActivitySim extend the base functionality of Python. Two package managers have been tested with the Gen3 Model: Mambaforge and Anaconda. In general, Mambaforge is preferred because it is free for organizations of all sizes, while Anaconda requires a paid subscription for organizations larger than 200 users. To install Mambaforge, follow the instructions on https://mamba.readthedocs.io/en/latest/installation.html#fresh-install. If Anaconda is preferred, it can be installed by following the instructions at https://www.anaconda.com/. The Gen3 Model was initially developed with Anaconda but was later changed to Mambaforge and, at the time of the Gen3 Model release, both package managers are supported. Model results are not affected by the choice of package manager, only a small increase in runtime was observed with MambaForge.

2.3 SOFTWARE INSTALLATION

In order to install the model, a user needs to download the model, install ActivitySim, and download the model inputs and dependencies. These steps are described in detail below.

Step 1: Download the Gen3 Model

Download the model from the link provided by MWCOG. Unzip the model and browse to the /Gen3_Model folder. There are at least three sub-folders in this directory:

/documentation - This folder contains all model documentation.

/source - This folder contains the scripts and settings files to run various components of the model. The configs subfolder contains the ActivitySim configuration files. The scripts subfolder contains the batch scripts, Cube scripts and Python scripts used to run the

[16] https://communities.bentley.com/products/mobility-simulation/cube/b/cube-blog/posts/check-out-what-s-new-in-cube-connect-edition-v-6-5

model. The software subfolder contains the supporting software executables. The visualizer subfolder contains the ABM Visualizer setup.

/2018_base - This is the 2018 base year scenario folder. This folder can be copied within the /Gen3_Model directory to run additional scenarios. This folder has an inputs folder, an outputs folder, and the batch scripts to run the model.

Additional folders relate to specific model scenarios. As part of the validated model, the 2018_base folder is expected to exist. Other folders may exist as well, such as “2045_e”" or “2045_alt1”. There is no requirement that the model folder name include the year, but it is a good practice do so since the files included in the input folder are generically-named. Using the year and a description as part of the folder name reduces the potential for confusion related to what inputs and outputs are included in a scenario folder.

Step 2: Install and Configure ActivitySim

As noted above, the Python Package Manager (either Mambaforge or Anaconda) will need to be installed. Once one of those two package managers is installed, a specific computer environment must be created to run ActivitySim. The environment is a configuration of Python that is for ActivitySim - this environment allows ActivitySim to use specific software libraries without interfering with the server’s installed version of Python (if one exists, it is not required) and keeps other Python installations from interfering with ActivitySim. Creating the environment uses one of the following commands from within the Miniforge Prompt (for Mambaforge) or the Anaconda PowerShell Prompt (for Anaconda or Miniconda), . For both package managers, use the cd /d command to change directories to the model’s ActivitySim config folder, which is <model location>/source/configs/activitysim (e.g., `cd /d e:\Gen3_Model\source\configs\activitysim`) prior to running the command below.

```
conda env create --file environment.yml
```

After installing the environment, do a quick test of it by activating it, using:

```
conda activate gen3_model121
```

Run “conda `env list`” to get the paths to the Anaconda (shown next to “base”) and “gen3_model12” environment installed with the Mambaforge or Anaconda Package Manager. The former path will be set for MAMBA_OR_ANACONDA_DIR and the latter will be used in the PYTHON variable in the run_mode.bat file as instructed below.

Once the environment is prepared, ActivitySim needs to be configured to ensure it can start Python in the package manager. To do this, open the run_model.bat file in 2018_base and follow these steps:

1. Look for `:: STEP ONE: Set either USE_ANACONDA or USE_MAMBA to True and the other to False`. Set one of these to True and the other to False corresponding to the package manager in use; by default USE_MAMBA is set to be used.
2. Look for `:: STEP TWO: Set the path to Python` and set the path to Python as appropriate on the server. A few examples are provided to guide where to look if the path isn't known.
3. Look for `:: STEP THREE (ANACONDA)` or `:: STEP THREE (MAMBA)` and set the path to the appropriate location on the server. This path allows the package manager to be activated so Python runs within the package manager's environment, which is critical to Python running properly. A few examples are provided to guide where to look if the path isn't known.

	PACKAGE MANAGER INSTALLED FOR "JUST ME"	**PACKAGE MANAGER INSTALLED FOR "ALL USERS"**
Anaconda	MAMBA_OR_ANACONDA_DIR= C:\Users%USERNAME%\Anaconda3 PYTHON=C:\Users%USERNAME%\Anaconda3\envs\gen3_model\python.exe or PYTHON=C:\Users%USERNAME%\AppData\Local\conda\conda\envs\gen3_model\python.exe	MAMBA_OR_ANACONDA_DIR= C:\ProgramData\Anaconda3 PYTHON=C:\Users%USERNAME%.conda\envs\gen3_model\python.exe
Mambaforge	MAMBA_OR_ANACONDA_DIR= C:\Users%USERNAME%\AppData\Local\mambaforge PYTHON=C:\Users%USERNAME%\AppData\Local\mambaforge\envs\gen3_model\python.exe	MAMBA_OR_ANACONDA_DIR= C:\ProgramData\mambaforge PYTHON=C:\Users%USERNAME%.conda\envs\gen3_model\python.exe

By default, the Model Visualizer is set to compare the current model run to household survey and Census data for calibration and validation comparison. In the case that the model output should be compared to a different base condition, there are five environment variables that need to be updated in the run_model.bat file. These are marked in the file by a comment of "VISUALIZER INPUTS". The code is shown below:

```
:: ================= VISUALIZER INPUTS ===================================
:: Visualizer Base Data
SET BASE_SUMMARY_DIR=%SCEN_DIRECTORY%inputs\Visualizer\data\summarized_survey
SET BASE_SCENARIO_NAME=SURVEY
```

```
:: for survey base legend names are different [Yes\No]
SET IS_BASE_SURVEY=Yes
SET BASE_SAMPLE_RATE=1.0
:: Scenario Name for Visualizer
SET BUILD_SCENARIO_NAME=Gen3
```

The first variable to set is the `BASE_SUMMARY_DIR`. This should be a file path to the output summary files from either a survey or a model run. The second is the `BASE_SCENARIO_NAME`, which is used in the visualizer to as the name of the base condition. It is recommended that this be kept relatively short, like “SURVEY”, “BASE”, “Base Model”, etc. The third variable is `IS_BASE_SURVEY` and that should be set to either “Yes” (the base data is a survey dataset) or “No” (the base data is a model run). The fourth variable to set is `BASE_SAMPLE_RATE`, which will adjust the base input data by a factor. In general, this will be 1.0. The fifth and final variable to set is the `BUILD_SCENARIO_NAME`. This is like the `BASE_SCENARIO_NAME` in that it is the name used for the current output model run. Again, this should be kept reasonably short, such as “Gen3”, “2045 Alt1”, or “Build” would be fine.

Step 3: Download Model Inputs and Dependencies from Centralized Location (e.g., Box or DTP LAN)

1. Download input files from Box, and unzip (“extract here”) inside the inputs (/Gen3_Model/2018_base/inputs) folder

2. Download software files from Box and unzip (“extract here”) inside the software (/source/software) folder

3. Download visualizer dependencies from Box and unzip (“extract here”) inside the dependencies (/source/visualizer/dependencies) folder

Once extracted, the inputs subfolder (inside the 2018_base folder) should be around 4 GB and the source folder should be around 1.5 GB.

2.4 MODEL CONFIGURATION

This section includes one-time preparation items required to run the Gen3 Model. If the server being used to run the model has 256 GB RAM and 24 processor cores, this is optional. In the case that there is more or less RAM and/or a different number of processor cores, ActivitySim needs to be configured to use the correct number of processors and RAM. To do this, open the configuration source file in source\configs\activitysim\configs_mp\settings_source.yaml and look for the following two items:

```
chunk_size: 200_000_000_000
num_processes: 20
```

In computer programming, “chunking” refers to strategies for improving performance by using special knowledge of a situation to aggregate related memory-allocation requests. In ActivitySim, chunking is a method to avoid performance problems and potentially crashing the host computer due to memory problems. Chunking is built into ActivitySim and splits the "choosers" (the data it is simulating) into chunks that do not overrun the server's RAM. To do this, ActivitySim must use a training mode to determine the chunk sizes, which it saves in the chunk cache file.

Chunk size should be around 80% of the physical RAM in the server – so 200_000_000_000 (200 GB) is appropriate for a server with 256 GB of RAM. A server with 128 GB of RAM should use 100_000_000_000; a server with 512 GB of RAM should use 410_000_000_000. The num_processes variable should be at least one processor fewer than the number of processors available on the server. In testing, we have found that using four processors fewer than the number of processors available seems to ensure the model runs without crashing the server.

If the chunk size is changed, the model needs to be run in chunk training mode. Chunk training prepares a file that tells ActivitySim how to divide parts of the model to be as efficient as possible. To do this, open the scenario folder (e.g. 2018_base) and edit the run_model.bat file. Look for the following two lines:

```
:: ActivitySim chunk size training switch [True/False]
SET ACTIVITYSIM_CHUNK_TRAINING=False
```

Change SET ACTIVITYSIM_CHUNK_TRAINING=True and then follow the directions to run the model. The model will run one model iteration to prepare the chunk cache file, which tells ActivitySim how much RAM it can use for each step. The chunk cache file is generated in the ActivitySim output cache folder. Specifically, if the chunking is run under \2018_base, the chunk cache file will be generated at "\2018_base\outputs\activitysim\cache\chunk_cache.csv". Once chunk training has been run, the ACTIVITYSIM_CHUNK_TRAINING line can be set to False and the model run normally. The chunk cache file generated by the chunking should be copied to replace the existing chunk cache file at "\source\configs\activitysim\chunk_cache.csv", which is used in a normal model run for all scenarios.

Other Config Options

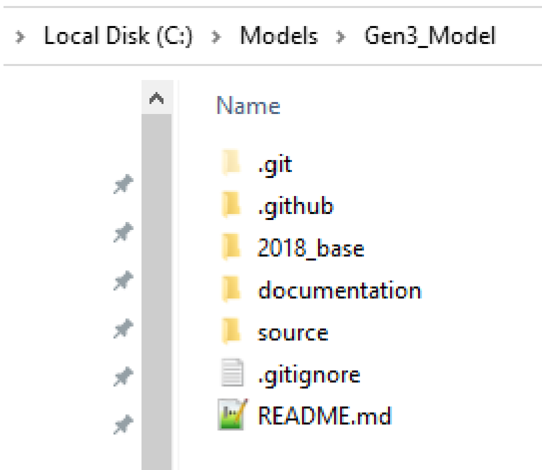
The items in this section are for options that may not be necessary for all model runs.

`RUN_MINI`: This option is used for a short model run that runs only the first feedback loop out of the four feedback loops and can be used for debugging and functionality testing. By default, this is set to False, set to True to activate this feature.

`AUTO_SHUTDOWN`: This option activates an automatic shutdown when the model completes the run, which is ideal when using on-demand cloud servers, where charges accrue by the time used. By default, this is set to False, set to True to activate this feature. This option will shut down the server *regardless of whether the model run was successful.* It is up to the analyst to check the model output logs to ensure that the model run was successfully completed. It is also up to the analyst to ensure that other users are not using the server prior to activating an automatic shutdown.

2.5 DIRECTORY STRUCTURE

FIGURE 3: MWCOG GEN3 MODEL FILE STRUCTURE



The contents of the Gen3 Model folders are described in more detail below.

.github: This is a folder used with Git to manage source code. DO NOT modify any files in this folder.

2018_base: This is the 2018_base scenario folder. Inside this folder are inputs and outputs from the model. This folder can be renamed by the user.

Documentation: Model documentation is included in this folder

Source: Model source code is located in this folder. In general, very few things should be changed inside this folder, and anything that is expected to be changed in this folder is described in this section of this document.

3.0 DATA INPUTS & OUTPUTS

3.1 MODEL INPUTS

The model inputs folder contains seven folders that organize the inputs into auxiliary, highway, land use, representative population (synthetic population), support, transit, and visualizer. These folders are discussed in detail below.

Auxiliary Data

The auxiliary input folder includes files that input transportation demand that is not addressed directly by the travel demand model. This auxiliary travel data, also known as exogenous or residual travel, includes airport passenger auto driver trips, external trips, truck trips, commercial vehicle trips, some school trips, taxi trips, visitor trips, and external/ through trips. In all cases except external trips, MWCOG uses endogenous models to forecast the demand for these markets. External trips, truck trips, and commercial vehicle trips are estimated in the model based on inputs in this folder.

TABLE 3: AUXILIARY FOLDER INPUT FILE LIST

FILENAME	DESCRIPTION
airpax.adr	Air passenger auto driver trips. Cube matrix with one unnamed table representing vehicle trips.
Ext_PsAs.dbf	External productions and attractions (autos, trucks, CVs). See Table 4.
taxi.adr	Special-generator taxi auxiliary trips. Cube matrix with one table named TaxiAdr_2018 representing vehicle trips.
visi.adr	Visitor auxiliary trips. Cube matrix with one table named VisiAdr_2018 representing vehicle trips.
xxaut.vtt	External/through trips. Cube matrix with one unnamed table representing vehicle trips.
xxcvt.vtt	External through trips. Cube matrix with three unnamed tables representing vehicle trips for light, medium, and heavy trucks.

TABLE 4: EXTERNAL PRODUCTIONS AND ATTRACTIONS (EXT_PSAS.DBF) FILE FORMAT

NAME	LENGTH	DESCRIPTION
TAZ	Integer	Transportation Analysis Zone Number
FACILITY	Character	Facility Name
AAWT_CTL	Integer	Average Annual Weekday Traffic
CNTFTR	Decimal	(Unused)
AUTO_XI	Integer	Auto External-Internal Vehicle Trips
AUTO_IX	Integer	Auto Internal-External Vehicle Trips
AUTO_XX	Integer	Auto External Through Vehicle Trips
CV_XX	Integer	Commercial Vehicle External Through Vehicle Trips
HBW_XI	Integer	Home-based Work External-Internal Vehicle Trips
HBS_XI	Integer	Home-based Shopping External-Internal Vehicle Trips
HBO_XI	Integer	Home-based Other External-Internal Vehicle Trips
NHB_XI	Integer	Nonhome-based External-Internal Vehicle Trips
CV_XI	Integer	Commercial Vehicle External-Internal Vehicle Trips
HBW_IX	Integer	Home-based Work Internal-External Vehicle Trips
HBS_IX	Integer	Home-based Shopping Internal-External Vehicle Trips
HBO_IX	Integer	Home-based Other Internal-External Vehicle Trips
NHB_IX	Integer	Non-home-based Internal-External Vehicle Trips
CV_IX	Integer	Commercial Vehicle Internal-External Vehicle Trips
TRCK_XX	Integer	Total Truck External Through Vehicle Trips
TRCK_XI	Integer	Total Truck External-Internal Vehicle Trips
TRCK_IX	Integer	Total Truck Internal-External Vehicle Trips
MTK_XI	Integer	Medium Truck External-Internal Vehicle Trips

HTK_XI	Integer	Heavy Truck External-Internal Vehicle Trips

Highway Data

The highway data is in the “hwy” folder and provides the input highway network data for use with highway skimming, highway assignment, and transit network processing. More details associated with the highway and transit network input data can be found in the network report that accompanies this User’s Guide.

TABLE 5: HIGHWAY FOLDER INPUT FILE LIST

FILENAME	DESCRIPTION
AM_Tfac.dbf	AM Period Toll Factors. See Table 6.
CPI_File.txt	Consumer Price Index (CPI) inflation factors. This file is a script that is input as source code during the model run.
Link.DBF	Database file of all links in the MWCOG highway network. See Table 7.
MD_Tfac.dbf	Mid-day (MD) Period Toll Factors. See Table 6.
Node.dbf	Database of all nodes in the MWCOG highway network. See Table 8.
NT_Tfac.dbf	Night-time (NT) Period Toll Factors. See Table 6.
PM_Tfac.dbf	PM Period Toll Factors. See Table 6.
Toll_Esc.dbf	Toll escalation factors. See Table 9.
Xtrawalk.txt	Extra walk links added to the highway network for use with the transit network processes.

TABLE 6: TOLL FACTOR INPUT FILE FORMAT

NAME	TYPE	DESCRIPTION
TOLLGRP	Decimal	Toll Group Number
[AM| MD| PM| NT]SOVTFTR	Decimal	Single Occupant Vehicle Toll Factor
[AM| MD| PM| NT]HV2TFTR	Decimal	Shared-Ride 2 Person Toll Vehicle Factor
[AM| MD| PM| NT]HV3TFTR	Decimal	Shared-Ride 3+ Person Toll Vehicle Factor
[AM| MD| PM| NT]COMTFTR	Decimal	Commercial Vehicle Toll Vehicle Factor
[AM| MD| PM| NT]TRKTFTR	Decimal	Truck Toll Vehicle Factor
[AM| MD| PM| NT]APXTFTR	Decimal	Airport Toll Vehicle Factor

TABLE 7: LINK.DBF FILE FORMAT

NAME	TYPE	DESCRIPTION
OBJECTID	Decimal	Internal Indexing Field
A	Decimal	A (From) Node Number
B	Decimal	B (To) Node Number
DISTANCE	Decimal	Length (in Miles * 100)
JUR	Decimal	Jurisdiction
SCREEN	Decimal	Screenline Number (if applicable)
FTYPE	Decimal	Facility Type 0 Centroid Connectors 1 Freeway 2 Major Arterial (Non-Freeway) 3 Minor Arterial

		4 Collector 5 Expressway 6 Ramp
TOLL	Decimal	Toll Value in Cents (Current-Year Prices)
TOLLGRP	Decimal	Toll Group Code
AMLANE	Decimal	Number of lanes during the AM period
AMLIMIT	Decimal	AM Peak Limit Code (0-9)
PMLANE	Decimal	Number of lanes during the PM period
PMLIMIT	Decimal	PM Peak Limit Code (0-9)
OPLANE	Decimal	Number of lanes during the MD and NT period
OPLIMIT	Decimal	MD and NT Peak Limit Code (0-9)
EDGEID	Decimal	Geometric network link identifier
LINKID	Decimal	Logical network link identifier
NETYEAR	Decimal	Planning year of network link
SHAPE_LENG	Decimal	Geometric length of network link (in feet)
PROJECTID	Character	Project identifier
TRANTIME	Decimal	Unused
WKTIME	Decimal	Unused
MODE	Decimal	Unused
SPEED	Decimal	Unused
STREETNAME	Character	Name of street

TABLE 8: NODE.DBF FILE FORMAT

NAME	TYPE	DESCRIPTION
N	Integer	Node Number

X	Integer	X Coordinate of node
Y	Integer	Y Coordinate of node
MRFZONE	Integer	Metrorail Station Fare Zone
CRMFZONE	Integer	MARC Station Fare Zone
CRVFZONE	Integer	VRE Station Fare Zone

TABLE 9: TOLL_ESC.DBF FILE FORMAT

NAME	TYPE	DESCRIPTION
TOLLGRP	Integer	Toll group code 1 = Flat toll (pertains to most existing tolled facilities); 2 = Toll that varies by time of day (e.g.. ICC), 3+= Tolls that change dynamically based on congestion level (e.g., VA HOT lanes/Express Lanes)
ESCFAC	Decimal	Deflation factor override
DSTFAC	Decimal	Distance (per mile) based toll factor in present year cents/dollar (optional)
AM_TFTR	Decimal	AM period Toll factor
PM_TFTR	Decimal	PM period Toll factor
OP_TFTR	Decimal	MD and NT period Toll factor
AT_MIN	Decimal	Area Type minimum override (optional)
AT_MAX	Decimal	Area Type maximum override (optional)
TOLLTYPE	Decimal	Toll Type 1 = Operating in base year 2 = Operating after base year

Land Use Data

The land use data includes land use and socioeconomic data inputs for the model.

FILENAME	DESCRIPTION
AT_override.TXT	File for overriding computed area types (e.g., used for Arlington National Cemetery and The Pentagon)
land_use.csv	Zonal data file for ActivitySim. See Table 10.

TABLE 10: LAND USE AND ZONAL DATA FILE FORMAT

NAME	TYPE	DESCRIPTION
TAZ	Integer	Zone ID
HH	Integer	Number of households
HHPOP	Integer	Household population
GQPOP	Integer	Group quarters population
TOTPOP	Integer	Total population
TOTEMP	Integer	Total employment
INDEMP	Integer	Industrial employment
RETEMP	Integer	Retail employment
OFFEMP	Integer	Office employment
OTHEMP	Integer	All other employment
JURCODE	Integer	Jurisdiction code (county or city)
LANDAREA	Decimal	Land Area of TAZ in square miles
HHINCIDX	Integer	Household income index
ADISTTOX	Decimal	Distance to external
TAZXCRD	Integer	TAZ X Coordinate
TAZYCRD	Integer	TAZ Y Coordinate
K_8	Integer	Kindergarten - 8th grade enrollment

G9_12	Integer	High school enrollment
COLLEGE	Integer	College enrollment
Park_Acres	Decimal	Park area of TAZ in acres
GC_Acres	Decimal	Golf course area of TAZ in acres
PRKCST	Decimal	Parking cost in TAZ
OPRKCST	Decimal	Off-peak parking cost in TAZ
TERMINAL	Integer	Terminal time for trip-ends in TAZ, in minutes
AREATYPE	Integer	Area type of TAZ

Synthetic Population Data

The synthetic population folder, popsyn, includes two data files of the synthesized population and household input files. These are inputs to ActivitySim.

FILENAME	DESCRIPTION
synthetic_hh.csv	Synthetic household listing from the population synthesizer. See Table 11.
synthetic_per.csv	Synthetic person listing from the population synthesizer. See Table 12.

ActivitySim and other activity-based travel models require a representative population (a.k.a., synthetic population) as an input. The synthetic population data includes a household file and a person file. The attributes in these files are provided in Table 11 and Table 12 and are a result of the best fit for the controlled attributes of the population. For more information, see the MWCOG Population Synthesizer Report[17]. The procedure for generating a synthetic population is not run by default but should be re-run any time the user changes the number of households or persons in one or more TAZs. The procedure for generating the synthetic population is described in Section 6.4.

[17]RSG. MWCOG Population Synthesizer. Final Report. Metropolitan Washington Council of Governments, National Capital Region Transportation Planning Board, August 4, 2021.https://www.mwcog.org/assets/1/6/MWCOG_Population_Synthesizer_COG_final.pdf

TABLE 11: SYNTHETIC HOUSEHOLDS FILE FIELDS

NAME	TYPE	DESCRIPTION
household_id	Integer	Household ID number
puma_geoid	Integer	Household PUMA Geographic ID
TAZ	Integer	Household TAZ ID
TYPE	Integer	Type of housing unit 1 = Household 2 = Institutional group quarters 3 = Noninstitutional group quarters
NP	Integer	Number of people in household
VEH	Integer	Number of vehicles in household
HHT	Integer	The type of household - Census PUMS HHT variable: N/A: Not applicable (group quarters or vacant) 1: Married couple household 2: Other family household: Male householder, no spouse present 3: Other family household: Female householder, no spouse present 4: Nonfamily household: Male householder: Living alone 5: Nonfamily household: Male householder: Not living alone 6: Nonfamily household: Female householder: Living alone 7: Nonfamily household: Female householder: Not living alone
hhincadj	Integer	The annual household income in the household, adjusted to the model base year
workers	Integer	The number of workers in the household
has_children	Boolean	1 if household has children

TABLE 12: SYNTHETIC PERSONS FILE FIELDS

NAME	TYPE	DESCRIPTION
puma_geoid	Integer	Household PUMA Geographic ID
TAZ	Integer	Household TAZ ID

household_id	Integer	Household ID number (must agree with household file)
per_num	Integer	Number of persons in household
AGEP	Integer	Age of person in years
SEX	Integer	Sex of person in household (1 = male, 2 = female)
WKHP	Integer	Usual hours worked per week in past 12 months
WKW	Integer	Weeks worked during past 12 months N/A: under 16 years old or did not work in past 12 months 1: 50-52 weeks worked 2: 48-49 weeks worked 3: 40-47 weeks worked 4: 27-39 weeks worked 5: 14-26 weeks worked 6: 13 or fewer weeks worked
ESR	Integer	Employment Status Recode N/A: Under 16 years old 1: Civilian employed, at work 2: Civilian employed, employed but not at work 3: Unemployed 4: Armed forces, at work 5: Armed forces, employed but not at work 6: Not in labor force
RAC1P	Integer	Race identifier 1 White alone 2 Black or African American alone 3 American Indian alone 4 Alaska Native alone 5 American Indian and Alaska Native tribes specified; or American Indian or Alaska native, not specified and no other races 6 Asian alone 7 Native Hawaiian and Other Pacific Islander alone

		8	Some other race alone
		9	Two or more major race groups
			Hispanic identifier
		1	Not Spanish/Hispanic/Latino
		2	Mexican
		3	Puerto Rican
		4	Cuban
		5	Dominican
		6	Costa Rican
		7	Guatemalan
		8	Honduran
		9	Nicaraguan
HISP	Integer	10	Panamanian
		11	Salvadoran
		12	Other Central American
		13	Argentinean
		14	Bolivian
		15	Chilean
		16	Colombian
		17	Ecuadorian
		19	Peruvian
		20	Uruguayan
		21	Venezuelan

22	Other South American
23	Spaniard
24	All Other Spanish/Hispanic/Latino

SCHG	Integer	School grade N/A: Not in school 1: Preschool 2: Kindergarten 3-14: grades 1-12 15: Undergrad college 16: College beyond bachelor's degree
MIL	Integer	Military Service N/A: under 17 years old 1: On active duty 2: Active in past, but not currently 3: Only active for training in Reserves/National Guard 4: Never served in military
NAICSP	Integer	NAICS ID of work occupation (Census Code)
INDP	Integer	Work industry code (Census code)
person_id	Integer	Person ID number used as index

Supporting Data

Supporting data is held in the support folder, and includes speed lookup files, volume-delay function parameters, toll equivalency parameters, and other parameters used in the model. These files are listed in Table 13.

TABLE 13: MODEL SUPPORTING DATA FILES

FILENAME	DESCRIPTION
AM_SPD_LKP.txt	Peak-period speed lookup file
cvdelta_3722.trp	Commercial vehicle calibration adjustment tables (used in auxiliary time-of-day script)
equiv_toll_min_by_inc.s	Equivalent toll minutes by income script file
hwy_assign_capSpeedLookup.s	Script to apply speed-capacity lookup
hwy_assign_conical_vdf.s	Script to apply volume-delay functions
MD_SPD_LKP.txt	Off-peak period speed lookup file
tkdelta_3722.trp	Truck calibration adjustment tables (used in auxiliary time-of-day script)
todixxi_2018HTS.dbf	Time-of-day factors by purpose, mode, direction, and time period
Toll_Minutes.TXT	Equivalent toll minutes lookup script
Truck_Com_Trip_Rates.DBF	Truck vehicle calibration adjustment tables (used in auxiliary time-of-day script)
True_Shape_2040_Nov20.shp (and dbf, shx, sbx, sbn, prj)	Network true-shape display file used in Cube when viewing network files. This file is solely for viewing the network and is not used during the model run. This file may change names (e.g. True_Shape_2045) based on MWCOG's long range plan.
Ver23_f_factors.dbf	Distribution friction factors (used in some of the auxiliary trip models)

Transit Data

Transit data is in the "trn" folder and includes all of the transit model input files. These files define the alignment of bus routes, rail routes and stops, fare inputs, system inputs, and factor inputs. Most of the transit input files are coded in the Cube Public Transport (PT) format as the Gen3 Model uses PT for transit modeling. The PT system and factor input files define what fare systems belong to what routes, and how people can move within the transit network - including when transfers can be made, how long transfers take, and how the transit systems appear in comparison to driving. The transit input files, including transit network input files (.lin), are listed in Table 14. The PT based transit skimming and assignment processes are explained in Section 6.5. Transit assignment is conducted in origin-destination (O-D) format for four time-of-day periods.

TABLE 14: TRANSIT INPUT FILES

FILENAME	DESCRIPTION
AM_TRN.FAC	AM transit factor input
AM_TRN_BM.FAC	AM bus-metro transit factor input
AM_TRN_CR.FAC	AM commuter rail transit factor input
Bus_Factor_File.dbf	Local Bus Time Degradation Factors. See Table 15.
COM_BUS.TB	List of Commuter Rail KNR Driveways added to the transit network during transit network processing
COM_LINK.TB	List of Commuter Rail lines added to the transit network during transit network processing. These lines are not included on the input highway network.
COM_NODE.TB	List of Commuter Rail nodes added to the transit network to represent stations
fare_am.dat	AM fare setup file
fare_md.dat	MD fare setup file
fare_nt.dat	NT fare setup file
fare_pm.dat	PM fare setup file
LRT_BUS.TB	List of Light Rail KNR Driveways added to the transit network during transit network processing. Note that this file is currently empty.
LRT_LINK.TB	List of Light Rail lines added to the transit network during transit network processing. These lines are not included on the input highway network. Note that this file is currently empty.

LRT_NODE.TB	List of Light Rail nodes added to the transit network to represent stations. Note that this file is currently empty.
MARC_Fare.dat	MARC train fare setup file
MD_TRN.FAC	AM transit factor input
MD_TRN_BM.FAC	AM bus-metro transit factor input
MD_TRN_CR.FAC	AM commuter rail transit factor input
MetLnkM1.TB	Metrorail link file
MetNodM1.TB	Metrorail node file
MET_BUS.TB	List of Metrorail KNR Driveways added to the transit network during transit network processing
MET_LINK.TB	List of Metrorail lines added to the transit network during transit network processing. These lines are not included on the input highway network.
MET_NODE.TB	List of Metrorail nodes added to the transit network to represent stations.
MFare1.a1	Metrorail Station XYs scaled to 1/100ths of miles
mfare1_Sta_Disc.ASC	Metrorail Station fare discount array in cents
MODE{m}{p}.LIN	Transit line input files, four periods ({p} - AM, MD, PM, and NT) for each mode ({m}): 1: WMATA Local Bus 2: WMATA Express Bus 3: Metrorail 4: Commuter Rail (Amtrak, MARC, VRE) 5: Light Rail 6: Other local bus in WMATA service area 7: Other express bus in WMATA service area 8: Other local bus outside of WMATA service area 9: Other express bus outside of WMATA service area 10: Bus Rapid Transit and Streetcar

NEW_BUS.TB	New mode (streetcar, BRT, etc.) line input file
NEW_LINK.TB	New mode (streetcar, BRT, etc.) link input file
NEW_NODE.TB	New mode (streetcar, BRT, etc.) node input file
NT_TRN.FAC	NT transit factor input
NT_TRN_BM.FAC	NT bus-metro transit factor input
NT_TRN_CR.FAC	NT commuter rail transit factor input
PM_TRN.FAC	PM transit factor input
PM_TRN_BM.FAC	PM bus-metro transit factor input
PM_TRN_CR.FAC	PM commuter rail transit factor input
Rail_Links.DBF	Metrorail link input file. See Table 16.
StaAcc.dbf	Rail mode, access type, and access distance input table
STATION.DBF	Station data input file. See Table 17.
station_names.dbf	List of station ID and name
station_names.dbf.def	Definition file for above.
tariff.txt	Metrorail tariff script input
TRNPEN.DAT	Transfer penalty input file
TRN_Deflator.txt	Transit deflation factor input file
TSYSD.PTS	Transit system input file
VRE_Fare.dat	Virginia Rail fare input file

TABLE 15: BUS FACTOR FILE FORMAT

NAME	TYPE	DESCRIPTION

YEAR	Integer	Year of analysis
AMIBFTR	Decimal	AM inbound time degradation
AMOBFTR	Decimal	AM outbound time degradation
MDIBFTR	Decimal	MD inbound time degradation
MDOBFTR	Decimal	MD outbound time degradation
PMIBFTR	Decimal	PM inbound time degradation
PMOBFTR	Decimal	PM outbound time degradation
NTIBFTR	Decimal	NT inbound time degradation
NTOBFTR	Decimal	NT outbound time degradation

TABLE 16: RAIL LINK FILE FORMAT

NAME	TYPE	DESCRIPTION
A	Integer	A node of link
B	Integer	B node of link
MODE	Integer	Transit mode (always 3)
SPEED	Integer	Speed in MPH
DISTANCE	Decimal	Distance in miles
TRANTIME	Integer	Transit Time in minutes

TABLE 17: STATION INPUT FILE FORMAT

NAME	TYPE	DESCRIPTION
OBJECTID	Integer	Unique ID number
SEQNO	Integer	Sequence number
MM	Text	Mode code (M for Metrorail, C for commuter rail, B for bus)

NCT	Integer	Station Access distance code
STAPARK	Text	If station has parking (Y or N)
STAUSE	Text	Is the station in use (Y or N/Blank)
SNAME	Text	Name of station
STAC	Integer	Station access node number
STAZ	Integer	Station Zone Number
STAT	Integer	Station stop node number (corresponds to rail network for MR and CR)
STAP	Integer	Station parking highway node number
STAN1	Integer	Station link access node number
STAN2	Integer	Station link access node number
STAN3	Integer	Station link access node number
STAN4	Integer	Station link access node number
STAPCAP	Integer	Station parking capacity
STAX	Integer	Station location x coordinate
STAY	Integer	Station location y coordinate
STAPKCOST	Integer	Station parking cost
STAOPCOST	Integer	Station off-peak parking cost
STAPKSHAD	Integer	Station peak shadow price (All 0)
STAOPSHAD	Integer	Station off-peak shadow price (All 0)
FIRSTYR	Integer	First year of opening
STA_CEND	Integer	Station project ID
STWALKTM	Decimal	Station walk time in minutes

External and Visitor Transit Data

External and visitor transit data and external transit matrices are in the trn\External_Visitor_Transit folder. There are 16 files in the folder, all have a naming pattern that includes the period, external or visitor, and an extension of OMX or TRP. The inputs are OMX files, the model will convert the OMX files to TRP (Cube Voyager Matrix) files, overwriting any existing TRP files. These files are listed in Table 18. Each matrix has 18 matrix tables, which are the access mode, an underscore, and the line haul model. The access modes are:

- WALK (walk access)
- PNR (park-n-ride)
- PNRE (park-n-ride egress)
- KNR (kiss-n-ride)
- KNRE (kiss-n-ride egress)

The line-haul modes are:

- AB (all-bus)
- MR (Metrorail only)
- BM (bus and Metrorail)
- CR (commuter rail)

These files are not adjusted in the model to account for growth - that must be done prior to the model run. The transit processing scripts will remove invalid paths in the commuter rail mode assignment.

TABLE 18: EXTERNAL AND VISITOR TRANSIT INPUT FILES

FILE	DESCRIPTION
{p}_External.{OMX/TRP}	Input external transit trip tables
{p}_Visitor.{OMX/TRP}	Input visitor transit trip tables

Visualizer

The visualizer folder holds a summary of input files used by the ABM Visualizer. There are numerous files in some of these folders, and they should be kept together.

The base and build folders are the specific input summaries used by the visualizer. The Census folder has census data used in the visualizer for some situations (auto ownership, for example).

JPEG is the image source folder for the visualizer. SHP is the source folder for shapefiles. The summarized_{name} folders are the summarized data inputs from the summarization scripts. During the visualizer build process, the necessary files are copied from the summarized_{name} folder to the base or build folder as appropriate.

In the Visualizer folder, there is a shapefile for zero-auto households by Census Tract (ct_zero_auto.shp+). This is used in the Visualizer for the zero-auto household comparison map.

3.2 MODEL OUTPUTS

The model outputs are divided into nine folders based on the model component that generated the outputs. These folders are described in more detail below. Note that some less important model outputs may be moved to the {scenario}\temp_files folder to be deleted later to save storage space.

The model outputs folder is divided into several folders and also includes a visualizer html file. All of these are described in this section.

The model outputs reflect the outputs of various processes on the inputs based on the parameters calibrated to various data sources. As such, output data reliability is a function of input data reliability and model reliability from the calibration data. There are many sources of uncertainty when using model outputs for future year forecasts and users should exercise caution when using outputs for future years.

Model Visualizer

In the root of the model output folder is the ABM visualizer. This HTML file is built with RMarkdown and Flexdashboard to show an interactive dashboard of ActivitySim outputs. The visualizer opening screen is shown in Figure 4. This visualizer is organized into five main groups: Welcome, Overview, Long Term, Tour Level, and Trip Level. The welcome screen is a simple map. Overview is a screen that includes basic summary information of the model and compares it to the survey or a base model. Long Term shows various long term choice comparisons, such as auto-ownership, the distance to work, university, or school, and where zero-auto households are located. The Tour Level section displays comparisons related to day models (such as the Coordinated Daily Activity Pattern model output) as well as those for tours (such as tour mode choice). The trip model shows trip model comparisons. The visualizer is automatically created at the end of the model run.

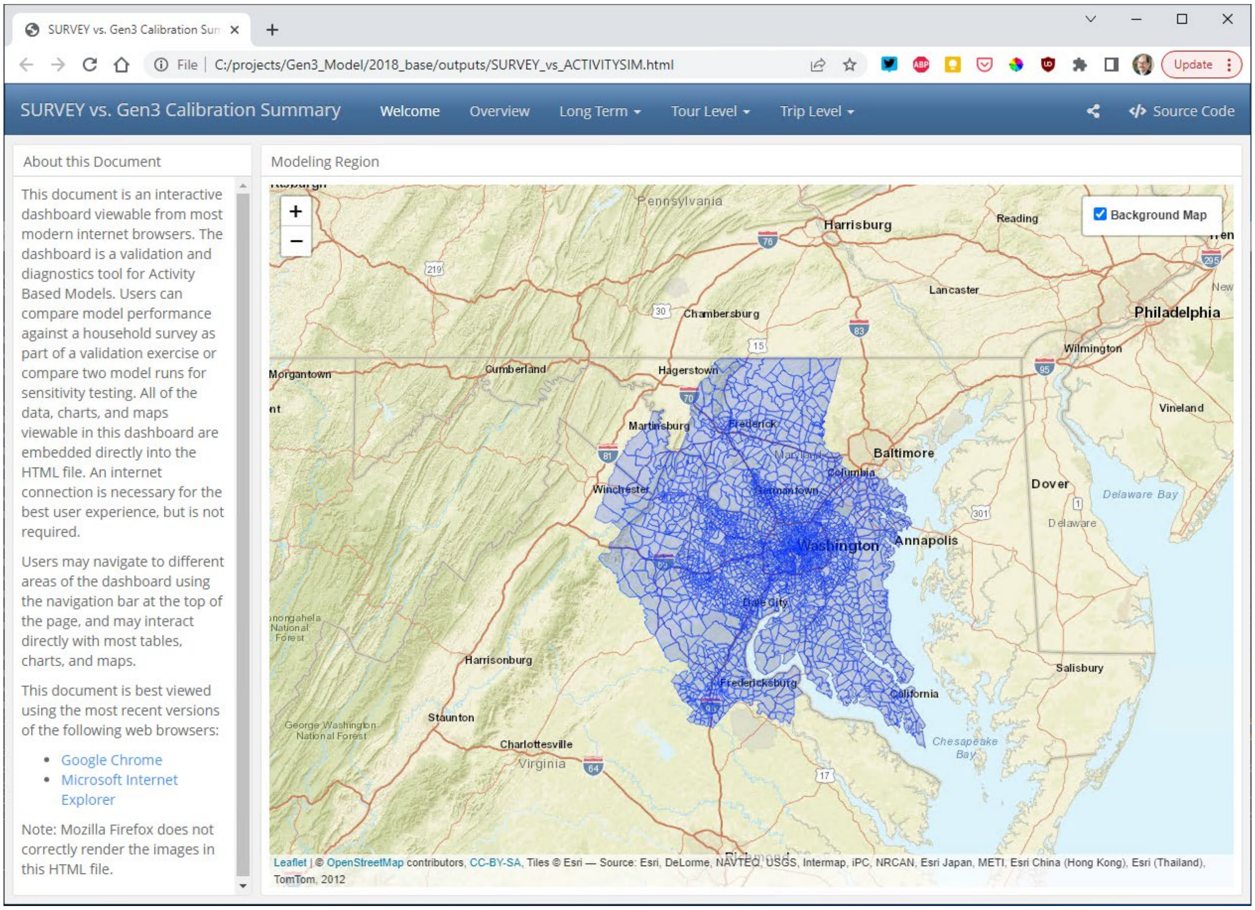


FIGURE 4: ABM VISUALIZER

ActivitySim

The ActivitySim folder contains output from running ActivitySim. Table 19 shows the contents of the folder in more detail.

TABLE 19: ACTIVITYSIM MODEL OUTPUT FILES

FILE	DESCRIPTION
activitysim.log	Model run progress terminal output
data_dict.csv	List of variables in each model dataframe
final_accessibility.csv	Accessibility measures created by ActivitySim. See Table 24.
final_checkpoints.csv	This is an internal ActivitySim file that relates to the pipeline files
final_households.csv	Synthetic households file with the additional variables created during model run. See Table 20.

final_joint_tour_participants.csv	Joint tour participants output file. See Table 25.
final_land_use.csv	Land use file with the additional variables created during model run. See Table 22.
final_persons.csv	Synthetic persons file with the additional variables created during model run. See Table 21.
final_tours.csv	All the tours created by ActivitySim and their attributes. See Table 26.
final_trips.csv	All the trips created by ActivitySim and their attributes. See Table 27.
final_vehicles.csv	Modeled household vehicle output file. See Table 28.
mp_accessibility_{n-}activitysim.log	Model terminal output per process n for running the accessibility step
mp_households_{n}-activitysim.log	Model terminal per process n for running the model
timing_log.csv	Run time for each step
Auto_trips_{per}.omx	Auto trip matrices for each period (am, md, pm, and nt). Includes
Nm_trips_{per}.omx	Non-motorized trips (walk and bike) for each period (am, md, pm, and nt). Each file includes two matrix tables, WALK and BIKE.
Sb_trips_{per}.omx	School bus trip matrices for each period (am, md, pm, and nt). Each file includes one matrix table, SCHOOLBUS.
Trn_trips_{per}.omx	Transit trip matrices for each period (am, md, pm, and nt). Each file includes 18 matrix tables: • WALK_AB (walk access to all-bus) • WALK_BM (walk access to bus+Metrorail) • WALK_MR (walk access to Metrorail) • WALK_CR (walk access to Commuter Rail) • PNR_AB (PNR access to all-bus) • PNR_BM (PNR access to bus+Metrorail) • PNR_MR (PNR access to Metrorail) • PNR_CR (PNR access to Commuter Rail)

	• PNRE_AB (Walk access, PNR egress to all-bus) • PNRE_BM (Walk access, PNR egress to bus+Metrorail) • PNRE_MR (Walk access, PNR egress to Metrorail) • PNRE_CR (Walk access, PNR egress to Commuter Rail) • KNR_AB (KNR access to all-bus) • KNR_BM (KNR access to bus+Metrorail) • KNR_MR (KNR access to Metrorail) • KNR_CR (KNR access to Commuter Rail) • KNRE_AB (Walk access, KNR egress to all-bus) • KNRE_BM (Walk access, KNR egress to bus+Metrorail) • KNRE_MR (Walk access, KNR egress to Metrorail) • KNRE_CR (Walk access, KNR egress to Commuter Rail)
trips_period.omx	Internal OD matrices per period

The final_households.csv table contains all the original synthetic household population in addition to the ones created in ActivitySim. Table 20 below shows the additional attributes of households created by ActivitySim.

TABLE 20: FINAL_HOUSEHOLDS.CSV FIELDS

NAME	TYPE	DESCRIPTION
household_id	Integer	Unique ID number of household (from population synthesizer)
puma_geoid	Integer	GEOID of Census Public Use Microdata Area where the household resides
home_zone_id	Integer	TAZ ID of home location
TYPE	Integer	Type of housing unit 1 = Household 2 = Institutional group quarters 3 = Noninstitutional group quarters
hhsize	Integer	Household size (number of persons in household)

NAME	TYPE	DESCRIPTION
auto_ownership	Integer	Auto ownership level
HHT	Integer	The type of household - Census PUMS HHT variable: N/A: Not applicable (group quarters or vacant) 1: Married couple household 2: Other family household: Male householder, no spouse present 3: Other family household: Female householder, no spouse present 4: Nonfamily household: Male householder: Living alone 5: Nonfamily household: Male householder: Not living alone 6: Nonfamily household: Female householder: Living alone 7: Nonfamily household: Female householder: Not living alone
hhincadj	Integer	Household income
workers	Integer	Number of workers in household
has_children	Integer	If the household has children
sample_rate	Decimal	Sampling rate of household (set to 1.0 for all records)
income	Integer	Household income (hhincadj with NA values converted to 0)
income_in_thousands	Integer	Household income in thousands
income_segment	Integer	4-level segmented income: low(<$50K), medium (>$50K, <$100K), high (>$100K, <$150K), very high (>$150K)
median_value_of_time	Decimal	Median household value of time (according to income)
hh_value_of_time	Decimal	Household value of time

NAME	TYPE	DESCRIPTION
num_workers	Integer	Number of workers in the household
num_non_workers	Integer	Number of non-workers in the household
num_drivers	Integer	Number of drivers in the household
num_adults	Integer	Number of adults in the household
num_children	Integer	Number of children in the household
num_young_children	Integer	Number of young children in the household
num_children_5_to_15	Integer	Number of children aged 5-15 yrs in the household
num_children_6_to_12	Integer	Number of children aged 6-12 yrs in the household
num_children_16_to_17	Integer	Number of children aged 16-17 yrs in the household
num_college_age	Integer	Number of college age children in the household
num_young_adults	Integer	Number of young adults in the household
non_family	Integer	Non-family household
family	Integer	Family household
home_is_urban	Integer	Home is in urban area
home_is_rural	Integer	Home is in rural area
home_jurisdiction	Integer	Jurisdiction that the home resides in
TAZ	Integer	TAZ ID of home location
num_predrive_child	Integer	Number of predriving age children in household
num_nonworker_adults	Integer	Number of nonworker in adults area
num_fullTime_workers	Integer	Number of fulltime workers in the household
num_partTime_workers	Integer	Number partTime in workers in the household
retired_adults_only_hh	Integer	Households with only retired persons

NAME	TYPE	DESCRIPTION
hh_work_auto_savings_ratio	Decimal	Time saved driving vs. taking transit to work
av_ownership	Integer	If the household owns an automated vehicle
num_under16_not_at_school	Integer	Number of under 16 persons not in school in the household
num_travel_active	Integer	Number of persons who travel in the household (with M or N CDAP pattern)
num_travel_active_adults	Integer	Number of adults who travel in the household (with M or N CDAP pattern)
num_travel_active_preschoolers	Integer	Number of preschoolers who travel in the household (with M or N CDAP pattern)
num_travel_active_children	Integer	Number of children who travel in the household (with M or N CDAP pattern)
num_travel_active_non_preschoolers	Integer	Number of non-preschoolers who travel in the household (with M or N CDAP pattern)
participates_in_jtf_model	Integer	Household with joint tours
joint_tour_frequency	Integer	Joint tour in frequency of the household
num_hh_joint_tours	Integer	Number joint in tours a household makes

The final_persons.csv table contains all the original synthetic persons population in addition to the ones created in ActivitySim. Table 21 shows the additional attributes of persons created by ActivitySim.

TABLE 21: FINAL_PERSONS.CSV FIELDS

NAME	TYPE	DESCRIPTION
person_id	Integer	Person ID number used as index
puma_geoid	Integer	Household PUMA Geographic ID

NAME	TYPE	DESCRIPTION
TAZ	Integer	Household TAZ ID
household_id	Integer	Household ID number (must agree with household file)
per_num	Integer	Number of persons in household
age	Integer	Age of person in years
SEX	Integer	Sex of person in household (1 = male, 2 = female)
WKHP	Integer	Usual hours worked per week in past 12 months
WKW	Integer	Weeks worked during past 12 months N/A: under 16 years old or did not work in past 12 months 1: 50-52 weeks worked 2: 48-49 weeks worked 3: 40-47 weeks worked 4: 27-39 weeks worked 5: 14-26 weeks worked 6: 13 or fewer weeks worked
ESR	Integer	Employment Status Recode N/A: Under 16 years old 1: Civilian employed, at work 2: Civilian employed, employed but not at work 3: Unemployed 4: Armed forces, at work 5: Armed forces, employed but not at work 6: Not in labor force
RAC1P	Integer	Race identifier (not used in model)
HISP	Integer	Hispanic flag (not used in model)

NAME	TYPE	DESCRIPTION
SCHG	Integer	School grade N/A: Not in school 1: Preschool 2: Kindergarten 3-14: grades 1-12 15: Undergrad college 16: College beyond bachelor's degree
MIL	Integer	Military Service N/A: under 17 years old 1: On active duty 2: Active in past, but not currently 3: Only active for training in Reserves/National Guard 4: Never served in military
NAICSP	Integer	NAICS ID of work occupation (Census Code)
INDP	Integer	Work industry code (Census code)
age_0_to_5	Integer	Person's age is between 0 and 5
age_6_to_12	Integer	Person's age is between 6 and 12
age_16_to_19	Integer	Person's age is between 16 and 20
age_16_p	Integer	Person's age more than 16
adult	Integer	Person is an adult
young	Integer	Person is 25 or younger
old	Integer	Person is 65 or older
male	Integer	Person is male
female	Integer	Person is female
esr	Integer	ESR (above) with NA values set to zero
wkhp	Integer	WKHP (above) with NA values set to zero

NAME	TYPE	DESCRIPTION
wkw	Integer	WKW (above) with NA values set to zero
schg	Integer	SCHG (above) with NA values set to zero
mil	Integer	MIL (above) with NA values set to zero
pemploy	Integer	Employment type
pstudent	Integer	Student type
ptype	Integer	Person type category
has_non_worker	Integer	Person’s household has nonworker
has_retiree	Integer	Person’s household has retiree
has_preschool_kid	Integer	Person’s household has preschooler
has_driving_kid	Integer	Person’s household has driving age child
has_school_kid	Integer	Person’s household has school age child
has_full_time	Integer	Person’s household has full time worker
has_part_time	Integer	Person’s household has part time worker
has_university	Integer	Person’s household has a university student
student_is_employed	Integer	Person is a student and is employed
nonstudent_to_school	Integer	Person is a non-student ptype and pstudent indicates student
is_student	Integer	Person is a student
is_gradeschool	Integer	Persons is a gradeschooler
is_highschool	Integer	Person is a high schooler
is_university	Integer	Person is a university student
school_segment	Integer	School segment type (1: gradeschool, 2: highschool, 3: university)

NAME	TYPE	DESCRIPTION
is_worker	Integer	Person is a worker
home_zone_id	Integer	TAZ ID of home location
PNUM	Integer	Member ID of person
income	Integer	Household income
income_in_thousands	Integer	Household income in thousands
income_segment	Integer	4-level segmented income: low(<$50K), medium (>$50K, <$100K), high (>$100K, <$150K), very high (>$150K)
is_fulltime_worker	Integer	Person is full-time worker
is_parttime_worker	Integer	Person is part-time worker
is_out_of_home_worker	Integer	Person works out-of-home
value_of_time	Integer	Persons's value of time
is_income_less25K	Integer	Household income <$25K
is_income_25K_to_60K	Integer	Household income >$25K & <$60K
is_income_60K_to_120K	Integer	Household income >$60K & <$120K
is_income_greater60K	Integer	Household income >$60K
is_income_greater120K	Integer	Household income >$120K
is_non_worker_in_HH	Integer	Persons's household has a non-worker
is_all_adults_full_time_workers	Integer	All adults are full-time workers in the household
is_pre_drive_child_in_HH	Integer	Person's household has a predriving age child
has_young_children	Integer	Person's household has young children
has_children_6_to_12	Integer	Person's household has children aged between 6 and 12

NAME	TYPE	DESCRIPTION
hh_child	Integer	Number of children in the household
home_jurisdiction	Integer	Jurisdiction home is located in
school_zone_id	Integer	TAZ ID of school location (where applicable)
distance_to_school	Decimal	Distance in miles to school
roundtrip_auto_time_to_school	Decimal	Round trip auto travel time to school
noUsualSchoolLocation	Integer	Person has no usual school location
work_from_home	Integer	Person works from home
workplace_zone_id	Integer	TAZ ID of workplace location (where applicable)
workplace_location_logsum	Decimal	Workplace location logsum
distance_to_work	Decimal	Distance in miles to work
workplace_in_cbd	Integer	Person’s workplace zone is in CBD
work_zone_area_type	Integer	Area type of workplace zone
roundtrip_auto_time_to_work	Decimal	Round trip auto travel time to work
work_auto_savings	Decimal	Time saved driving vs. taking transit to work
work_auto_savings_ratio	Decimal	Scaled time saved driving vs. taking transit to work (-1,1)
usualWorkLocationIsHome	Integer	Person's usual work location is home
noUsualWorkLocation	Integer	Person has no usual work location
work_jurisdiction	Integer	Jurisdiction workplace is located in
work_park_cost	Decimal	Parking cost of work zone
transit_pass_subsidy	Integer	Person receives transit subsidy
free_parking_at_work	Integer	Person gets free parking at work

NAME	TYPE	DESCRIPTION
telecommute_frequency	String	Telecommute frequency
cdap_activity	String	CDAP pattern (M = Mandatory, N = Non-mandatory, H = Home/out of region)
travel_active	Integer	Person travels out of home
under16_not_at_school	Integer	Person is under 16 and does not go to school
has_preschool_kid_at_home	Integer	Person has preschool kid at home
has_school_kid_at_home	Integer	Person has school kid at home
mandatory_tour_frequency	Integer	Frquency of manadatory tours
work_and_school_and_worker	Integer	Persons is a worker and goes to work and school
work_and_school_and_student	Integer	Persons is a student and goes to work and school
num_mand	Integer	Number of mandatory tours for each person
num_work_tours	Integer	Number of work tours for each person
has_pre_school_child_with_mandatory	Integer	Presence of pre school kid with mandatory tours
has_driving_age_child_with_mandatory	Integer	Presense of driving age school children with mandatory tours
num_joint_tours	Integer	Number of joint tours for each person
non_mandatory_tour_frequency	Integer	Non-mandatory tours for each person
num_non_mand	Integer	Number of non mandatory tours
num_escort_tours	Integer	Number of escort tours
num_eatout_tours	Integer	Number of eatout tours
num_shop_tours	Integer	Number of shopping tours
num_maint_tours	Integer	Number of maintenance tours

NAME	TYPE	DESCRIPTION
num_discr_tours	Integer	Number of discretionary tours
num_social_tours	Integer	Number of social tours
num_non_escort_tours	Integer	Number of non-escort tours
num_shop_maint_tours	Integer	Number of shopping and maintenance tours
num_shop_maint_escort_tours	Integer	Number of shopping and maintenance and escort tours
num_add_shop_maint_tours	Integer	Number of additional shopping and maintenance tours
num_soc_discr_tours	Integer	Number of social and discretionary tours
num_add_soc_discr_tours	Integer	Number of additional social and discretionary tours

The `final_land_use.csv` table contains all the original land use fields in addition to the ones created in ActivitySim. The table below shows the additional land use attributes created by ActivitySim.

TABLE 22: FINAL_LAND_USE.CSV FIELDS

FIELD	TYPE	DESCRIPTION
zone_id	Integer	Zone ID
HH	Integer	Number of households
HHPOP	Integer	Household population
GQPOP	Integer	Group quarters population
TOTPOP	Integer	Total population
TOTEMP	Integer	Total employment
INDEMP	Integer	Industrial employment
RETEMP	Integer	Retail employment

OFFEMP	Integer	Office employment
OTHEMP	Integer	All other employment
JURCODE	Integer	Jurisdiction code
LANDAREA	Decimal	Land Area of TAZ in square miles
HHINCIDX	Integer	Household income index
ADISTTOX	Integer	Distance to external
TAZXCRD	Integer	TAZ X Coordinate
TAZYCRD	Integer	TAZ Y Coordinate
K_8	Integer	Kindergarten - 8th grade enrollment
G9_12	Integer	High school enrollment
COLLEGE	Integer	College enrollment
Park_Acres	Decimal	Park area of TAZ in acres
GC_Acres	Decimal	Golf course area of TAZ in acres
PRKCST	Integer	Parking cost in TAZ
OPRKCST	Integer	Off-peak parking cost in TAZ
TERMINAL	Integer	Terminal time for trip-ends in TAZ, in minutes
AREATYPE	Decimal	Area type of TAZ
household_density	Decimal	Total household divided by zone area
employment_density	Decimal	Total employment divided by zone area
density_index	Decimal	Density index defined as (household_density * employment_density)/(employment_density + household_density)
TOPOLOGY	Integer	Topology code (not used)
emp_adjust	Decimal	Employment adjustment factor accounting for work-from-home and in-migration

ADJ_INDEMP	Decimal	Adjusted industrial employment accounting for work-from-home and in-migration used in destination choice
ADJ_RETEMP	Decimal	Adjusted retail employment accounting for work-from-home and in-migration used in destination choice
ADJ_OTHEMP	Decimal	Adjusted other employment accounting for work-from-home and in-migration used in destination choice
ADJ_OFFEMP	Decimal	Adjusted office employment accounting for work-from-home and in-migration used in destination choice
ADJ_TOTEMP	Decimal	Adjusted total employment accounting for work-from-home and in-migration used in destination choice

The `final_accessibility.csv` table contains all the accessibility measures created in ActivitySim. Table 24 shows accessibility measures created by ActivitySim. The accessibility measure used in ActivitySim is a decay function that is shown in Equation 1.

EQUATION 1: ACTIVITYSIM ACCESSIBILITY FORMULA

$$A_i = ln\left(\sum_{j=1}^{Zones} size_j * e^{(I_{ij}*d)}\right)$$

Where:

A is the accessibility
Size is a size parameter, such as retail employment or total employment
I is an impedance parameter from zone i to zone j (usually time, see Table 23)
d is a dispersion parameter (see Table 23)

TABLE 23: ACCESSIBILITY IMPEDANCE VALUES AND DISPERSION PARAMETERS

MODE	IMPEDANCE VALUE	DISPERSION PARAMETER
Auto	Auto Time	-0.05
Transit	Transit Weighted Time	-0.05
Non-motorized	Walk Distance	-1.0

TABLE 24: FINAL_ACCESSIBILITY.CSV FIELDS

FIELD	TYPE	DESCRIPTION
zone_id	Integer	MAZ zone ID
auPkRetail	Decimal	Destination retail accessibility by auto at peak time
auPkTotal	Decimal	Destination total accessibility by auto at peak time
auOpRetail	Decimal	Destination retail accessibility by auto at off-peak time
auOpTotal	Decimal	Destination total accessibility by auto at off-peak time
trPkRetail	Decimal	Destination retail accessibility by transit at peak time
trPkTotal	Decimal	Destination total accessibility by transit at peak time
trPKHH	Decimal	Destination residential accessibility by transit at peak time
trOpRetail	Decimal	Destination retail accessibility by transit at off-peak time
trOpTotal	Decimal	Destination total accessibility by transit at off-peak time
nmRetail	Decimal	Destination retail accessibility by non-motorized modes
nmTotal	Decimal	Destination total accessibility by non-motorized modes
auShare	Decimal	Regional auto share constant
trShare	Decimal	Regional transit share constant
nmShare	Decimal	Non-motorized share constant
TotalAcc	Decimal	Total accessibility

The `final_joint_tour_participants.csv` table contains data on the joint tours in ActivitySim. Table 25 shows these data.

TABLE 25: FINAL_JOINT_TOUR_PARTICIPANTS.CSV FIELDS

FIELD	TYPE	DESCRIPTION
participant_id	Integer	Persons ID of participant
tour_id	Integer	Tour ID
household_id	Integer	Household ID
person_id	Integer	Person ID of participant
participant_num	Integer	Number of persons on tour

The `final_tours.csv` table contains all the information on the tours created in ActivitySim. Table 26 shows the attributes of ActivitySim tours.

TABLE 26: FINAL_TOURS.CSV FIELDS

NAME	TYPE	DESCRIPTION
tour_id	Integer	Tour ID
person_id	Integer	Person id
tour_type	String	Tour type: work, school, othmaint, social, eatout, shopping, othdiscr, escort, eat, maint, business
tour_type_count	Integer	Number of tours of tour_type parent has (parent's max tour_type_num)
tour_type_num	Integer	If there are multiple of the same type tours, they will be numbered
tour_num	Integer	Index of tour (of any type) for parent tour
tour_count	Integer	Number of tours of any type for parent (parent's max tour_num)
tour_category	String	Category of tour. One of 'mandatory', 'non_mandatory', 'atwork', or 'joint'
number_of_participants	Integer	Number of participants on a tour

destination	Integer	TAZ ID for tour destination
origin	Integer	TAZ ID for tour origin
household_id	Integer	Household ID of the person making the tour
tdd	Integer	ID of the in-outbound bin combination
start	Integer	Start half hour bin of tour, ranging 1-48
end	Integer	End half hour bin of tour, ranging 1-48
duration	Integer	Tour duration in half hour unit
composition	String	Composition of a joint tour: all adults, all children, mixed
destination_logsum	Decimal	Tour destination logsum
tour_mode	String	Tour mode
mode_choice_logsum	Decimal	Mode choice logsum
selected_vehicle	String	Selected vehicle for tour
atwork_subtour_frequency	Integer	Frequency of atwork subtours for workers
parent_tour_id	Integer	ID of the parent tour if current tour is a subtour
stop_frequency	Integer	Number of stops on a tour
primary_purpose	String	Primary purpose of the tour: work, school, univ, othmaint, social, eatout, shopping, othdiscr, escort, atwork

The `final_trips.csv` table contains all the information on the trips created in ActivitySim. Table 27 shows the attributes of ActivitySim trips.

TABLE 27: FINAL_TRIPS.CSV FIELDS

FIELD		DESCRIPTION
trip_id	Integer	Trip ID

person_id	Integer	ID of the person trip belongs to
household_id	Integer	ID of the household person belongs to
primary_purpose	String	Purpose of the tour trip belongs to: work, school, univ, othmaint, social, eatout, shopping, othdiscr, escort, atwork
trip_num	Integer	Index of trip in the tour
outbound	Integer	Outbound trip direction
trip_count	Integer	Number of trips on a tour
destination	Integer	MAZ ID for trip destination
origin	Integer	MAZ ID for trip origin
tour_id	Integer	ID of tour trip belongs to
purpose	Integer	Trip purpose: othmaint, home, othdiscr, atwork, work, social, eatout, shopping, escort, school, univ, parking
destination_logsum	Decimal	Trip destination logsum
depart	Integer	Departure half hour bin: 1-48
trip_mode	String	Trip mode
mode_choice_logsum	Decimal	Trip mode choice logsum

TABLE 28: FINAL_VEHICLES.CSV FIELDS

FIELD		DESCRIPTION
vehicle_id	Integer	Vehicle ID
household_id	Integer	ID of the household the vehicle belongs to
vehicle_num	Integer	ID number of the vehicle in the household
vehicle_type	String	Type of vehicle. These are listed as BODY_AGE_FUEL, where BODY indicates the type of vehicle (Car, SUV, Truck, etc), AGE is the age in years,

		and FUEL is the fuel source (Gas, Hybrid, Diesel, BEV, PEV)
auto_operating_cost	Integer	The operating cost of the vehicle
Range	Integer	The range of the vehicle
MPG	Integer	The fuel efficiency (in miles per gallon)
is_av	Integer	If the vehicle is an autonomous vehicle

Auxiliary

The auxiliary folder includes model outputs for miscellaneous trips, which includes truck trips, commercial vehicle trips, auto-driver external-internal trips, auto-driver internal-external trips, through trips (auto driver, commercial vehicle, and trucks), airport auto-driver trips, supplemental taxi trips, and visitor trips. The output files for this model are listed in Table 29.

TABLE 29: AUXILIARY MODEL OUTPUT FILES

FILENAME	DESCRIPTION
autopaxixxi.trp	Auto driver Internal-external and external-internal trip matrix
COM.SQZ	Jurisdiction-Jurisdiction commercial vehicle trip matrix
COM.TEM	Temporary commercial vehicle trip matrix
COMext.TEM	Temporary external-internal and internal-external commercial vehicle trip matrix
DJ.EQV	Jurisdiction - TAZ equivalency file (script input file)
ext.tem	Temporary exernal auto driver trip matrix
ExternalPsAs.dbf	External Productions and Attractions by zone
HTK.SQZ	Jurisdiction-Jurisdiction heavy truck trip matrix
HTK.TEM	Temporary heavy truck trip matrix
HTKext.TEM	Temporary external heavy truck tripmatrix

FILENAME	DESCRIPTION
{iter}_{pp}_adr.mat	Auto-driver external-internal and internal-external trips for iteration {iter} and period {pp}. Includes three matrix tables, {pp}_ADRs_1 (auto driver trips), {pp}_ADRs_2 (unused), {pp}_ADRs_3 (unused).
{iter}_{pp}_misc.tt	Miscellaneous trips for iteration {iter} and period {pp}. Includes 9 matrix tables: {pp}_XXTrk (truck through trips), {pp}_XXAdr (auto through trips), {pp}_TxAde (supplemental taxi trips), {pp}_VtAdr (visitor trips), {pp}_ScAdr (supplimental school trips, not used), {pp}_MedTk (medium truck trips for assignment), {pp}_HvyTk (heavy truck trips for assignment), {pp}_APAdr (airport trips), {pp}_ComVe (commercial vehicle trips for assignment).
{iter}_COMext.VTT	External-internal and internal-external commercial vehicle trip matrix for iteration {iter}
{iter}_COMMER.PTT	Internal commercial vehicle trip matrix for iteration {iter}. Includes three tables, COM_Int (internal trips), COM_Ext (internal-external and external-internal trips), and COMAllVeh (sum of both matrices)
{iter}_ComVeh_Truck_dbg.dbf	Output debug file for iteration {iter}, includes zonal inputs and outputs
{iter}_ComVeh_Truck_Ends.dbf	Truck and commercial vehicle trip ends by zone for iteration {iter}
{iter}_Ext_CVTruck_Gen_PsAs.dbf	External-internal and internal-external commercial vehicle and truck trip ends for iteration {iter}
{iter}_Ext_CVTruck_Gen_PsAs.txt	Summary report of external truck and commercial vehicle trip productions and attractions for iteration {iter}
{iter}_Final_Int_Motor_PsAs.dbf	Final internal truck and commercial vehicle trip ends by zone for iteration {iter}
{iter}_HTKext.VTT	External-internal and internal-external heavy truck matrix for iteration {iter}

FILENAME	DESCRIPTION
{iter}_HTRUCK.PTT	Internal heavy truck matrix for iteration {iter}. Includes three tables, HTK_Int (internal trips), HTK_Ext (internal-external and external-internal trips), and HTKAllVeh(sum of both matrices)
{iter}_MTKext.VTT	External-internal and internal-external medium truck matrix for iteration {iter}
{iter}_MTRUCK.PTT	Internal medium truck matrix for iteration {iter}. Includes three tables, MTK_Int (internal trips), MTK_Ext (internal-external and external-internal trips), and MTKAllVeh (sum of both matrices)
{iter}_Prepare_Internal_Ends.txt	Internal PA trip report for truck and commercial vehicle trips for iteration {iter}
{iter}_Truck_Com_Trip_Generation.txt	Internal truck and commercial trip end report for iteration {iter}
ixxidcutils.mat	Auto driver Internal-external and external-internal destination choice utilities matrix
ixxi_access.dbf	Auto driver Internal-external and external-internal accessibility
ixxi_prod.dbf	Auto driver Internal-external and external-internal production and attraction file
MTK.SQZ	Jurisdiction-Jurisdiction medium truck matrix
MTK.TEM	Temporary medium truck matrix
MTKext.TEM	Temporary external medium truck matrix
trip_distribution_extTrk.rpt	External truck and commercial vehicle trip distribution report
trip_distribution_intTrk.rpt	Internal truck and commercial vehicle trip distribution report

Highway Assignment

The highway assignment outputs are in the hwy_assign folder. The files in this folder are output for each iteration and include vehicle trip tables and text-based loaded network files. The files are listed in Table 30.

TABLE 30: HIGHWAY ASSIGNMENT OUTPUT FILES

FILENAME	DESCRIPTION
{pp}CHK.LKLOOP	Period {pp} link file
{iter}_{pp}.VTT	Vehicle trip tables for assignment, iteration {iter} and period {pp}
{iter}_{pp}_load_link.asc	Output loaded link files for iteration {iter} and period {pp}
{iter}_Asim_{pp}.VTT	Output vehicle trip tables from ActivitySim for iteration {iter} and period {pp} (converted from ActivitySim OMX files). Includes three matrix tables, SOV, SHARED2, and SHARED3.
{iter}_ue_iteration_report_{AM/PM}_hov.txt	Traffic assignment report for AM/PM HOV assignment
{iter}_ue_iteration_report_{AM/PM}_nonHov.txt	Traffic assignment report for AM/PM non-HOV assignment
{iter}_ue_iteration_report_{MD/NT}.txt	Traffic assignment report for MD/NT assignment

Land Use

The land use outputs are in the landuse folder. These files are outputs of land use processing scripts.

TABLE 31: LAND USE OUTPUT FILES

FILENAME	DESCRIPTION
AreaType_File.dbf	Computed area types. See Table 32.

AreaType_File.txt	Computed area type summary report.
Floating_LU.dbf	Computed 1-mile floating population and employment density. See Table 33.
TAZ_XYs.dbf	TAZ X and Y locations. See Table 34.
ZONEV2.A2F	Zonal parking and terminal times. See Table 35.
ztermtm.asc	Computed zonal terminal time report file.

TABLE 32: AREATYPE_FILE FIELDS

NAME	TYPE	DESCRIPTION
TAZ	Integer	Zone ID
POP_10	Integer	Population within 1 mile
EMP_10	Integer	Employment within 1 mile
AREA_10	Decimal	Land area within 1 mile
POPDEN	Decimal	Floating 1 mile population density
EMPDEN	Decimal	Floating 1 mile employment density
POPCODE	Integer	Population density classification
EMPCODE	Integer	Employment density classification
ATYPE	Integer	Area type

TABLE 33: FLOATING_LU FIELDS

NAME	TYPE	DESCRIPTION
TAZ	Integer	Zone ID
HH00	Integer	Households within 0 miles (zonal households)
POP00	Integer	Population within 0 miles (zonal population)

EMP00	Integer	Employment within 0 miles (zonal employment)
AREA00	Decimal	Area within 0 miles (zonal area)
HH10	Integer	Households within 1 mile
POP10	Integer	Population within 1 mile
EMP10	Integer	Employment within 1 mile
AREA10	Integer	Land area within 1 mile

TABLE 34: TAZ_XYS FIELDS

NAME	TYPE	DESCRIPTION
N	Integer	Zone ID
X	Integer	X coordinate of TAZ centroid
Y	Integer	Y coordinate of TAZ centroid

TABLE 35: ZONEV2 FIELDS

NAME	TYPE	DESCRIPTION
I	Integer	Zone ID
HBWParkCost	Integer	Home-based Work trip parking cost in cents
HBSParkCost	Integer	Home-based Shopping trip parking cost in cents
HBOParkCost	Integer	Home-based Other trip parking cost in cents
NHBParkCost	Integer	Non-home-based trip parking cost in cents
HB_TermTime	Integer	Terminal time for home-based trips
NHB_TermTime	Integer	Terminal time for nonhome-based trips

Reports

The reports folder includes various reports for the Cube processes of the model, as well as a copy of all messages sent to the screen during the model run. These files are listed in Table 2928.

In general, if there is a problem with the non-ActivitySim portion of the model, the last files written to this folder will show what happened. The fulloutput file should display an error message or an error location, and any Cube reports would show the error.

TABLE 29: REPORTS FOLDER OUTPUT FILES

FILENAME	DESCRIPTION
{scenario_dir}_fulloutput.txt	Copy of full output to the screen
Adjust_Runtime.rpt	Cube report
AreaType_File.rpt	Cube land use processing report
CheckStationAccess.rpt	Cube transit network report
HWY_Deflator.txt	Cube highway network report
{iter}_Highway_Assignment.rpt	Cube highway assignment report
{iter}_Highway_Skims_4TOD.rpt	Cube highway skimming report
{iter}_Highway_Skims_Mod_4TOD.rpt	Cube highway skimming report
{iter}_IXXI_TripGeneration.rpt	Cube auxiliary model report
{iter}_Misc_Time-of-Day.rpt	Cube auxiliary model report
{iter}_ModNet.rpt	Cube highway network report
{iter}_Prepare_Ext_ComTruck_Ends.rpt	Cube auxiliary model report
{iter}_Prepare_Int_ComTruck_Ends.rpt	Cube auxiliary model report
{iter}_Prepare_Trip_Tables_for_Assignment.rpt	Cube highway assignment preparation report
{iter}_Prepare_Trip_Tables_for_Assignment.tab	Cube highway assignment preparation report

{iter}_Prepare_Trip_Tables_For_Assignment.txt	Cube highway assignment preparation report
{iter}_TRANSIT_SKIMS_AB.RPT	Cube transit skimming report
{iter}_TRANSIT_SKIMS_BM.RPT	Cube transit skimming report
{iter}_TRANSIT_SKIMS_CR.RPT	Cube transit skimming report
{iter}_TRANSIT_SKIMS_MR.RPT	Cube transit skimming report
{iter}_Trip_Distribution_ExtTrk.rpt	Cube auxiliary model report
{iter}_Trip_Distribution_IntTrk.rpt	Cube auxiliary model report
{iter}_Truck_Com_Trip_Generation.rpt	Cube auxiliary model report
{iter}_Average_Link_Speeds.rpt	Cube highway network report
{iter}_Transit_Assgn_CR.RPT	Cube transit assignment report
MFARE2_CPI.TXT	Cube transit network report
mod2.rpt	Cube transit network report
pp_initial_Highway_Skims_4TOD.rpt	Cube highway skimming report
pp_initial_Highway_Skims_Mod_4TOD.rpt	Cube highway skimming report
pp_initial_ModNet.rpt	Cube highway network report
pp_initial_Remove_PP_Speed.rpt	Cube highway network report
pp_Metrorail_skims.rpt	Cube transit skimming report
pp_MFARE1.rpt	Cube transit network report
pp_prefarV23.rpt	Cube transit network report
Prepare_non_motorized_skims.rpt	Cube non-motorized network skimming report
PT_NetProcess_PT.rpt	Cube transit network report
set_CPI.rpt	Cube CPI processing report

TRN_Deflator.txt	Cube transit network report
Truck_Com_Trip_Generation.rpt	Cube auxiliary model report
V2.3_highway_build.rpt	Cube highway network report
V2.5_PTNet_Build.rpt	Cube transit network report
V2.5_PTNet_Build_Iteration.rpt	Cube transit network report

Skims

The skims folder contains nearly 1,500 skim files. These files are all named with one of two patterns. The first pattern is {iter}_{pp}_{am}_{tm}_{em}.FAR/SKM/TTT. These are transit skims, and the naming follows the pattern in Table 30. A second pattern is used for auto skims, {iter}_{mode}{_option}.skm, and the pattern elements are listed in Table 31. The highway skim files include four tables, time in minutes, distance (multiplied by 10), toll in cents, and variable-price toll in cents.

A subfolder of this folder is OMX_Skims and includes a copy of each skim file that has been converted to the Open Matrix format for use in ActivitySim. In addition to the copies, there is a county.omx file, which is a matrix of destination jurisdiction code and is built on-the-fly from the land use input data. Additionally, a file called skims.omx is in this folder and is a consolidated copy of all skim files in this folder for the current iteration.

TABLE 36: TRANSIT SKIM FILE NAME PATTERN ELEMENTS

TOKEN	DATA ELEMENT
{iter}	Iteration Number (i1, i2, i3, i4)
{pp}	Time period (AM, MD, PM, NT)
{am}	Access Mode (WK = Walk, DR = PNR, KR = KNR)
{tm}	Transit Mode (AB = All-bus, MR = Metrorail only, BM = Bus and Metro combined path, CR = Commuter Rail)

{em}	Egress Mode (WK = Walk, DR = PNR, KR = KNR)
File Extension	FAR = Fare Skim SKM = Skim TTT = Sum of transit time

TABLE 37: HIGHWAY SKIM FILE NAME PATTERN ELEMENTS

TOKEN	DATA ELEMENT
{iter}	Iteration Number (i1, i2, i3, i4)
{mode}	Mode (sov, hov2, hov3)
{_option}	If there is no _option, it is a skim with time in minutes, distance in miles, and tolls in dollars. If the option is _MC, it is a skim for mode choice…

The transit skim files include several tables, these tables are listed in Table 32.

TABLE 38: TRANSIT SKIM MATRIX TABLES

NAME	TYPE	DESCRIPTION
IVTLB	Decimal	Local bus in-vehicle time (minutes)
IVTXB	Decimal	Express bus in-vehicle time (minutes)
IVTMR	Decimal	Metrorail in-vehicle time (minutes)
IVTCR	Decimal	Commuter Rail in-vehicle time (minutes)
IVTLR	Decimal	Light rail in-vehicle time (minutes)
IVTBR	Decimal	Bus Rapid Transit in-vehicle time (minutes)
IWAIT	Decimal	Intial wait time (minutes)
XWAIT	Decimal	Transfer wait time (minutes)

WALKT	Decimal	Walk time (minutes)
DRVACCT	Decimal	Drive access time (minutes)
FARE	Decimal	Fare (dollars)
DRVACCD	Decimal	Drive access distance (miles)
PNRCOST	Decimal	PNR cost (not used)
BRDLB	Decimal	Local bus boardings (not used in skims)
BRDXB	Decimal	Express bus boardings (not used in skims)
BRDMR	Decimal	Metrorail boardings (not used in skims)
BRDCR	Decimal	Commuter Rail boardings (not used in skims)
BRDLR	Decimal	Light Rail boardings (not used in skims)
BRDBR	Decimal	Bus Rapid Transit boardings (not used in skims)
TOTTT	Decimal	Total travel time (minutes)
WALKACCT	Decimal	Walk access time (minutes)
WALKOTHT	Decimal	Walk transfer time (minutes)
XFERS	Decimal	Number of transfers
XFERPEN	Decimal	Transfer penalty (minutes)
PNRT	Decimal	PNR time (minutes)
PNRC	Decimal	PNR cost (cents)
TOTIVT	Decimal	Total in-vehicle time (minutes)
PROB	Decimal	Probability of using bus+Metrorail routes

Transit Assignment

The transit assignment outputs are in the trn_assign folder. These files follow the same nomenclature as the skims files shown in Table 36. For each iteration, time period, access mode, transit mode, and egress mode, there are four files:

- LINKVOL.DBF: a link volume file. See Table 39.
- S2SVOL.DBF: a station-to-station volume file. See Table 40.
- PRN file: Cube transit assignment report file.
- TRP file: the Trip table. Includes one matrix, which represents transit person-trips for that group.

TABLE 39: LINKVOL OUTPUT TABLE FIELDS

NAME	TYPE	DESCRIPTION
A	Integer	A node of link
B	Integer	B node of link
MODE	Integer	Mode number
OPERATOR	Integer	Operator number
NAME	String	Line name (ID value)
LONGNAME	String	Line name (longer)
DIST	Decimal	Distance of link
TIME	Decimal	Transit travel time of link
LINKSEQ	Integer	Number of link in route
HEADWAY_1	Integer	Route headway
STOPA	Integer	If the A node is a stop (1) or non-stop (0)
STOPB	Integer	If the B node is a stop (1) or non-stop (0)
VOL	Integer	Volume for link, mode, operator, name
ONA	Integer	Boardings in current direction at A node
OFFA	Integer	Alightings in current direction at A node

ONB	Integer	Boardings in current direction at B node
OFFB	Integer	Alightings in current direction at B node
REV_VOL	Integer	Volume for reverse link (same mode, operator, name)
REV_ONA	Integer	Boardings in reverse direction at A node
REV_OFFA	Integer	Alightings in reverse direction at A node
REV_ONB	Integer	Boardings in reverse direction at B node
REV_OFFB	Integer	Alightings in reverse direction at B node

TABLE 40: STATION-TO-STATION OUTPUT TABLE FIELDS

NAME	TYPE	DESCRIPTION
I	Integer	Trip origin zone id
J	Integer	Trip destination zone id
FromNode	Integer	Trip origin station number
ToNode	Integer	Trip destination station number
Mode	Integer	Trip mode
Accum	Integer	Movement type code (all 5, indicating adjacent-by-mode analysis)
VOL	Decimal	Volume from station FromNode to station ToNode

Transit Network

The transit network outputs are in the trn_net folder. There are over 2,000 files in this folder following a few patterns:

- LEG and NT files: these are network link files (text-based) for access and egress legs. The file names include the access mode, transit mode, and time period (if applicable)
- LIN files: these are transit line files in Cube format (text-based)

- {iter}_{acc}_{trn}_{egg}_LINK.DBF and LINKVOL.DBF (using the pattern listed in Table 36): these files are identical and use the fields listed in Table 39 except the boarding, alighting, and volume fields
- NET: This file is the transit network for the mode combination. This can be read in Cube to show routes, connections (access, egress, and transfer links), etc.
- RTE: This is the Cube route file. This file can be read in Cube to trace transit paths.
- Skim.PRN: This is a skim report file

4.0 RUNNING THE MODEL

4.1 WHAT TO PREPARE/CHANGE

The initial model inputs are for the 2018 base year model. Inputs need to reflect the conditions that the model is expected to test. This can include inputs, such as the highway network, transit network, land use, and/or other files that tell the model about the transportation and land use conditions that the model will be simulating. Other files that can be tested include model files to reflect policy decisions that affect how, when, where, or if people travel. For example, some model files can increase the number of persons working from home that would cause fewer people to make work trips.

For long-range plans, air quality analysis, and traffic impact studies, changes are generally limited to the input highway network, input transit network, land use input files, auxiliary model input files, and external and visitor transit matrices. These files should be updated to reflect the horizon year of the study. In the case of long-range plans and air quality analyses, any changes to model parameter files should be discussed and approved by FHWA and the interagency coordinating group.

The travel model can be used for many additional studies, such as:

- Autonomous Vehicle (AV) Ownership
- Telecommute behavior changes
- Work from home changes
- Toll analysis

4.2 WHAT SCRIPTS TO RUN

The primary model script, run_model.bat, runs the entire model. Model users should review section 2.0 prior to running this script.

4.3 VISUALIZER PREPARATION

The Visualizer is prepared to compare the model run to the household survey data. This situation is ideal for model calibration and validation purposes; however, it may not be ideal for scenario analysis.

After the transit assignment step, the Visualizer script summarizes the current model run to the folder {scenario}\inputs\Visualizer\data\summarized_abm. When the Visualizer runs, it compares that data to the data in the path defined in the run_model batch file for the key BASE_SUMMARY_DIR.

To setup the Visualizer to compare against a previous model run, copy the summarized_abm folder in the current model run folder TO A DIFFERENT FOLDER NAME, such as summarized_base_model. Then, set the run_model varibles as listed in Table 3029.

TABLE 30: VISUALIZER VARIABLES IN RUN_MODEL.BAT

VARIABLE NAME	DESCRIPTION	EXAMPLE
BASE_SUMMARY_DIR	Folder of data to compare as the base scenario.	%SCEN_DIRECTORY%inputs\Visualizer\data\summarized_base_model
BASE_SCENARIO_NAME	The name of the base scenario to use in the Visualizer.	Base Model
IS_BASE_SURVEY	Tells the Visualizer script to look for certain files – must be ‘No’ when comparing to a prior model run.	No
BASE_SAMPLE_RATE	The sample rate of the base data. For comparing to a prior model run that has completed all iterations, use 1.0.	1.0
BUILD_SCENARIO_NAME	The name of the build scenario to use in the Visualizer.	Test Scenario

Note that the build scenario will be set automatically to use the summarized_abm folder and will overwrite any files in that folder.

4.4 RUNNING THE MODEL ON AN ON-PREMISES SERVER VERSUS IN THE CLOUD

As discussed in Section 2.4, there is an automatic shutdown option for use with cloud servers where a cost might be accumulated for a server that is running. However, users who use the automatic shutdown option should be mindful of others who may be using the same server.

4.5 EXPECTED MODEL WARNINGS

Several messages are output to the screen that are expected. Some of these are DOS error messages that are frequently seen when running a subsequent run on the same file structure. Others are Python messages and warnings that look ominous but are known and will be fixed at some point in the future.

DOS Messages

`A subdirectory or file already exists`: this message is shown on the screen when the model script attempts to make a folder that already exists. This happens during skimming in the second, third, and fourth iterations where the script attempts to create the omx_skims subfolder that is created in the first iteration.

`SQLalchemy is not installed. No support for SQL output.`: This message means that a library called SQLachemy is not installed, which is not used in any of the model processes. It can be ignored.

ActivitySim Messages

``PerformanceWarning: DataFrame is highly fragmented. This is usually the result of calling `frame.insert` many times, which has poor performance.  Consider joining all columns at once using pd.concat(axis=1) instead. To get a de-fragmented frame, use `newframe = frame.copy()` ``: this is a warning related to some of the ActivitySim models and can be ignored. It is not an error.

mp_households_9 WARNING - activitysim.abm.models.trip_scheduling - trip_scheduling.i100.outbound.num_2 coercing 75 depart choices to most initial: This is a warning that some trips failed the scheduling model, it is normal and should be a small number of trips.

4.6 DEBUGGING MODEL CRASHES

This model has several moving parts, and the first step to debugging a crash is to determine if there is an easy reason - such as insufficient disk space. This model requires a lot of space.

The next step is to determine where the model crashed. At times, the model may continue running after an initial crash and then finally stop after a subsequent crash. One of the best places to check for this is {scenario}\outputs\reports\{scenario}_fulloutput.txt, which is a log of everything written to the screen. From this, a user should be able to determine:

- Did the model crash in the initial iteration or a feedback loop?
- Did the model run ActivitySim correctly? Did the model start ActivitySim?
- Is there an error message that was written to the screen that tells what crashed?

Pre-ActivitySim Crashes

In the case that the model created Cube Skims (in {scenario}\output\skims) but did not create OMX skims, a likely cause is the package manager paths not being configured correctly. This problem will show an error similar to the following text.

```
(path)\__init__.py:138: UserWarning: mkl-service package failed to import, therefore Intel(R) MKL initialization ensuring its correct out-of-the box operation under condition when Gnu OpenMP had already been loaded by Python process is not assured. Please install mkl-service package, see http://github.com/IntelPython/mkl-service
  from . import _distributor_init
Traceback (most recent call last):
  File "(path)\(filename).py", line 1, in <module>
    import pandas as pd, numpy as np
  File "(path)\__init__.py", line <n>, in <module>
    raise ImportError(
ImportError: Unable to import required dependencies:
numpy:


IMPORTANT: PLEASE READ THIS FOR ADVICE ON HOW TO SOLVE THIS ISSUE!


Importing the numpy C-extensions failed. This error can happen for
many reasons, often due to issues with your setup or how NumPy was
installed.
```

The above block of messages comes from the runABM script and it indicates that Anaconda or Mamba was not activated. The way to fix this:

1. If seen during a model run, use control+c to interrupt the model (the model will crash eventually, but may attempt to run for a while before crashing)
2. Verify that USE_MAMBA or USE_ANACONDA is set correctly and that the python and package manager paths are set correctly- see Section 2.3: Step 2: Install and Configure ActivitySim. *Even if the Python path looks correct, double check that it is correct! This is the most likely cause of this error!*

ActivitySim Crashes

A way to determine what happened in ActivitySim is to check the output log. If ActivitySim crashed, the output log ({scenario}\outputs\activitysim\log\activitysim.log) will either show the error or show that an error occurred in a process log (mp_households_n-activitysim.log, where n is the process number). If activitysim.log has no errors, the last line will likely be:

```
{date} {time} INFO - activitysim.core.tracing - Time to execute all models {seconds} ({minutes})
```

In this case, ActivitySim ran correctly.

Common ActivitySim Crashes

In addition to the items listed in the previous section, the following crashes are the most common in our experience.

No Space Left on Device

As the error implies, there is no space left on the drive where the output folder is stored. Additionally, sometimes the HDF5 library will throw an exception that looks similar to the output below:

```
  File "tables\hdf5extension.pyx", line 1297, in tables.hdf5extension.Array._
create_array
tables.exceptions.HDF5ExtError: Problems creating the Array.
```

Probabilities do not add up to 1.0

This error happens when one or more choosers have no available alternative. Generally, this happens when there are restrictions on all alternatives for that chooser. The best way to handle this issue is to trace one of the households for that model to investigate if there is a data problem or a model problem.

This error should not be seen in the production version of the Gen3 Model. However, it can be seen during estimation data bundle preparation for various reasons if data shows an impossible combination. One example is if a person is marked as works-from-home and has a work location TAZ included in the person file used to create the estimation data bundles.

Expected {x} fields, found {y} fields

This happens when a configuration file is formatted incorrectly. Frequently the specific line where the error occurs is printed on the screen. If there are fewer fields found than expected (y < x), locate the line in the configuration file and check to see if there is a missing field. If there are more fields than expected (y > x), locate the line and ensure there isn't an additional field and also ensure that if an expression includes a comma that it is in quotes.

KeyError: {data field}

This means that the model specification or chooser annotation is referencing a field that it cannot find. One of the causes of that is if the model has a specific list of chooser fields. This is found in the model's settings file (yaml file) under the key `SIMULATE_CHOOSER_COLUMNS`, and there is a list of fields that are available to the model. Another cause is if the field does not exist for the chooser. For example, the household variable `auto_ownership` is not available to other models without being added in an annotation file.

Could not find {output_dir, data_dir, configs_dir} {path}

This message indicates that ActivitySim cannot find the folder type (output_dir, data_dir, or configs_dir) at the path specified. Check all input and output folder paths and the configuration of ActivitySim and/or the model.

Cube Crashes

There are two places to check Cube outputs to determine why Cube crashed. One of them is the report files in {scenario}\outputs\reports - this can be sorted by date/time and checked in the order of file creation time (newest to oldest). Additionally, some steps may not finish and may leave files in {scenario}. These will generally look like voyaNNN.prn, where NNN is a sequential number. Note that "VoyagerCrashDump_NNNNN.log" files rarely contain usable information and are quite common to see even in model scripts that successfully run.

Cube print log files can be difficult to understand since Cube frequently will attempt to run an entire step, even if an error happens. For this reason, it's best to search for all lines that begin with "F(" - these will be followed by a number that indicates the error. It is generally best to fix the first error and re-run the step since errors will frequently cascade (e.g. a missing file will cause an error saying that it can't find the file, and additional errors will be displayed when parts of that file - matrices, field names, etc. - are referenced).

In the case of transit assignment errors indicating that no path can be found - this error should not happen but is likely if ActivitySim is set to cache the skim files. Ensure that read_skim_cache and write_skim_cache in source\configs\activitysim\configs\network_los.yaml are False. Those should never be set to True in a production model. This ensures that ActivitySim is using the correct skim files and not old data ("old" can mean prior-iteration data in this case).

4.7 MODEL STEPS AND SPEED FEEDBACK LOOPS

The model runs in three phases - the first phase is an initial phase, the second phase is the feedback loop, and the third phase is the final transit assignment and post-processing. These steps are shown in Figure 5.

The initial phase includes four steps. The first step sets pricing factors using Consumer Price Index data. The second step builds the highway network from the input data. The third step prepares initial highway skim data to be used in ActivitySim. The final step prepares transit fare data, which only needs to be done once.

The feedback loop is run four times and feeds back travel times based on ActivitySim output.

The feedback loop includes six steps. The first is transit path building and skimming, which prepares the transit skim data for use in ActivitySim. The second step is ActivitySim, which models the movements of internal passenger trips. The third step is auxiliary trips, which prepare truck, commercial vehicle, and trips that are not handled by ActivitySim, including airport passengers, visitors, through traffic, external-internal, and internal-external trips. The fourth step is highway assignment, which assigns all the auto trips from ActivitySim and the auxiliary models. The fifth step is a step to update link speeds, which is only run for iterations 2, 3, and 4.

The final step includes transit assignment, which only needs to be run at the end of the model, and two post-processing steps to report transit ridership and prepare a dashboard of the ActivitySim output.

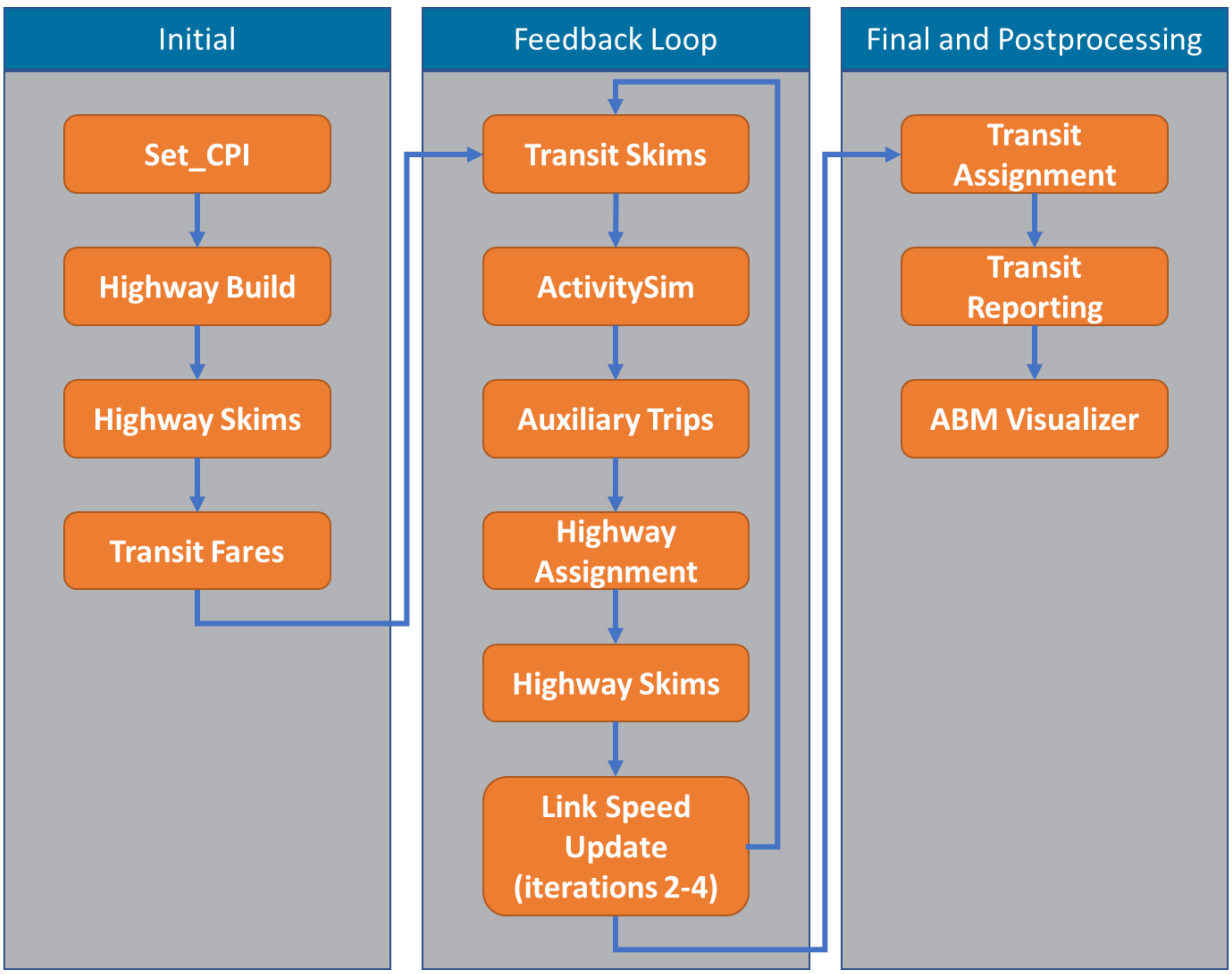


FIGURE 5: MODEL FLOW SUMMARY

5.0 WORKING WITH ACTIVITYSIM

ActivitySim is a complex modeling system that uses open-format files and attempts to standardize as much of the formatting as possible.

5.1 HOW ACTIVITYSIM RUNS

ActivitySim starts by reading the input files listed in the settings file - which are the land use file, synthetic population and household files, and skim matrices. Once the files are read into memory, ActivitySim starts into the models listed in the settings file.

The first three model steps are special. The `initialize_landuse` (first) runs through annotations on the land use. The annotation files are described later in this section, but generally they add variables to the data. The second step is `compute_accessibility`. This model computes zonal accessibility data that is used throughout the model. The third step is `initialize_households`, which runs through annotations on the households and persons files.

The next several steps (everything between initialize_households and write_data_dictionary) are specific model steps. Each of these steps has a model settings file, a model specification, and a coefficients file. Some of these steps have an annotation file, and mode choice models include a coefficient template file. These files specify how each model step will run. Annotation files are described in section 5.2. The model settings, specification, coefficient, and coefficient template files are all discussed in Section 5.2.

The final steps of the model direct ActivitySim to write the outputs. These include the final four steps of the process, `write_data_dictionary`, `track_skim_usage`, `write_tables`, and `write_trip_matrices`. The most important of these two steps are `write_tables` and `write_trip_matrices` - write tables directs ActivitySim to write the tables listed in the settings file (under `output_tables`) and `write_trip_matrices` directs ActivitySim to write out OMX trip matrix files that can be input into Cube.

5.2 ACTIVITYSIM CONFIGURATION AND YAML FILES

Most ActivitySim model steps use three files - a settings file, a specification file, and a coefficient file. Mode choice models use a fourth file, a coefficient template file, that bridges the alternatives in the specification file with the coefficients by purpose.

Model Settings Files

The specification files - named as the step and with a yaml extension - tell ActivitySim what files to use for a model and other pertinent information about the model. These are mostly simple but can become complex at times.

There are two keys that are required for each model, SPEC and COEFFICIENTS. SPEC is the file name of the specification file, and COEFFICIENTS is the file name of the coefficients list. These two files are discussed later in this section.

Sometimes these files will include a line that says annotate_choosers. This directs ActivitySim to run an annotation file on the choosers BEFORE running the model simulation. This allows additional variables to be included for that model. This is common with mode choice models where things like the availability of a mode may be defined in an annotation file to make the model spec easier to read.

Sometimes these files will include a line that says annotate_<table>, where table is persons, households, or another table in ActivtySim. This directs ActivitySim to run an annotation AFTER the model simulation. This is used to add results from a model step to the internal and output ActivitySim tables.

Sometimes these files will include CONSTANTS, which defines constants used in that model step.

The mode choice versions of these files also define the nest structure of the model since mode choice in ActivitySim is usually a nested logit model.

Model Specification Files

The model specification files determine how the model calculates the utilities for each alternative. These files are CSV files with fields for description, expression, and fields for each alternative. The expression defines the data, and the fields for each alternative define the coefficient, which can be a coefficient name or a number. A best practice is to only code a number when an expression is used to limit the availability of an alternative, and for those -999 should be used as the coefficient value. For all other applicable coefficients, the name should be used. The coefficient name needs to be identical to that used in either the coefficients file or the coefficients template file (mode choice models only).

Model Coefficient Files

The model coefficient files simply list the coefficient name, value, and if the value should be constrained by estimation software. The coefficient names are case-sensitive and should not have spaces. The values should only be numbers, and the constrain field should only be either F (for false) or T (for true). The constrain field does not affect model results. It is only used in model estimation.

Model Coefficients Template File

In the case of mode choice models, the coefficient template file links alternative coefficients to purposes. The file is a CSV file and has a field for the coefficient name and fields for each purpose in the model. The coefficient name field should match the coefficient names used in the model specification file. The fields for each purpose should match the coefficients file. When the mode choice model is simulated, ActivitySim will look at the specification file for the coefficient, then look for that coefficient in the template file. Then it will select the coefficient name for the tour/trip purpose it is simulating and select the corresponding coefficient value from the coefficients list.

Annotation Files

The annotation files are files that add a variable to a table. In some cases, this is to format data that ActivitySim is expecting, such as the ptype annotations in annotate_persons.csv. In some cases, it involves getting data from another table.

Each annotation file is a CSV file with three fields:

Description: This is a text description of the data. It is not used by ActivitySim.

Target: This is the variable name. This is case-sensitive and cannot have spaces.

Expression: This is a Python expression that defines the data in the target.

Expression Examples

Example 1: Get number of retired adults in household

This example uses the _PERSON_COUNT function (see Example 3 for more information) to count the number of persons in the household that are ptype 5, which is retired adults. This is placed in the variable _num_retired_adults. This variable is not written to the output file - beginning a variable name with an underscore tells ActivitySim that it should not be written to the output file. Note that the expression is in quotes because it includes commas in the expression - the quotes will prevent ActivitySim from parsing the expression as several fields (and ultimately throw an error).

Target: `_num_retired_adults`

Expression: `"_PERSON_COUNT('ptype == 5', persons, households)"`

Example 2: Determine if a household is retired adults only

This example looks at the household size (household.hhsize) and compares it to the _num_retired_adults variable to determine if the household has only retired adults. This variable will be written to the output file.

Target: `retired_adults_only_hh`

Expression: `(households.hhsize > 0) & (households.hhsize == _num_retired_adults)`

Example 3: Person Count Query

This is a query that is used in other annotation lines. The target begins with an underscore, which tells ActivitySim to not include this file in the output. The expression uses a lambda, which is essentially a function that uses query, persons, and households as input arguments and then queries the persons table based on the query argument, groups by household_id (which is a reserved term in ActivitySim), gets the size, reindexes to the order of the households index (note that this is a Pandas Series operation[18], NOT the ActivitySim operation), fills NA values with 0, and then returns the values as an integer.

Target: `_PERSON_COUNT`

Expression: `"lambda qry, persons, households: persons.query(qry).groupby('household_id').size().reindex(households.index).fillna(0).astype(np.int8)"`

Example 4: Using Reindex to Assign a Variable from Another Table

This example gets the jurisdiction code (JURCODE) from the land use file and sets it as the home_jurisdiction variable based on the household's home_zone_id. Reindex is an ActivitySim function that takes the destination variable as the first argument and the destination index as the second argument. The destination index needs to be the same data value-type as the table being reindexed - in this case, land use data is indexed on the TAZ ID, and households.home_zone_id is the TAZ ID of the home, so reindex returns the JURCODE for each home.

Target: `home_jurisdiction`

Expression: `"reindex(land_use.JURCODE, households.home_zone_id)"`

5.3 WORKING WITH ACTIVITYSIM MODEL OUTPUTS

This section provides some additional guidance about using the ActivitySim output files. ActivitySim outputs matrix files, which are summaries of trips meant for import into software to assign these matrices (such as Cube Voyager). ActivitySim also outputs several tabular files in comma-separated values (CSV) format, which are described in detail in section 3.2.

Working with household, person, tour, and trip files

Tabular file data dictionaries are listed in section 3.2. In all files, there is an id field that can be used to join to other files. The id values (household_id, person_id, tour_id, trip_id) are unique such that only one needs to be used to join (e.g. if joining a person value to the tour table, only the person_id needs to be used to join. All tables include as many id values as possible, so in the trip table, the household_id, person_id, and tour_id fields are included.

The tabular file data from ActivitySim is CSV files, which can easily be input into R, Python, and many other software packages. Excel is generally not recommended, however, since the files

[18] https://pandas.pydata.org/docs/reference/api/pandas.Series.reindex.html

tend to be larger than Excel allows. Excel allows 1,048,576 rows and a full ActivitySim run has several output tables that drastically exceed this limit. Python code to read the files into a Pandas dataframe and create some crosstabs is included in section -1873382512.645.

5.4 MODEL CALIBRATION AND VALIDATION

This section lists advanced techniques to use with the Gen3 model to assist with calibrating and validating the model. These techniques are different than what is used in model estimation[19] because estimation was conducted using ActivitySim only, whereas calibration and validation were conducted in the main model process to allow for highway and transit assignments and potential changes to the skims. Additionally, these techniques will allow for more rapid testing and adjusting ActivitySim.

The procedures in this chapter are advanced. These procedures involve editing the DOS batch files, editing ActivitySim files (many of which are text-based), and running scripts to use the outputs and adjust the model configuration files.

Calibration Preparation

Prior to starting recalibration, comment out the temp file moving script. This is near the end of run_modelsteps.bat, and should look like this once commented out:

```
:: Move unimportant files to a temp folder
::call %BATCH_DIR%\move_temp_files_gen3.bat %1
```

The reason for this is because that script will move the ActivitySim pipeline files that are used to re-start ActivitySim from partway through the run.

Next, run the entire model. This ensures that the skims used in ActivitySim are the final skims after the feedback iterations.

Alternatively, if a full model run is available and the temp files (in {scenario}\temp_files) have not been deleted, simply moving all of the pipeline files (pipeline.h5 and mp_households_*n*-pipeline.h5, where n is 0 - number of processors) back to the ActivitySim output folder ({scenario}\outputs\activitysim) would be fine.

The next item to prepare is to shut off steps that do not need to be run. This is best completed by editing the run_modelsteps.bat script. This script is broken up into sections - the first section from the beginning of the file to `ECHO ====== Pump Prime Iteration` sets several environment variables needed for the script (as does run_model.bat, which calls this script). The next sections are for the iterations, and all have the same set of calls to scripts - transit skims, ActivitySim, auxiliary trips, highway assignment, and highway skimming. For calibration and validation of ActivitySim only (and when using an existing model run), all steps can be commented-out except the Iteration 4 RunABM script. Do NOT comment out any `set`

[19] RSG and Metropolitan Washington Council of Governments. “GEN3 Model Phase 2 Model Estimation”, Technical Report Prepared for Metropolitan Washington Council of Governments, March 2, 2023.

statements, as those are necessary for the model to run. For those that understand DOS batch scripting, a GOTO statement can be added before the Pump Prime Iteration marker sending execution to immediately following the Iteration 4 marker, and then commenting the call to transit skims and adding GOTO MSEND after the runABM call.

The next script to edit is RunABM.bat (in source\scripts\batch). For calibration, it may not be necessary to run all of the households, so the sample for iteration 4 can be changed from 0 (run all) to another number (this needs to be the number of households, unfortunately ActivitySim does not yet support percentages). In the case that skims are not changing, the scripts to convert Voyager scripts to OMX (look for `start /w Voyager.exe ..\source\scripts\cube\SKM_to_OMX.s /start -Pvoya -S..\%1`) and the script to fix the index in the OMX files (look for `%PYTHON% ..\source\scripts\python\cube_to_omx.py %_iter_% %_prev_% %SCEN_DIRECTORY%`) and the script to consolidate the skims OMX file (look for `%PYTHON% ..\source\scripts\python\build_omx.py %_iter_% %_prev_% %SCEN_DIRECTORY%`) can be commented out (using either `REM` or `::` at the beginning of the line). Additionally, the second skim builder (look for `%PYTHON% ..\source\scripts\python\make_county_omx.py`) can also be commented out since the files created by this process will have already been created.

In the case of repeated ActivitySim runs, the skim cache can be turned on. This is done by editing the network_los.yaml file in source\configs\activitysim\configs. The options `read_skim_cache` and `write_skim_cache` can be set to True (case sensitive!) to direct ActivitySim to cache the skims file in the output cache folder. Be advised that this option will result in a 75-100 GB cache file written in the output cache folder. Once calibration is complete, this option needs to be set to False and the skim cache file can be deleted.

To reduce ActivitySim runtime, ActivitySim can be set to start at a model by opening the settings file in source\configs\activitysim\configs_mp\settings_source.yaml and uncommenting and setting `resume_after:` to the step before the step in calibration. To avoid running several additional models, comment out (using a hash symbol, #) the steps following the step to calibrate to the `### mp_summarize step` comment. This will start ActivitySim at the step to calibrate and run the uncommented steps, and then run the outputs (writing output tables and matrices). Do not comment out anything past the `### mp_summarize step` comment, or it can cause configuration errors (the `### mp_households step` and `### mp_summarize step` mark where multiprocessing starts and stops). Note that the model output tables (final_households.csv, final_persons.csv, etc.) are only written if the appropriate steps are run - for example, the final_trips.csv file is not written unless the stop frequency model is run. Additionally, the output tables will not include data and fields for steps that were not run - for example, if the at-work subtour models are not run, the final_tours.csv file will not include the parent_tour_id field and will not include any at-work tour data at all. For most situations, running just the step to calibrate is fine. For tour mode choice, the group of models of tour_mode_choice_simulate, atwork_subtour_frequency, atwork_subtour_destination, atwork_subtour_scheduling, and atwork_subtour_mode_choice need to be run.

If any annotation files before the step being run are changed (e.g. annontate_households, annotate_persons, annotate_land_use), the model will need to be run from the start to reflect those changes. If the annotate choosers file for the step being run is changed, then just that step needs to be run.

At this point, the model should run and output updated results for only the steps selected. The outputs can be read into another software package to estimate alternative specific constants.

Model Calibration

Model calibration refers to the adjustment of models to match observed data. If possible, this observed data should be different data than the data used to estimate models (if applicable). For example, if the work-from-home model was estimated using household survey data, the calibration data should be Census American Community Survey data.

An example model calibration script is supplied in Section 7.3. Additionally, the ActivitySim Visualizer displays many comparisons of the current model run to the survey by default (note that this can be changed by following the steps in Section 2.3 Step 2).

In-depth concepts surrounding model calibration/validation are included in the Federal Highway Administration Transportation Model Improvement Program report *The Model Validation and Reasonableness Checking Manual*[20].

Model Validation

Validation of the model refers the comparison of model output results against observed data. Prior to model validation, data must be gathered and formatted in a way that it can be compared with model outputs, and model outputs would need to be formatted in a way that can be compared against observed data.

Validation of ActivitySim outputs is best completed through the ActivitySim Visualizer. This visualizer is already set up to aggregate data as appropriate and handles much of the joining and linking needed to show charts.

Two major items of validation are not addressed by the visualizer: highway and transit assignment validation. RSG has prepared Jupyter Notebooks to prepare data for these two items.

Highway Validation

The highway validation Jupyter Notebook reads the output highway network (Gen3_Model\2018_base\outputs\hwy_net\i4_Assign_Output.net), converts it to a DBF, and then processes the data to populate the highway validation spreadsheet. This notebook then compares highway counts and model assignment on a segment (with counts) basis and

[20] Cambridge Systematics. Travel Model Validation and Reasonableness Checking Manual, Second Edition, prepared for Travel Model Improvement Program, Federal Highway Administration, Washington, D.C.
https://www.fhwa.dot.gov/planning/tmip/publications/other_reports/validation_and_reasonableness_2010/fhwahep10042.pdf.

aggregated to area type, facility type, jurisdiction, and screenline. An example table from this spreadsheet is presented in Figure 6.

Table 4: Highway Validation - Estimated VMT vs. Observed VMT based on Links with Traffic Count - By Area and Facility T

Model Run: **Gen3 Model (08-19-2023)**

Estimate_2018

	Facility Type						
Area Type	Freeway	Major Arterial	Minor Arterial	Collector	Expressway	Ramp	TOTAL
1	495,732	798,977	349,888	105,608	430,032	0	2,180,237
2	4,980,233	3,175,255	1,645,788	265,385	708,954	0	10,775,615
3	8,557,753	2,694,711	2,018,915	547,666	1,606,706	0	15,425,750
4	5,012,742	1,995,201	1,803,523	382,034	738,183	0	9,931,683
5	9,408,154	3,654,892	2,191,623	523,169	1,361,223	27,529	17,166,591
6	6,706,914	5,040,081	5,884,461	524,826	660,262	0	18,816,544
TOTAL	35,161,528	17,359,117	13,894,198	2,348,688	5,505,360	27,529	74,296,420

Observed_2018

	Facility Type						
Area Type	Freeway	Major Arterial	Minor Arterial	Collector	Expressway	Ramp	TOTAL
1	554,106	700,612	363,478	93,258	412,572	0	2,124,026
2	5,466,431	3,419,008	1,769,923	362,173	888,596	0	11,906,131
3	8,878,084	2,964,965	2,343,673	785,763	1,927,039	0	16,899,524
4	4,872,291	2,042,296	2,122,981	535,084	787,959	0	10,360,611
5	9,472,552	3,391,423	2,144,360	719,425	1,297,292	36,284	17,061,336
6	6,119,478	4,381,387	4,234,617	657,833	667,139	0	16,060,454
TOTAL	35,362,942	16,899,691	12,979,032	3,153,536	5,980,597	36,284	74,412,082

Estimate/Observed Ratio

	Facility Type						
Area Type	Freeway	Major Arterial	Minor Arterial	Collector	Expressway	Ramp	TOTAL
1	0.89	1.14	0.96	1.13	1.04	-	1.03
2	0.91	0.93	0.93	0.73	0.80	-	0.91
3	0.96	0.91	0.86	0.70	0.83	-	0.91
4	1.03	0.98	0.85	0.71	0.94	-	0.96
5	0.99	1.08	1.02	0.73	1.05	0.76	1.01
6	1.10	1.15	1.39	0.80	0.99	-	1.17
TOTAL	0.99	1.03	1.07	0.74	0.92	0.76	1.00

FIGURE 6: EXAMPLE HIGHWAY VALIDATION SPREADSHEET OUTPUT

Transit Validation

The transit validation Jupyter Notebook reads all of the transit loading line and station-to-station output files and summarizing them to match ridership data. The spreadsheet shows an overview of transit loadings by mode, Metrorail loadings by station group, Metrorail station from-to comparison, and commuter rail loadings by station. An example of this spreadsheet is included as Figure 7.

	2018 Daily Boardings and Alightings Estimates (Gen3 Model)														
Mode Name	All Bus: Sum of ONA	All Bus: Sum of OFFB	Bus & Metrorail: Sum of ONA	Bus & Metrorail: Sum of OFFB	Metrorail: Sum of ONA	Metrorail: Sum of OFFB	Metrorail: Boardings without Transfers	Commuter Rail: Sum of ONA	Commuter Rail: Sum of OFFB	**Combined: Sum of ONA**	Combined: Sum of OFFB	**2018 Observed Station ENTRIES or BOARDINGS**	2018 Ratio E/O for Boardings vs. Entries	**Notes**	**Gen 2.4 2014 E/O**
Local Metrobus	272,402	272,403	61,227	61,226	0	0		4,526	4,526	338,155	338,156		n/a		
Express Metrobus	20,864	20,863	13,623	13,623	0	0		266	266	34,753	34,753		n/a		
Metrorail	0	0	169,952	169,951	678,011	678,011	659,167	28,563	28,563	876,525	876,526	641,227	1.03	see Note 1	1.01
Commuter Rail	0	0	0	0	0	0		53,284	53,284	53,284	53,284	56,580	0.94	see Note 2	0.76
Other Local Bus in the WMATA Area	127,226	127,226	43,660	43,660	0	0		2,454	2,454	173,339	173,339		n/a		
Other Express Bus in the WMATA Area	2,414	2,414	2,556	2,556	0	0		183	183	5,152	5,153		n/a		
Other Local Bus beyond the WMATA Area	19,856	19,856	2,983	2,983	0	0		1,216	1,216	24,055	24,055		n/a		
Other Express Bus beyond the WMATA Area	26,475	26,475	14,256	14,257	0	0		1,961	1,961	42,693	42,693		n/a		
Bus Rapid Transit and Street Car	123	123	3,155	3,155	0	0		390	390	3,668	3,668		n/a		
All Bus	469,236	469,238	138,304	138,304	0	0		10,607	10,607	618,147	618,149	575,642	1.07	see Note 3	1.09
Metrobus Total	293,266	293,267	74,849	74,849	0	0	0	4,793	4,793	372,908	372,908	360,000	1.04	see Note 4	
Other Bus in WMATA Area	129,639	129,639	46,215	46,216	0	0	0	2,637	2,637	178,492	178,492	141,390	1.26	see Note 5	
Other Bus not in WMATA Area	46,331	46,332	17,239	17,239	0	0	0	3,177	3,177	66,748	66,749	74,252	0.90	see Note 6	

FIGURE 7: EXAMPLE TRANSIT VALIDATION SPREADSHEET TABLE

6.0 MORE DETAILED DESCRIPTION OF MODEL SYSTEM

6.1 HIGHWAY SKIMMING AND ASSIGNMENT SCRIPTS AND INPUTS

Overview

Highway skimming begins with path building, the process of building minimum-impedance paths from every TAZ to every other TAZ. After paths have been built, the paths can be "skimmed," i.e., the paths are traversed, and key variables are summed over the paths. The variables that are skimmed include travel times, distances, and tolls. The resultant zone-to-zone sums are saved in one or more skim matrices. The input to the skimming process is usually a loaded network with congested travel speeds, generated from a traffic assignment process. Although traffic assignment is conducted for four time-of-day periods, The Gen3 Model is set up to use skims for four time-of-day periods: (AM peak period, midday, PM peak period, and nighttime). Highway skims in the Gen3 Model are generated after each traffic assignment step.

Highway skims are generated by time period and by highway travel mode (SOV, HOV 2-occupant, HOV 3+occupant). In addition, truck skims are generated for the midday period only. Mode-specific paths are very important in the Washington, D.C. region, due to special operating restrictions, particularly during the AM peak period.

The TPB's highway skimming develops zone-to-zone (3722 x 3722) skim matrices for the entire model region for use in ActivitySim and the auxiliary models. The entire highway skimming process is applied with the scripts named *Highway_Skims_4TOD.s Remove_PP_Speed.s, and Prepare_Non_Motorized_Skims.s*. These are invoked with the *PP_Highway_Skims.bat* file in the initial or pump-prime iteration (see page A-4 of Appendix A) and the *Highway_Skims.bat* file (see page A-11) in the standard iterations. The *Remove_PP_Speed.s* script is executed in the pump-prime iteration only. The principal inputs and outputs are shown in Table 3130 and Table 3231, respectively.

TABLE 31: HIGHWAY SKIMMING PROGRAM INPUTS

DESCRIPTION	FILE NAME	FILE FORMAT
Network File	<iter>_HWY.NET	Cube Binary Network
Toll Minutes Equivalent	support\toll_minutes.txt	Text
AM toll factors by Vehicle Type	Inputs\AM_Tfac.dbf	DBF
MD toll factors by Vehicle Type	Inputs\MD_Tfac.dbf	DBF

PM toll factors by Vehicle Type	Inputs\PM_Tfac.dbf	DBF
NT toll factors by Vehicle Type	Inputs\NT_Tfac.dbf	DBF

TABLE 32: HIGHWAY SKIMMING OUTPUT FILES

DESCRIPTION	FILE NAME	FILE FORMAT
Total highway skims	<ITER>_SKIMTOT.TXT	Text
Truck skims	<ITER>_MD_TRK.SKM	TP+ Binary Skim
SOV skims	<ITER>_<Prd>_SOV.SKM	TP+ Binary Skim
HOV2 skims	<ITER>_<Prd>_HOV2.SKM	TP+ Binary Skim
HOV3+ skims	<ITER>_<Prd>_HOV3.SKM	TP+ Binary Skim
AM highway skims	<ITER>_HWY_AM.SKM	TP+ Binary Skim
Off peak highway skims	<ITER>_HWY_OP.SKM	TP+ Binary Skim
Network with added station centroid connectors	<ITER>_HWYMOD.NET	Cube Binary Network
Walk access links	WalkAcc_Links.dbf	DBF
Highway network with PP speeds removed	ZoneHWY.NET	Cube Binary Network

Application Details

The highway skimming process is used to develop time, cost, and toll values between origin/destination (i/j) pairs of zones on a minimum-impedance path. The skimming process reads a highway network input file with preexisting restrained speeds. The restrained speeds used in the pump prime (PP) iteration initially are table look-up values based on time period (AM, ,Mid-day or MD, PM, Night Time or NT), facility type, and area type. After the PP iteration is completed (i.e., after the PP traffic assignment process is completed), the highway skimming is accomplished using traffic assignment-based link speeds. The generalized impedance for which paths are developed for highway skimming is defined as follows:

EQUATION 2: ASSIGNMENT GENERALIZED COST

$$Impedance_v = (T)_v + (Toll_v * Tfac_v * Vfac_{vf})$$

Where:

Impedance = Total impedance for vehicle type v
T = Capacity-restrained time for vehicle type v
Toll = Toll (2018 dollars) for vehicle type v
Tfac = Toll factor for vehicle type v
VFac = Vehicle factor for vehicle-type v on facilty f

Note: Vehicle classes are: SOVs, HOV2-occs, HOV3+occs, Commercial Vehicles, Trucks, and airport passenger vehicles.

	AM	MD	PM	NT
SOV	2.1	2.5	2.5	2.5
HOV 2	1.2	3.3	1.7	3.3
HOV 3+	0.8	3.3	0.8	3.3
Commercial Vehicles	1.7	1.7	1.7	1.7
Trucks	1.7	1.7	1.7	1.7
Air passengers	1.7	1.7	1.7	1.7

The assumed time rates are provided by vehicle class and time period in toll_minutes.txt, which is located in the Support folder. The values shown are derived from average household income levels and information from the 2007/08 HTS. The values should not be altered.

The equivalent toll minutes for 2018 were increased from the 2007/2008 values using the US Consumer Price Index inflation factor[21] of 1.21 from May 2007 to May 2018.

The vehicle factors are provided by time period in the input files AM_Tfac.dbf and MD_Tfac.dbf. The file is available to allow for the ability to reflect a facility-specific toll policy differential

[21] US Bureau of Labor Statistics, CPI Inflation Calculator. https://www.bls.gov/data/inflation_calculator.htm. Accessed 5/31/2023.

between vehicle classes. The default assumption that tolls do not vary between vehicle classes, except for trucks, which are assumed to pay 2.5 times the toll that an auto would pay.

Information about the "toll setting" process that is used to estimate reasonable toll values can be found in two technical memos[22].

The Remove_PP_Speed.s script is used to remove the "PP" iteration speed attributes from the highway network. This is necessary in the initial (PP) iteration when table lookup speeds are to be replaced by traffic assignment speeds in the PP iteration.

6.2 TOLL SEARCHING

TPB staff implemented an integrated toll setting procedure that streamlined the intermediate steps and eliminated manual intervention in between these steps. Since its implementation in 2015, this integrated toll setting procedure has been incorporated into COG/TPB's travel demand modeling processes and has by and large remained the same [23].

In application, model users usually use the toll inputs files that are provided by COG/TPB in the model transmittal package. In cases where the toll inputs need to be altered, this integrated toll searching procedure can be carried out once and for all by executing the "wrapper" batch file for the Pump Prime model run.

The "wrapper" batch file automatically calls the "run model steps" batch file which then executes a series of child batch files. At the bottom of the "run model steps" batch file, a child batch file called "Toll_Setting.bat" is added following all the regular travel demand forecasting steps for the Pump Prime model run. The toll-setting batch file automates the toll-searching and post-toll-searching steps into one integrated process. A screenshot of this batch file is shown in Figure 8.

[22] Jinchul Park to Team B Modelers, "Processes Related to Toll Setting in Version 2.3 Model (Draft)," Memorandum, October 12, 2012; Jinchul Park to Files, "HOT Lane Modeling Process of MWCOG/TPB (Draft),"
Memorandum, October 12, 2012.

[23] In 2018, the toll searching algorithm was slightly modified for the latest Version 2.3.75 travel model. The modified algorithm can exit the iterative toll searching process much faster on special occasions where a cap toll rate is imposed. This modification, while being able to reduce the toll searching time on special occasions, didn't affect toll searching results at all.

Toll_Setting.bat

```
CD %1

IF EXIST TOLL_SETTING del TOLL_SETTING /s /q
md TOLL_SETTING
md TOLL_SETTING\AM
md TOLL_SETTING\MD
md TOLL_SETTING\PM
md TOLL_SETTING\FINAL_TOLL

CD TOLL_SETTING

if errorlevel 1 goto error

copy ..\i4_Assign_Output.net AM /y
copy ..\i4_Assign_Output.net MD /y
copy ..\i4_Assign_Output.net PM /y

START ..\..\DP_TS_AM.BAT
START ..\..\DP_TS_MD.BAT
START ..\..\DP_TS_PM.BAT

:wait
@ping -n 11 127.0.0.1>nul
if exist *.flag goto wait

for /f %%a in ('dir AM\OUT*.TXT /B /O:-D') do (set AMTOLL=%%a
goto amstop)
:amstop

for /f %%a in ('dir MD\OUT*.TXT /B /O:-D') do (set MDTOLL=%%a
goto mdstop)
:mdstop

for /f %%a in ('dir PM\OUT*.TXT /B /O:-D') do (set PMTOLL=%%a
goto pmstop)
:pmstop

echo The final toll file are %AMTOLL%, %MDTOLL%, and %PMTOLL%.

CD FINAL_TOLL

start /w Voyager.exe  ../../../scripts/Post_Toll_Search.s /start -Pvoya -S./

goto end
:error
REM  Processing Error....
PAUSE

:end
CD..
```

FIGURE 8: TOLL SETTING BATCH FILE

As shown in Figure 8, Lines 3-8 of the toll-setting batch file create sub-folders under the Pump Prime modeling directory for the subsequent execution of incremental toll search by time of day (“TOLL_SETTING\AM”, “TOLL_SETTING\PM”, “TOLL_SETTING\MD”), as well as post-toll-search (“TOLL_SETTING\FINAL_TOLL”) processes.

Lines 14-24 execute the time-of-day (TOD) toll searching processes. Specifically, Lines 14-16 copy the loaded highway network from the Pump Prime model run as the inputs to the toll searching algorithm; Lines 18-20 launch batch files that run automated toll searching scripts for three different time periods; Lines 22-24 ensure the subsequent steps will not start until the three toll searching processes are complete.

Lines 26-36 find the latest modified out*.txt files from the toll searching outputs, which store the final estimated toll rates for different time periods. Finally, Line 42 executes the post-toll-search script that pulls in the final toll rates and generates the final toll file which will be fed into the final travel demand model run.

Above all, the implementation of the current toll setting procedure involves ten key model files in a travel demand modeling directory. As outlined in Table 3332, nine of these files were either modified or added relative to the previous toll setting procedure. It should be noted that, with the exception of the Toll_Setting.bat file, all the listed files were already created for the previous toll setting procedure and were incorporated into the current procedure with no or very minor changes. The majority of these changes involved altering only the relative paths coded in the files. As mentioned earlier, prior toll searching scripts (TS_V23_AM.s, TS_V23_PM.s, TS_V23_MD.s) contained three parameters for which values needed to be manually specified. In the current procedure, these scripts were modified to either automatically find the value for a specific parameter from the Pump Prime model run, or to eliminate the parameter from the toll searching process.

TABLE 33: KEY MODEL FILES FOR THE TOLL SETTING PROCEDURE

File Name	File Type	Status	Location
run_Model_XXXX_PP.bat*	Batch File	Unmodified	root directory
run_ModelSteps_XXXX_PP.bat	Batch File	Modified	root directory
Toll_Setting.bat	Batch File	Added	root directory
DP_TS_AM.bat	Batch File	Added	root directory
DP_TS_PM.bat	Batch File	Added	root directory
DP_TS_MD.bat	Script File	Added	root directory
TS_V23_AM.s	Script File	Added	\Scripts
TS_V23_PM.s	Script File	Added	\Scripts
TS_V23_MD.s	Script File	Added	\Scripts
Post_Toll_Search.s	Script File	Added	\Scripts

Note*: "XXXX" denotes model year.

The current toll setting procedure was tested based on the base-year model. In this test, it took about 32 hours to execute the entire integrated toll setting procedure in Cube Voyager 6.5 using one of TPB's travel modeling computer servers (tms8).

In the current setup, the user needs to execute a second full Gen3 Model run, which uses the updated toll input file resulting from the above-described Pump Prime model run, to generate the final travel demand forecasts. This "two-run" process is not straightforward and very time-consuming. COG/TPB staff is currently working to consolidate the "two-run" toll searching process into one Gen3 Model run, for which the user will be allowed to turn on or turn off the toll searching process. The default toll input files will be used when the toll searching option is turned off.

6.3 TRANSIT SKIMMING AND ASSIGNMENT SCRIPTS AND INPUTS

Background

While the Gen3, Phase 1, Model adopts the Bentley Citilabs Cube Public Transport (PT) Best-Pathing algorithm, which identifies a single best transit path between an origin zone (O) and a destination zone (D), and loads all O/D demand onto it, COG and RSG decided to switch to the PT Multipathing algorithm in the Gen3, Phase 2, Model, which builds multiple transit paths and finds the respective probabilities of using them. This decision was made largely due to the theoretical appeal of the multi-routing algorithm (i.e., PT is designed as a multi-path path builder to provide a more realistic representation of an urbanized environment with multiple transit options) and the significant runtime reduction associated with it.

Subsequently, COG staff implemented the PT Multipathing algorithm in the Gen3 Model in May 2022 based on an example implementation from the PT Arlington Model, which was developed by Bentley, the developer of the Cube software, as part of the Arlington County Travel Model.[24] Some of the specifications were further modified based on the RSG suggestions.

When examining the transit skims resulting from a test model run, however, COG staff noticed a hyperpath issue in the proposed PT Multipathing implementation.[25] Specifically, staff found that the multiple "hyperpaths" created by the PT multi-path path builder for commuter rail and "Bus + Metrorail" sub-modes did not always use the transit mode(s) of interest. Joel Freedman and Bill Woodford, the modeling experts at RSG, commented that the hyperpath issue is common in strategic transit path-finding algorithms and that multipath algorithms are probably not fully compatible with nested transit choice models. After reviewing the COG implementation files and testing different methodologies to address the hyperpath issue, Filippo Contiero, a PT expert at Bentley, acknowledged that there is no perfect solution in the current PT algorithm.

COG staff proposed a workaround to address the hyperpath issue in the Gen3 Model. The implementation and testing results of this workaround are detailed in a technical memo dated August 16, 2022.[26] While the proposed workaround resolved the hyperpath issue for commuter rail, it did not fully address the issue for "Bus + Metrorail" due to a discrepancy in the proposed methodology. Following Joel's suggestion, COG staff decided to adopt the workaround partially and move forward with the PT multipathing implementation that includes this partial fix to the hyperpath issue. Filippo concurred with this decision.

This section documents the implementation of the PT multi-pathing algorithm as well as the partial fix for the Gen3 Model.

Implementation

Switching from PT best-pathing to PT multipathing did not involve modifying any script file in the base Gen3 Model (i.e., the Gen3, Phase 1, Model received from RSG in January 2022). Instead, PT multipathing was implemented by modifying the PT factor files (.FAC) that are included as part of the transit inputs files (\inputs\trn). Specifically, the following changes were made to the factor files:

- Route enumeration parameters (e.g., ***AONMAXFERS, MAXFERS, EXTRAXFERS1, EXTRAXFERS2, SPREADFACT, REWAITMAX, RECOSTMAX***, etc.) and route evaluation parameters (e.g., ***ALPHA, LAMBDAW, LAMBDAA, CHOICECUT, SERVICEMODEL***, etc.) were either added or modified to enable multi-routing in PT. The

[24] See, for example, Christine Sherman Baker and Filippo Contiero, "Modeling Public Transport in the Arlington Co. Tour-Based Travel Model" (Washington, D.C., January 28, 2022), https://www.mwcog.org/events/2022/1/28/travel-forecasting-subcommittee/.
[25] See, Feng Xie and Meseret Seifu to Files, "Plotting Commuter Rail In-Vehicle Time (IVT) Skims based on the Gen3 Model with Proposed Public Transport (PT) Implementation", MWCOG/TPB Memorandum, June 7, 2022. Note that the term "hyperpath" is defined by Nguyen, Sang, Stefano Pallottino, and Michel Gendreau. "Implicit Enumeration of Hyperpaths in a Logit Model for Transit Networks." Transportation Science 32, no. 1 (February 1, 1998): 54–64. https://doi.org/10.1287/trsc.32.1.54
[26] See, Feng Xie to Files, "Implementing a Workaround to Address the Hyperpath Issue in the Public Transport (PT) Multipathing Implementation for the Gen3 Travel Model", MWCOG/TPB Memorandum, August 16, 2022.

specifications for those parameters were borrowed from those in the PT Arlington Model.[27]

- Mode-to-mode transfer penalties (***XFERPEN, XFERCONST*** and ***XFERFACTOR***) were added. Penalty values were specified per suggestion from RSG.
- Value of time (***VALUEOFTIME***) by mode, which was originally uniform across all transit modes, was updated per suggestion from RSG to reflect the higher value of time for Metrorail and commuter rail riders.
- On-board transit runtime factors (***RUNFACTOR***) by mode were kept the same as those in the base model, except for light rail (Mode 5), whose runtime factor was updated to be consistent with Bus Rapid Transit (Mode 10).
- Boarding penalties (***BRDPEN***) by mode were kept unchanged, except that the boarding penalty for Metrorail (Mode 4) was changed from 4 minutes to 5 minutes for off-peak periods. In the Gen3, Phase 1, Model, the boarding penalty for Metrorail was the only difference between the peak-period factor files (AM_TRN.FAC|PM_TRN.FAC) and off-peak factor files (MD_TRN.FAC|NT_TRN.FAC). This change made PT factor files identical across all time periods.
- Different initial and transfer wait curve definitions (***IWAITCURVE, XWAITCURVE***) specified in the PT factor files and system file (TSYSD.PTS) were tested per suggestion from RSG but were not adopted due to problematic pathtracing results. Instead, the original wait curve definitions from the TPB's developmental Gen2/Ver. 2.5 Model were used. The testing resulted in a different time stamp for the system file, but its content did not change relative to the base model.
- PT fare specifications (***FARESYSTEM, OPERATOR***), which were proposed by COG staff in a developmental Gen2/Ver. 2.3 Model[28] and were later incorporated into the Gen3 Model, are kept unchanged.
- Specifications for the ***MUSTUSEMODE*** keyword[29] by transit submode were kept unchanged during the implementation of PT multipathing. However, they were modified later as part of the partial fix to the hyperpath issue, which will be discussed in the next section.

Implementing A Partial Fix to the Hyperpath Issue

In the August 16 memo, COG staff proposed a workaround to address the hyperpath issue for both commuter rail and "Bus + Metrorail". As part of the workaround implementation, COG staff created two additional sets of PT factor files (AM|MD|PM|NT_TRN_CR.FAC and AM|MD|PM|NT_TRN_BM.FAC) for commuter rail and "Bus + Metrorail", respectively.

The fix for commuter rail was straightforward. COG staff simply changed the specifications for ***MUSTUSEMODE*** to "***MUSTUSEMODE***=4" in the commuter rail factor files, which enforces the use of at least one commuter rail (Mode 4) link on every route enumerated or evaluated for commuter rail and thus eliminates the hyperpath issue for the commuter rail sub-mode.

---

[27] The PT Arlington Model includes multiple user classes, and the route enumeration/evaluation parameters slightly differ by user class. The Gen3 Model, which includes a single user class in PT, adopts the parameters for the most representative user class (i.e., User Class 1) in the PT Arlington Model.
[28] See, Feng Xie to Mark Moran, "Proposed Cube Public Transport (PT) Fare Systems in the TPB Ver. 2.3 Travel Model", MWCOG/TPB Memorandum, September 2, 2020.
[29] ***MUSTUSEMODE*** specifies required transit modes in a route for enumeration or evaluation,

However, the methodology proposed to address the hyperpath issue for “Bus + Metrorail” was much more complicated because there is currently no perfect solution in the PT algorithm. In a nutshell, the proposed methodology involved the following steps:

I. The specifications for ***MUSTUSEMODE*** were changed to “***MUSTUSEMODE***=3,5” in the “Bus + Metrorail” factor files, which enforces the use of at least one Metrorail (Mode 3) <u>or</u> light rail (Mode 5) link[30] in the “Bus + Metrorail” path building.

II. An additional transit skim matrix (herein referred to as “MW[x]”) was computed, which indicates the probability of taking the routes that use both bus and Metrorail/light rail links (herein referred to as “B/M” routes), as opposed to the routes that use Metrorail/light rail modes only (herein referred to as “M/O” routes).

III. Transit skims for “Bus + Metrorail” were re-factored based on the value of MW[x]:
   a. If MW[x] equals 0, there are no B/M routes being built, and thus final transit skims for the “Bus + Metrorail” sub-mode are zeroed out.
   b. If MW[x] equals 1, all the enumerated or evaluated routes are B/M routes, and thus the final transit skims are kept unchanged.
   c. If MW[x] is between 0 and 1, the routes built for the “Bus + Metrorail” submode include both M/O routes and B/M routes. A convoluted refactoring method is used to derive the transit skims just for B/M routes.

After reviewing the August 16 memo, Joel (RSG) recommended a partial solution that would include Step I to Step III.b of the proposed workaround, but skip Step III.c. COG staff agreed with the RSG recommendation on the following grounds:

- The refactoring method in Step III.c was developed based on the assumption that the M/O routes built for the “Bus + Metrorail” sub-mode are identical to the M/O routes built for the “Metrorail Only” sub-mode. While both Bentley and COG staff suspected that the assumption is valid for the majority of cases,[31] there is a small chance that the two sets of routes are slightly different, as the minimum-cost paths are developed on different transit networks (one includes bus links, but the other not). A detailed discussion of this discrepancy and its implications on the resulting transit skims can be found in the August 16 memo.
- COG staff found that the Origin/Destination (O/D) pairs falling into the categories of Step III.a and Step III.b accounted for about 97% of all O/Ds. On the other hand, the O/D pairs that needed to be addressed by the imperfect refactoring method in Step III.c accounted for only about 3% of all cases. Joel suggested that, by skipping Step III.c, the effects of using the incorrect skims for some 3% of the OD pairs on the subsequent mode choice and transit assignment models would be very small.
- Furthermore, Joel pointed out that even if the refactoring method in Step III.c is perfectly correct, it would fix only the transit skims for the “Bus + Metrorail” submode, but not the route set and the associated “trip leakage” issue in transit assignment. Thus, he was concerned that “calculating corrected skims leads to an inconsistency between the path attributes fed to mode choice versus the paths to which the resulting trips are assigned, which can make debugging the results difficult”.

---

[30] Note that light rail is treated as Metrorail in transit path building in the Gen3 Model.

[31] Based on the pathtrace summaries for a limited number of origin/destination (O/D) pairs, the two sets of M/O routes were indeed identical for the majority of O/D pairs, but Bentley and COG staff could not find a way to quantify the percentage of all 3722 by 3722 O/D pairs where this assumption held.

For the reasons stated above, COG and RSG decided to move forward with a PT Multipathing implementation with the partial fix to the hyperpath issue. In addition to the PT factor files that were specifically created for commuter rail and “Bus + Metrorail” submodes, the implementation of the partial fix also involved updating five Cube scripts in the Gen3 Model. Specifically,

- The “Transit_Skims_PT_AB.s” script and “Transit_Skims_PT_MR.s” script include minor, aesthetic changes that don’t affect modeling results.
- The “Transit_Skims_PT_CR.s” script was modified to address the hyperpath issue found in commuter rail path building.
- The “Transit_Skims_PT_BM.s” script was modified to partially address the hyperpath issue found in “Bus + Metrorail” path building.
- The “PT_asgn_CR.s” script was modified to address an issue arising from the above mentioned change to the “***MUSTUSEMODE***” specification for commuter rail. Due to the change, PT can no longer build any path for a small number of external commuter rail trips (fewer than 800) and thus reports a fatal error during transit assignment. After consulting with RSG and BMG, the sub-contractor who developed the external commuter rail trip table based on the transit on-board surveys (TOBS), revisiting the TOBS data and recoding the transit mode for these trips would take a significant amount of time and resources, and COG staff provided a quick fix to this issue by removing the few unassigned external commuter rail trips from transit assignment.

COG staff transmitted the implementation files to RSG on September 13 through Box.[32] RSG staff subsequently incorporated them into the Gen3, Phase 2, Model code and used them for the Gen3, Phase 2, Model development work.

6.4 POPULATIONSIM INPUTS AND PROCESS

PopulationSim is a Population Synthesizer, which is software designed to develop a representative population for the region. Setting up and running the Population Synthesizer is discussed in the MWCOG Population Synthesizer Final Report[33]. This section includes only a discussion of the outputs, data preparation and application, scenario applications, controls and settings, and validation notes. Validation statistics of the population synthesizer is included in the final report and in the MWCOG Validation Summary[34].

Population Synthesis outputs

As the scripts run, data will be downloaded, processed, and stored in additional folders within the root directory.

PopulationSim input data (seed sample data, marginal controls, and raw downloads from the US Census Bureau) are stored in the data/ directory. The following shows the appearance of

[32] Box Link: https://app.box.com/s/ntxh6l5j87ly889wvxuvxztrbk8p3xbs
[33] RSG. MWCOG Population Synthesizer. Final Report. Metropolitan Washington Council of Governments, National Capital Region Transportation Planning Board, August 4, 2021.
[34] RSG. MWGOC Gen3 Model Calibration and Validation Report. November 20, 2023.

the directory after a base year run (2018). The size of this directory following the downloading and processing of the data is approximately **2 GB**.

```
data/
├── Census/               Raw data from US Census Bureau
├── PUMS/                 Raw data from ACS PUMS
├── TAZ/                  TAZ shapefile from MWCOG
├── land_use/             Round 9.1a Cooperative Forecast DBF files
├── xwalk/                Geographic crosswalks needed for pre-processing
├── control_taz_2018.csv  Marginal controls at TAZ level
├── geo_cross_walk.csv    TAZ to PUMA crosswalk required by PopulationSim
├── seed_hh.csv           Residential seed sample household data
├── seed_hh_gq.csv        Group quarter seed sample household data
├── seed_per.csv          Residential seed sample person data
└── seed_per_gq.csv       Group quarter seed sample person data
```

PopulationSim output data appears in the `output/` directory. This includes the raw output from PopulationSim, zip archives of the output, and the combined residential/group quarters synthetic population data. The following shows the appearance of the directory after a base year run (2018).

```
output/
├── output_gq_2018/                         group quarters output + validation
├── output_res_2018/                        residential output + validation
├── combined_synthetic_hh_2018.csv          synthetic data
├── combined_synthetic_per_2018.csv         synthetic data
└── popsim_output_2018_20201001_1916.zip    archive of PopulationSim output
```

After one run of PopulationSim, the size of this directory is approximately **2 GB**.

Data preparation and application

There are two main data processing steps in the MWCOG Population Synthesizer: 1) the preparation of inputs for the PopulationSim software ("pre-processing") and 2) the processing of the PopulationSim outputs ("post-processing"), such as combining Group Quarters and Residential outputs and preparation of validation charts and summaries. The following sub-sections present the details of data preparation and application of the MWCOG Population Synthesizer.

Input data preparation

The main data inputs to PopulationSim are:

- A disaggregate population sample (seed sample)
- Marginal control distributions (control variables)[35]

[35] In the context of List Balancing or Iterative Proportional Fitting, marginal controls refer to the row and column totals of the seed table.

PopulationSim can work with both household-level and person-level controls. The controls can also be specified at multiple geographic levels. The geographic resolution of the seed sample is referred to as the “seed” geography. The marginal controls can be specified at the level of the seed geography or any number of sub-seed geographies. The marginal controls can also be specified at a “meta” geographic level which is above the seed geography. For the MWCOG implementation, the PUMS dataset was used as the seed sample, which is available at the Public Use Microdata Area (PUMA) level. The marginal controls are generated at the TAZ level from Census data (ACS and Decennial Census) and multi-year land use forecasts. As a result, the hierarchical geographic structure for the MWCOG implementation is defined as follows:

- Region
- PUMA
- TAZ

A geographic crosswalk file defines the hierarchical structure of the various geographies. The crosswalk between TAZ and PUMA is created by one of the pre-processing scripts, `04_create_crosswalk.py,` and formatted for PopulationSim in `06_create_controls.py`.

Seed sample

The seed sample data are generated by the script `05_create_seed_sample.py`. The main requirement for the seed sample is that it should be representative of the modeling region. The seed sample must contain the appropriate fields needed to specify various marginal controls. Also, it must contain variables that are needed for the ABM but that are not specified as controls. The ACS PUMS data satisfies these requirements and was used as the seed sample. The 2014-2018 5-year ACS PUMS data is the most recent vintage of the PUMS sample and the closest to the selected base year (2018). PUMS data for District of Columbia, Maryland, Virginia, and West Virginia were used. The pre-processing scripts filter the PUMS sample to include only those records belonging to the PUMAs overlapping the modeling region.

Marginal controls

The control data are generated by the script `06_create_controls.py`. The residential marginal controls are produced from the 2018 ACS 5-year dataset downloaded at the tract level, based on specifications in `configs/census_variables_needed.csv` (see also `03_get_census.py`). The tract-level data are aggregated to TAZ level based on the area fraction of each Census tract that overlaps each TAZ. The group quarters data are not available at the sub-state level in the ACS 2018 sample so the 2010 Summary File 1 Census data were used at the block level. The block-level data were aggregated to TAZ by summing blocks within each TAZ. Finally, the Census/ACS counts were scaled to the Round 9.1a Cooperative Forecasts for each simulation year and each TAZ by calculating the ratio between the forecast data total and the Census/ACS data total within TAZ and multiplying each Census/ACS count by this adjustment factor. Table 3433 presents the list of control variables specified in the MWCOG Population Synthesizer.

TABLE 34. DATA SOURCES FOR SEED SAMPLE AND MARGINAL CONTROLS

Variable	Categories	PUMS Field	Control Source
Total HH		WGTP	Round 9.1a Forecasts
HH Size	1, 2, 3, 4+	NP	2018 ACS 5-year. Census Tract [Table S2501]
HH Income	0-$25K, $25K-$50K, $50K-$100K, $100k-$150K, $150k-$200K, $200K+	HINCP	2018 ACS 5-year. Census Tract [Table B19001]
Number of Workers	0, 1, 2, 3+	ESR	2018 ACS 5-year. Census Tract [Table B08202]
Presence of Children	0, 1	HUPAC	2018 ACS 5-year. Census Tract [Table S1101]
Person Age	0-4, 5-19, 20-34, 35-64, 65+	AGEP	2018 ACS 5-year. Census Tract [Table S0101]
Person Race	White, Hispanic, Black, Asian, Other	HISP, RAC1P	2018 ACS 5-year. Census Tract [Table DP05]
Total GQ units		TYPE	Round 9.1a Forecasts
GQ Type	University, Military, Other Non-Institutional (group homes, missions, shelters, etc.)	SCHG, MIL, TYPE	2010 Census SF1 [Table P042]

HH Size Control Adjustments

The household size controls were further adjusted to resolve the inconsistencies between the TAZ-level population inferred from the Census/ACS household size distributions and the TAZ-level household and population estimates from the Round 9.1a Cooperative Forecasts. A lookup table between average household size and household size distribution (1,2,3, 4+ categories) was computed from the Tract level Census/ACS data. The household size controls were recomputed for TAZs with problematic household size controls. To identify problematic TAZs, minimum and average implied populations were computed for each TAZ using the existing household size controls. The minimum implied population is computed by counting only 4 persons for the 4-plus household size category. For the average implied population, the average number of persons in a 4-plus person household is used. Problematic TAZs are the ones for which either of the following conditions is true.

1. Round 9.1a Cooperative Forecasts population < minimum implied population
2. Round 9.1a Cooperative Forecast population > 1.5 * average implied population

Using the above rules, about 600 TAZs were tagged. The household size controls were recomputed for these TAZs using the household size distribution lookup table. This fixed the inconsistencies for more than 90% of the problematic TAZs. **Error! Reference source not**

found. shows some example TAZs that failed one of the above checks and how their household size controls were adjusted.

TABLE 35: EXAMPLE HOUSEHOLD SIZE CONTROL ADJUSTMENTS

	ROUND 9.1A CF			PopulationSim Controls						Data Checks		
TAZ	Total HH	HH Persons	Avg HH Size	hh_1	hh_2	hh_3	hh_4p	Min implied persons	Avg implied persons	Check 1	Check 2	Version
159	1,211	4,098	3	502	420	87	202	2,411	2,546	✓	✗	Initial
159	1,211	4,098	3	171	335	244	461	3,417	3,726	✓	✓	Adjusted
411	660	1,396	2	190	155	140	175	1,620	1,737	✗	✓	Initial
411	660	1,396	2	283	227	80	70	1,257	1,304	✓	✓	Adjusted
1500	1,578	2,402	2	855	539	139	45	2,530	2,560	✗	✓	Initial
1500	1,578	2,402	2	1,018	459	69	32	2,271	2,292	✓	✓	Adjusted
3669	2,242	6,065	3	371	761	252	858	6,081	6,656	✗	✓	Initial
3669	2,242	6,065	3	586	745	383	528	5,337	5,691	✓	✓	Adjusted

Scenario Applications

The marginal controls for PopulationSim need to be updated for modeling scenarios involving a change in demographics. To prepare the synthetic population for such scenarios, the user must create appropriate marginal control data. For example, when modeling an aging population scenario, the person-age controls must be adjusted to represent an aging population. For such a scenario, however, just updating the age controls will not be sufficient. The aging population will likely impact other distributions such as household income and auto ownership. The user must evaluate such effects and updated all marginal distributions to represent demographic distributions under an aging population scenario. The modified controls can be specified at a different geographic level compared to the base year. This depends on the availability of the control data at that geographic level and the accuracy of the data. Typically, the seed data remains the same.

Controls and settings

This section presents the final settings for the MWCOG Population Synthesizer. Different settings and configurations were tried during the initial testing of PopulationSim, such as varying the maximum expansion factor, combining controls, and altering importance factors on controls. The settings and configuration that resulted in the best overall validation performance were retained as the final version. The final version of the MWCOG PopulationSim uses a maximum expansion factor of 30. This is the default maximum expansion factor used by other agencies such as the Metropolitan Transportation Council (MTC), Oregon Department of Transportation, Portland Metro, Fresno Council of Governments, and Metropolitan Council (Minneapolis) in their PopulationSim implementation. Table 3635 (residential) and Table 3736 (group quarters) present the final set of controls and importance factors.

TABLE 36: MWCOG POPULATIONSIM MARGINAL CONTROLS, RESIDENTIAL

Target	Geography	Seed Table	Importance
hh_total	TAZ	households	1,000,000,000
hh_size_1	TAZ	households	5000
hh_size_2	TAZ	households	5000
hh_size_3	TAZ	households	5000
hh_size_4_plus	TAZ	households	5000
hh_inc_0_25	TAZ	households	1000
hh_inc_25_50	TAZ	households	1000
hh_inc_50_100	TAZ	households	1000
hh_inc_100_150	TAZ	households	1000
hh_inc_150_200	TAZ	households	1000
hh_inc_200_plus	TAZ	households	1000
hh_worker_0	TAZ	households	5000
hh_worker_1	TAZ	households	5000
hh_worker_2	TAZ	households	5000
hh_worker_3plus	TAZ	households	5000
hh_w_kid	TAZ	households	1000
hh_wo_kid	TAZ	households	1000
per_age_0_4	TAZ	persons	1000
per_age_5_19	TAZ	persons	1000
per_age_20_34	TAZ	persons	1000
per_age_35_64	TAZ	persons	1000
per_age_65plus	TAZ	persons	1000
per_race_anyhispanic	TAZ	persons	5000
per_race_white	TAZ	persons	5000
per_race_black	TAZ	persons	5000
per_race_asian	TAZ	persons	5000
per_race_other	TAZ	persons	5000

TABLE 37: MWCOG POPULATIONSIM MARGINAL CONTROLS, GROUP QUARTERS

Target	Geography	Seed Table	Importance
gq_noninst	TAZ	households	1,000,000,000
gq_univ	TAZ	persons	1000
gq_mil	TAZ	persons	1000
gq_other	TAZ	persons	1000

Validation

One of the most critical steps in population synthesis is validating the final synthetic population. Validation can give clues about inconsistencies among controls, data processing errors, or misspecification of any settings. This section describes the validation procedures and then presents the validation results from the final base-year (2018) run and future-year (2045) run.

Validation procedures

PopulationSim reports the difference between the synthesized totals and the control totals for all the controls at each geographic level. The Python validation script generates advanced summary statistics and validation plots. These are described briefly below.

Validation summary statistics

The validation script reports the following information for each control: the total number of records (household/person) desired by the control, the total number of records synthesized, the difference between the synthesized total and the control total, and the percentage difference. Statistics that inform us of convergence at a more disaggregate level are also computed – please note that these statistics are computed for the geography at which the controls are specified *i.e.* TAZ or Region as the case might be. The following three statistics are computed as a part of this exercise:

1. the average percentage difference between the control totals and the synthesized totals,

$$Average\ Percentage\ Difference\ (APD) = \frac{\sum_{i=1}^{N}(PercDiff_i)}{N}$$

   Where, $PercDiff_i$ is the percentage difference between the control totals and the synthesized totals at each geography and N is the total number of geographies.

2. the standard deviation (STDEV) of the percentage difference – this measure informs us of how much dispersion from the average exists,

$$STDEV = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(PercDiff_i - APD)^2}$$

3. the percentage root mean square error (PRMSE) - an indicator of the proximity of synthesized and control totals.

$$PRMSE = \left(\frac{\sqrt{\frac{\sum_{i=1}^{N} (Diff_i)^2}{N}}}{\sum_{i=1}^{N} Control_i} \right) * N * 100$$

   Where, $Diff_i$ is the difference between the control totals and the synthesized totals at each geography and N is the total number of geographies.

4. The number of geographies for which the control is non-zero (N) is also reported.

Traditionally, the performance of population synthesis is assessed at the regional level. These disaggregate statistics provide a tool to investigate data inconsistencies and misspecification errors. In the absence of any data inconsistencies and errors, the average percentage differences are expected to be close to zero across all controls. However, the observed data usually has some inconsistencies. Therefore, an average percentage difference in the range of -5% to +5% is considered acceptable. Please note that the control on the total number of households, in any case, should match perfectly. The sum of average percentage differences across all control categories should also be close to zero. For example, the average percentage difference on the 1-person household control may not be close to zero but the sum across all household size categories should be close to zero. A systematic pattern across all control categories may indicate issues in the control data or a specification error. The STDEV and PRMSE should ideally be in the -20% to +20% range. However, these statistics are very sensitive to the actual control values. For example, a control on a minority segment of the population may have a very low control value for each geography. A difference of 5 on a control value of 100 results in a percentage difference of 5% while the same difference on a control value of 20 results in a percentage difference of 20%. A higher percentage difference is generally acceptable for zones with a low control value, but the high percentage differences increase the STDEV and PRMSE. Therefore, a higher STDEV or PRMSE may be acceptable for controls with low control values that are hard to match exactly.

Validation chart

The validation chart is a visualization of the disaggregate summary statistics – mean percentage difference, STDEV, and PRMSE of percentage differences. A form of dot and whisker plot is generated for each control where the dots are the mean percentage differences and horizontal bars are twice the STDEV or PRMSE centered around zero.

Frequency distribution plots[36]

These are simply frequency distribution plots of differences between control and synthesized values across the geography at which the controls were specified.

---

[36] Frequency distribution plot example:
https://activitysim.github.io/populationsim/validation.html#frequency-distribution-plots

Expansion factor distribution[37]

While a synthetic population may match the controls well, it is important to know how uniform the household weights are, and how different they are from the initial weights. The closer the final weights are to the initial PUMS weight, the higher is the chance of matching the distribution of uncontrolled variables. An expansion factor is computed for each record in the PUMS data as total final weight/initial weight. A distribution plot of these expansion factors is created for each PUMA. A good synthetic population would have most of these expansion factors as close to one as possible.

6.5 ACTIVITYSIM RESIDENT TRAVEL MODEL

ActivitySim is an activity-based travel model that simulates transportation decisions throughout a person's day. This is done by reading a synthetic population of households and persons in the model region and modeling decisions that are made throughout the day. This model begins with long-term choices - decisions that are not made on a daily basis, such as work and school locations. This includes whether a person works from home full-time or out-of-home, and then if they work out-of-home, the frequency that they telecommute. The second set of choices are daily activity pattern and mandatory tour frequency pattern, which dictate the makeup of tours in a day. The third set of choices are tour-level choices, which determine tour mode, tour scheduling, and sometimes the tour destination (mandatory tours - work and school tours - have their primary tour destination set by the long-term work and school location choice models). Finally, the fourth set of choices are trip-level choices, which determine the individual trip choices for an entire tour.

Long-Term Models

These steps model the choices that are not easily changed by persons in the model and choices that are largely based on those outcomes. This group of models includes the following models, in order:

1. School Location
2. Work From Home
3. Workplace Location
4. Transit Fare Subsidy
5. AV Ownership
6. Auto Ownership
7. Vehicle Type Choice
8. Free Parking

[37] Expansion factor distribution example: https://activitysim.github.io/populationsim/validation.html#expansion-factor-distributions

9. Telecommute Frequency

School Location

The usual school location choice models assign a usual school location for the primary mandatory activity of each child and university student in the synthetic population. The models are composed of a set of accessibility-based parameters (including one-way distance between home and primary destination and the tour mode choice logsum - the expected maximum utility in the mode choice model which is given by the logarithm of the sum of exponentials in the denominator of the logit formula) and size terms, which describe the quantity of grade-school or university opportunities in each possible destination.

The school location model is made up of four steps:

1. Sampling - selects a sample of alternative school locations for the next model step. This selects 30 locations from the full set of model zones using a simple utility.
2. Logsums - starts with the table created above and calculates and adds the mode choice logsum expression for each alternative school location.
3. Simulate - starts with the table created above and chooses a final school location, this time with the mode choice logsum included.
4. Simulation constraint - compare modeled zonal destinations to target zonal size terms and re-simulate choices until convergence.

These steps are repeated until shadow pricing convergence criteria are satisfied or a max number of iterations is reached.

Work From Home

The work from home model determines if a worker will work from home and not have a usual out-of-home workplace. This situation is where a company may not have a local office, or the employee is a sole proprietor with no other office. This is different from telecommuting where a worker that telecommutes HAS a usual out-of-home workplace.

The work from home model is a binary logit model. It is run before the workplace location model and workers that are determined to work from home are filtered out and not simulated in the workplace location model, free parking model, or telecommute frequency model.

Workplace Location

The usual work location choice models assign a usual work location for the primary mandatory activity of each worker that works out-of-home in the synthetic population. The models are composed of a set of accessibility-based parameters (including one-way distance between home and primary destination and the tour mode choice logsum - the expected maximum utility in the mode choice model which is given by the logarithm of the sum of exponentials in the denominator of the logit formula) and size terms, which describe the quantity of work opportunities in each possible destination.

The work location model is made up of four steps:

1. Sample - selects a sample of alternative work locations for the next model step. This selects 30 locations from the full set of model zones using a simple utility.
2. Logsums - starts with the table created above and calculates and adds the mode choice logsum expression for each alternative work location.
3. Simulate - starts with the table created above and chooses a final work location, this time with the mode choice logsum included.
4. Simulation constraint - compare modeled zonal destinations to target zonal size terms and re-simulate choices until convergence.

These steps are repeated until shadow pricing convergence criteria are satisfied or a max number of iterations is reached.

Transit Fare Subsidy

The transit fare subsidy estimates the amount of transit subsidy, if any, for workers in the model. This model uses a continuous distribution (listed in constants.yaml as the transit_subsidy array). The model uses a random number generator to determine the subsidy amount.

Automated Vehicle Ownership

The Autonomous Vehicle (AV) ownership model predicts whether a household will own an AV. This is a binary logit model. For calibration purposes, this model uses a constant set to -999 to disable owning an AV. For application purposes requiring the AV model, that constant will need to be disabled or set to 0 to allow the model to determine if a household will own an AV. This model is adapted from previous DaySim work[38,39].

Auto Ownership

The auto ownership model selects a number of autos for each household in the simulation. The primary model components are household demographics, zonal density, and accessibility. The model is a multinomial logit model with choices of 0 autos, 1 auto, 2 autos, 3 autos, or 4+ autos. In the case that a household owns an AV, there is a specification that disables the 0 auto and 4+ auto alternatives.

Vehicle Type Choice

The vehicle type choice model selects a vehicle type for each household vehicle. A vehicle type is a combination of the vehicle's body type, age, and fuel type. For example, a 13-year-old gas powered van would have a vehicle type of van_13_gas. This model is a multinomial logit model

[38] Bradley, Mark. AVs and TNCs in Daysim. Presentation to SACOG. 1/17/2009. https://www.sacog.org/sites/main/files/file-attachments/avs_and_tnc_in_daysim-sacsim-rsg_0.pdf?1548293104

[39] Ou, Yanmei and Griesenbeck, Bruce. Estimating the Potential Impacts of AVs and TNCs using ActivityBased Travel Demand Model in MTP/SCS Scenario Development. Presentation at 2018 Innovations in Travel Modeling Conference, Atlanta, GA. 2018. https://onlinepubs.trb.org/onlinepubs/Conferences/2018/ITM/YOu.pdf

with simultaneous choice of body type, age, and fuel type. More information on this model can be found on the ActivitySim Documentation Website[40].

Vehicle Allocation

The vehicle allocation model selects which vehicle would be used for a tour of given occupancy. The alternatives for the vehicle allocation model consist of the vehicles owned by the household and an additional non household vehicle option. (Zero-auto households would be assigned the non-household vehicle option since there are no owned vehicles in the household). A vehicle is selected for each occupancy level set by the user such that different tour modes that have different occupancies could see different operating characteristics. The output of the vehicle allocation model is appended to the tour table with column names `vehicle_occup_{occupancy}` and the values are the vehicle type selected.

Free Parking

The Free Parking Eligibility model predicts the availability of free parking at a person’s workplace. It is applied for people who work in zones that have parking charges, which are generally located in the Central Business Districts. The purpose of the model is to adequately reflect the cost of driving to work in subsequent models, particularly in mode choice. This is a binary logit model.

Telecommute Frequency

Telecommuting is defined as workers who work from home instead of going to work. This model is only applied to workers with a regular workplace outside of home.

For all workers that work out of the home, the telecommute models predicts the level of telecommuting. This is a multinomial logit model with alternatives of the frequency of telecommuting in days per week (0 days, 1 day, 2 to 3 days, 4+ days).

Daily Models

There are two daily models in ActivitySim that bridge the gap between long-term models and tour models. These are the Coordinated Daily Activity Pattern and the mandatory tour frequency model, which are run in that order.

Coordinated Daily Activity Pattern

The Coordinated Daily Activity Pattern (CDAP) model predicts the choice of daily activity pattern (DAP) for each member in the household, simultaneously. The DAP is categorized in to three types as follows:

[40] https://activitysim.github.io/activitysim/v1.2.0/models.html#vehicle-type-choice

Mandatory (M): the person engages in travel to at least one out-of-home mandatory activity - work, university, or school. The mandatory pattern may also include non-mandatory activities such as separate home-based tours or intermediate stops on mandatory tours.

Non-mandatory (N): the person engages in only maintenance and discretionary tours, which, by definition, do not contain mandatory activities.

Home (H): the person does not travel outside the home.

The CDAP model is a sequence of vectorized table operations:

1. Create a person level table and rank each person in the household for inclusion in the CDAP model. Priority is given to full time workers (up to two), then to part time workers (up to two workers, of any type), then to children (youngest to oldest, up to three). Additional members up to five are randomly included for the CDAP calculation.
2. Solve individual M/N/H utilities for each person
3. Take as input an interaction coefficients table and then programmatically produce and write out the expression files for households size 1, 2, 3, 4, and 5 models independent of one another
4. Select households of size 1, join all required person attributes, and then read and solve the automatically generated expressions
5. Repeat for households size 2, 3, 4, and 5. Each model is independent of one another.

Mandatory Tour Frequency

The individual mandatory tour frequency model predicts the number of work and school tours taken by each person with a mandatory DAP. The primary drivers of mandatory tour frequency are demographics, accessibility-based parameters such as drive time to work, and household automobile ownership. It also creates mandatory tours in the data pipeline. This is a multinomial logit model with alternatives of 1 work, 2 work, 1 school, 2 school, and work+school alternatives. Not all alternatives are allowed for all person types, and that is set by using constants in the model specification to disable irrelevant alternatives (e.g. school pre-driving age children have a constant of -999 for the 1 work, 2 work, and work+school alternatives since it is very uncommon and/or against laws for children of this age to work).

Tour Models

There are sixteen total tour models. These models determine how, when, and sometimes why a person leaves their house.

The first tour model is mandatory tour scheduling, which determines when a person travels to and returns from work or school, if applicable. The next group of five models are joint tour models, the following three are non-mandatory models. The next two models allocate vehicles and simulate tour mode choices. Following that is a group of four at-work subtour models and finally, the stop frequency model.

Mandatory Tour Scheduling

The mandatory tour scheduling model selects a tour departure and duration period (and therefore a start and end period as well) for each mandatory tour. The primary drivers in the model are accessibility-based parameters such as the mode choice logsum for the departure/arrival hour combination, demographics, and time pattern characteristics such as the time windows available from previously scheduled tours. This model uses person Person Time Windows, which are adjacent time periods that are available for travel. Time windows are stored in a timetable table and each row is a person and each time period.

Joint Tour Models

This group of models estimates fully-joint tours, which are tours where multiple people from a household travel together for the entire tour. An example of this would be a family eating out, where all members of the household stay together for the entire tour.

This model includes its own frequency model that determines the number of joint tours, a composition and participation models to determine the household members to include on the tour, a destination choice model, and finally a scheduling model.

Joint Tour Frequency

The joint tour generation models are divided into three sub-models: the joint tour frequency model, the party composition model, and the person participation model. In the joint tour frequency model, the household chooses the purposes and number (up to two) of its fully joint travel tours in a typical day. It also creates joints tours in the data pipeline.

Joint Tour Composition

In the joint tour party composition model, the makeup of the travel party (adults, children, or mixed - adults and children) is determined for each joint tour. The party composition determines the general makeup of the party of participants in each joint tour in order to allow the micro-simulation to faithfully represent the prevalence of adult-only, children-only, and mixed joint travel tours for each purpose while permitting simplicity in the subsequent person participation model.

Joint Tour Participation

In the joint tour person participation model, each eligible person sequentially makes a choice to participate or not participate in each joint tour. Since the party composition model determines what types of people are eligible to join a given tour, the person participation model can operate in an iterative fashion, with each household member choosing to join or not to join a travel party independent of the decisions of other household members. In the event that the constraints posed by the result of the party composition model are not met, the person participation model cycles through the household members multiple times until the required types of people have joined the travel party.

Joint Tour Destination

The joint tour destination choice model operates similarly to the usual work and school location choice model, selecting the primary destination for travel tours. The only procedural difference between the models is that the usual work and school location choice models select the usual location of a mandatory activity whether or not the activity is undertaken during the travel day, while the joint tour destination choice model selects the location for an activity which has already been generated.

The tour's primary destination is the location of the activity that is assumed to provide the greatest impetus for engaging in the travel tour. In the household survey, the primary destination was not asked, but rather inferred from the pattern of stops in a closed loop in the respondents' travel diaries. The inference was made by weighing multiple criteria including a defined hierarchy of purposes, the duration of activities, and the distance from the tour origin. The model operates in the reverse direction, designating the primary purpose and destination and then adding intermediate stops based on spatial, temporal, and modal characteristics of the inbound and outbound journeys to the primary destination.

The joint tour destination choice model is made up of three model steps:

1. Sample - selects a sample of alternative locations for the next model step. This selects 30 locations from the full set of model zones using a simple utility.
2. Logsums - starts with the table created above and calculates and adds the mode choice logsum expression for each alternative location.
3. Simulate - starts with the table created above and chooses a final location, this time with the mode choice logsum included.

Joint Tour Scheduling

The joint tour scheduling model selects a tour departure and duration period (and therefore a start and end period as well) for each joint tour. This model uses Person Time Windows. The primary drivers in the models are accessibility-based parameters such as the auto travel time for the departure/arrival hour combination, demographics, and time pattern characteristics such as the time windows available from previously scheduled tours. The joint tour scheduling model does not use mode choice logsums.

Non-Mandatory Models

Non-Mandatory Tour Frequency

The non-mandatory tour frequency model selects the number of non-mandatory tours made by each person on the simulation day. It also adds non-mandatory tours to the tours in the data pipeline. The individual non-mandatory tour frequency model operates in two stages:

4. A choice is made using a random utility model between combinations of tours containing zero, one, and two or more escort tours, and between zero and one or more tours of each other purpose.
5. Up to two additional tours of each purpose are added according to fixed extension probabilities.

Non-Mandatory Tour Destination

The non-mandatory tour destination choice model chooses a destination zone for non-mandatory tours. The three step (sample, logsums, final choice) process also used for mandatory tour destination choice is used for non-mandatory tour destination choice.

Non-Mandatory Tour Scheduling

The non-mandatory tour scheduling model selects a tour departure and duration period (and therefore a start and end period as well) for each non-mandatory tour. This model uses Person Time Windows.

Tour Mode Choice Simulation

The mandatory, non-mandatory, and joint tour mode choice model assigns to each tour the “primary” mode that is used to get from the origin to the primary destination. The tour-based modeling approach requires a reconsideration of the conventional mode choice structure. Instead of a single mode choice model used in a four-step structure, there are two different levels where the mode choice decision is modeled: (a) the tour mode level (upper-level choice); and, (b) the trip mode level (lower-level choice conditional upon the upper-level choice).

The mandatory, non-mandatory, and joint tour mode level represents the decisions that apply to the entire tour, and that will affect the alternatives available for each individual trip or joint trip. These decisions include the choice to use a private car versus using public transit, walking, or biking; whether carpooling will be considered; and whether transit will be accessed by car or by foot. Trip-level decisions correspond to details of the exact mode used for each trip, which may or may not change over the trips in the tour.

The mandatory, non-mandatory, and joint tour mode choice structure is a nested logit model which separates similar modes into different nests to more accurately model the cross-elasticities between the alternatives. The eighteen modes are incorporated into the nesting structure specified in the model settings file. The first level of nesting represents the use a private car, non-motorized means, or transit. In the second level of nesting, the auto nest is divided into vehicle occupancy categories, and transit is divided into walk access and drive access nests. The final level splits the auto nests into free or pay alternatives and the transit nests into the specific line-haul modes.

The primary variables are in-vehicle time, other travel times, cost (the influence of which is derived from the automobile in-vehicle time coefficient and the persons’ modeled value of time), characteristics of the destination zone, demographics, and the household’s level of auto ownership.

Vehicle Allocation Model

The vehicle allocation model selects which vehicle would be used for a tour of given occupancy. The alternatives for the vehicle allocation model consist of the vehicles owned by the household and an additional non household vehicle option. (Zero-auto households would be assigned the non-household vehicle option since there are no owned vehicles in the household). A vehicle is selected for each occupancy level set by the user such that different tour modes that have different occupancies could see different operating characteristics.

The model has three occupancy levels: 1, 2, and 3.5. The auto operating cost for occupancy level 1 is used in the drive alone mode and drive to transit modes. Occupancy levels 2 and 3.5 are used for shared ride 2 and shared ride 3+ auto operating costs, respectively. Auto operating costs are selected in the mode choice pre-processors by selecting the allocated vehicle type data from the vehicles table. If the allocated vehicle type was the non-household vehicle, the auto operating costs uses the previous default value from the constants.yaml file. All trips and at-work subtours use the auto operating cost of the parent tour.

At-work Subtour Models

The at-work subtour model is applied to work tours to determine if a worker leaves and returns to work during the day. This could include a lunch tour, where a worker leaves the office location for lunch, or could include another tour where the worker leaves the office and returns to the office prior to returning home.

The models in this group include a frequency model to determine if and how many subtours are made, and a destination, scheduling, and a mode choice model for the subtours that are made.

At-work Subtour Frequency

The at-work subtour frequency model selects the number of at-work subtours made for each work tour. It also creates at-work subtours by adding them to the tours table in the data pipeline. These at-work sub-tours are travel tours taken during the workday with their origin at the work location, rather than from home. Explanatory variables include employment status, income, auto ownership, the frequency of other tours, characteristics of the parent work tour, and characteristics of the workplace zone.

Choosers: work tours

Alternatives:

- none
- 1 eating-out tour
- 1 business tour
- 1 maintenance tour
- 2 business tours
- 1 eating-out tour + 1 business tour

At-work Subtour Destination

The at-work subtours destination choice model is made up of three model steps:

6. Sample - selects a sample of alternative locations for the next model step. This selects X locations from the full set of model zones using a simple utility.
7. Logsums - starts with the table created above and calculates and adds the mode choice logsum expression for each alternative location.
8. Simulate - starts with the table created above and chooses a final location, this time with the mode choice logsum included.

At-work Subtour Scheduling

The at-work subtours scheduling model selects a tour departure and duration period (and therefore a start and end period as well) for each at-work subtour. This model uses Person Time Windows.

This model is the same as the mandatory tour scheduling model except it operates on the at-work tours and constrains the alternative set to available Person Time Windows. The at-work subtour scheduling model does not use mode choice logsums. The at-work subtour frequency model can choose multiple tours so this model must process all first tours and then second tours since isFirstAtWorkTour is an explanatory variable.

Choosers: at-work tours

Alternatives: alternative departure time and arrival back at origin time pairs WITHIN the work tour departure time and arrival time back at origin AND the person time window. If no time window is available for the tour, make the first and last time periods within the work tour available, make the choice, and log the number of times this occurs. Dependent tables: skims, person, land use, work tour

Outputs: at-work tour departure time and arrival back at origin time, updated person time windows

At-work Subtour Mode Choice

The at-work subtour mode choice model assigns a travel mode to each at-work subtour using the Tour Mode Choice model.

Stop Frequency

The stop frequency model assigns to each tour the number of intermediate destinations a person will travel to on each leg of the tour from the origin to tour primary destination and back. The model incorporates the ability to make more than one stop in each direction, up to a maximum of 3, for a total of 8 trips per tour (four on each tour leg).

Intermediate stops are not modeled for drive-transit tours because doing so can have unintended consequences because of the difficulty of tracking the location of the vehicle.

The stop frequency model's output is ultimately the trip list that moves the model from tours to trips.

Trip Models

There are five trip models. These models determine the actual purpose of the trip (which may not be the same as the tour), the destination of the trip, a purpose and destination model that is used to clean up trips that are unable to be assigned a destination in the trip destination model, a scheduling model, and finally a mode choice model.

Trip Purpose

For trip other than the last trip outbound or inbound, assign a purpose based on an observed frequency distribution. The distribution is segmented by tour purpose, tour direction and person type. Work tours are also segmented by departure or arrival time period.

Trip Destination

The trip (or stop) location choice model predicts the location of trips (or stops) along the tour other than the primary destination. The stop-location model is structured as a multinomial logit model using a zone attraction size variable and route deviation measure as impedance. The alternatives are sampled from the full set of zones, subject to availability of a zonal attraction size term. The sampling mechanism is also based on accessibility between tour origin and primary destination and is subject to certain rules based on tour mode.

All destinations are available for auto tour modes, so long as there is a positive size term for the zone. Intermediate stops on walk tours must be within 3 miles of both the tour origin and primary destination zones. Intermediate stops on bike tours must be within 8 miles of both the tour origin and primary destination zones. Intermediate stops on walk-transit tours must either be within 3 miles walking distance of both the tour origin and primary destination or have transit access to both the tour origin and primary destination. Additionally, only short and long walk zones are available destinations on walk-transit tours.

The intermediate stop location choice model works by cycling through stops on tours. The level-of-service (LOS) variables (including mode choice logsums) are calculated as the additional utility between the last location and the next known location on the tour. For example, the LOS variable for the first stop on the outbound direction of the tour is based on additional impedance between the tour origin and the tour primary destination. The LOS variable for the next outbound stop is based on the additional impedance between the previous stop and the tour primary destination. Stops on return tour legs work similarly, except that the location of the first stop is a function of the additional impedance between the tour primary destination and the tour origin. The next stop location is based on the additional impedance between the first stop on the return leg and the tour origin, and so on.

Trip Purpose and Destination

After running trip purpose and trip destination separately, the two models can be run together in an iterative fashion on the remaining failed trips (i.e., trips that cannot be assigned a destination). Each iteration uses new random numbers.

Trip Scheduling

For each trip, assign a departure half-hour based on an input lookup table of percents by tour purpose, direction (inbound/outbound), tour half-hour, and trip index.

The tour half-hour is the tour start half-hour for outbound trips and the tour end half-hour for inbound trips. The trip index is the trip sequence on the tour, with up to four trips per half tour.

For outbound trips, the trip depart half-hour must be greater than or equal to the previously selected trip depart half-hour.

For inbound trips, trips are handled in reverse order from the next-to-last trip in the leg back to the first. The tour end half-hour serves as the anchor time point from which to start assigning trip time periods.

Outbound trips on at-work subtours are assigned the tour depart half-hour and inbound trips on at-work subtours are assigned the tour end half-hour.

The assignment of trip depart time is run iteratively up to a max number of iterations since it is possible that the time period selected for an earlier trip in a half-tour makes selection of a later trip time period impossible (or very low probability). Thus, the sampling is re-run until a feasible set of trip time periods is found. If a trip can't be scheduled after the max iterations, then the trip is assigned the previous trip's choice (i.e., assumed to happen right after the previous trip) or dropped, as configured by the user. The trip scheduling model does not use mode choice logsums.

Alternatives: Available time periods in the tour window (i.e., tour start and end period). When processing stops on work tours, the available time periods are constrained by the at-work subtour start and end period as well.

Trip Mode Choice

The trip mode choice model assigns a travel mode for each trip on a given tour. It operates similarly to the tour mode choice model, but only certain trip modes are available for each tour mode. The correspondence rules are defined according to the following principles:

- Pay trip modes are only available for pay tour modes (for example, drive-alone pay is only available at the trip mode level if drive-alone pay is selected as a tour mode).
- The auto occupancy of the tour mode is determined by the maximum occupancy across all auto trips that make up the tour. Therefore, the auto occupancy for the tour mode is the maximum auto occupancy for any trip on the tour.

- Transit tours can include auto shared-ride trips for particular legs. Therefore, 'casual carpool', wherein travelers share a ride to work and take transit back to the tour origin, is explicitly allowed in the tour/trip mode choice model structure.
- Walking is allowed for any trip less than 3 miles.
- The availability of transit line-haul sub-modes on transit tours depends on the skimming and tour mode choice hierarchy. Free shared-ride modes are also available in walk-transit tours, albeit with a low probability. Paid shared-ride modes are not allowed on transit tours because no stated preference data is available on the sensitivity of transit riders to automobile value tolls, and no observed data is available to verify the number of people shifting into paid shared-ride trips on transit tours.

The trip mode choice model's explanatory variables include household and person variables, level-of-service between the trip origin and destination according to the time period for the tour leg, urban form variables, and alternative-specific constants segmented by tour mode.

6.6 TRUCK MODEL

The origin/destination truck trip generation rates are based on area type and land activity variables as shown in Table 3837. The truck trip generation model includes provisions to remove external trucks generated because external truck travel is accounted for exogenously. The truck trip generation process also includes network checks provisions to ascertain whether or not truck access from each TAZ to the highway network is valid. There are some zonal centroids in the regional network that have a single connection to a parkway where trucks are prohibited. In these types of cases, truck trip generation is suppressed. Finally, the truck model also considers a limited number of special generator TAZs, or locations where truck traffic generation is known to be more intensive. Global trip generation adjustments are applied to the special generator TAZs - The medium truck generation is factored by 2.70 while heavy trucks are factored by 5.3.

TABLE 38: TRUCK TRIP GENERATION RATES AS A FUNCTION OF TRUCK TYPE, AREA TYPE, AND LAND USE CATEGORY

Vehicle Type	Area Type	Land Use Category				
		Office	Retail	Industrial	Other	HH
Medium Truck (Single Unit 6+ Tires)	1 (CBD)	0.004	0.088	0.088	0.014	0.070
	2 - 4	0.005	0.125	0.125	0.020	0.100
	5	0.006	0.150	0.150	0.024	0.120
	6	0.006	0.150	0.150	0.024	0.120
Heavy Truck (All Combination Vehicles)	1 (CBD)	0.001	0.027	0.055	0.002	0.011
	2 - 4	0.002	0.039	0.078	0.003	0.015
	5	0.002	0.043	0.086	0.003	0.017

	6	0.002	0.043	0.086	0.003	0.017

Ref: I:\ateam\docum\FY09\Version2.3_modelDoc_2008-07\tgcheck.xls

6.7 COMMERCIAL VEHICLE MODEL

The trip generation of zonal commercial vehicle trips is developed with Equation 3[41]

EQUATION 3: COMMERCIAL VEHICLE PRODUCTIONS

$$\boldsymbol{CommVehP_i} = (\mathbf{0.056} * \boldsymbol{INDEMP_i} + \mathbf{0.168} * \boldsymbol{OFFEMP_i} + \mathbf{0.494} * \boldsymbol{RETEMP_i} + \mathbf{0.082} * \boldsymbol{OTHEMP_i} + \mathbf{0.13} * \boldsymbol{HH_i}) * \boldsymbol{ATFAC_a}$$

Where:

indemp = industrial employment
offemp = office employment
retemp = retail employment
othemp = other employment
HH = households
ATFAC = area type adjustment factor:

Area type	Factor
1	1.05
2	0.90
6	1.20

Note: no factor is applied to area types 3-5.

6.8 AUXILIARY MODEL AND INPUT TRIP TABLES

These models, which MWCOG staff generally refer to as exogenous travel, miscellaneous travel, or residual travel models,[42] reflect the auxiliary automobile travel in the region that ActivitySim does not account for, such as

- Airport Passengers
- Visitors
- Visitor Taxi Trips
- Auto-Person External-External (EE) Trips
- Trucks and Commercial Vehicles

[41] Allen, *Development of a Model for Commercial Vehicle Trips*, 46.
[42] See, for example, Ray Ngo to DTP Technical Staff et al., "Exogenous Demand Inputs to the TPB Travel Demand Model: Update for Round 9.2 Cooperative Forecasts," Memorandum, June 21, 2021.

- Auto-Person Internal-External (IE), External-Internal (EI) Trips

Figure 9 shows the auxiliary models used in the Gen3 model and how they fit together as components of the final traffic assignment.

The auto-person external-internal trips are trips made by a non-resident of the MWCOG region with one end in the region and one end at an external station. Similarly, the auto-person internal-external trips are trips made by a resident of the MWCOG region with one end in the region and one end at an external station. In both cases, these are routine or semi-routine trips, not visitor trips.

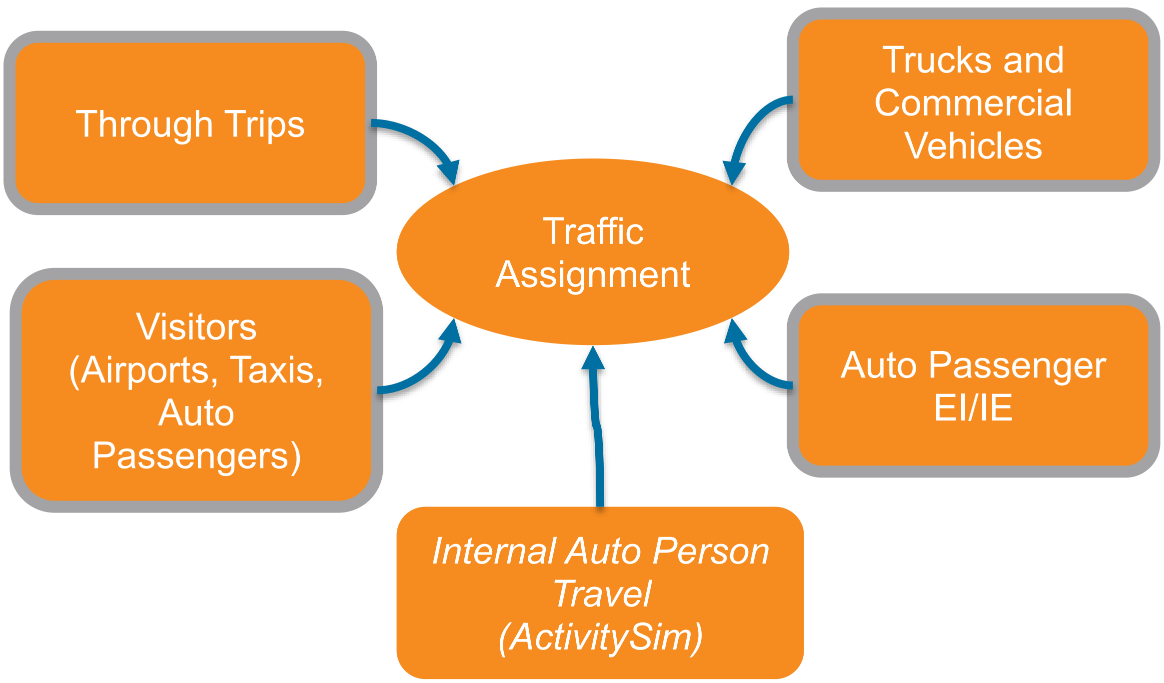


FIGURE 9: TRAFFIC ASSIGNMENT COMPONENTS AND AUXILIARY MODELS

The auxiliary model process used in the Gen3, Phase 1, Model is shown in Figure 9. This process used static trip matrices as inputs for the auxiliary traffic sources. The updated process in the Gen3, Phase 2, Model is shown in Figure 10. In the cases of airport passengers, visitors, and external-external auto trips, existing endogenous models are used to prepare trip tables that are inputs for the regional travel demand model. Truck and commercial vehicles and auto passenger internal-external and external-internal trip tables are created as part of this process. The supplementary school trip table was removed due to the possibility of double counting with trips from ActivitySim.

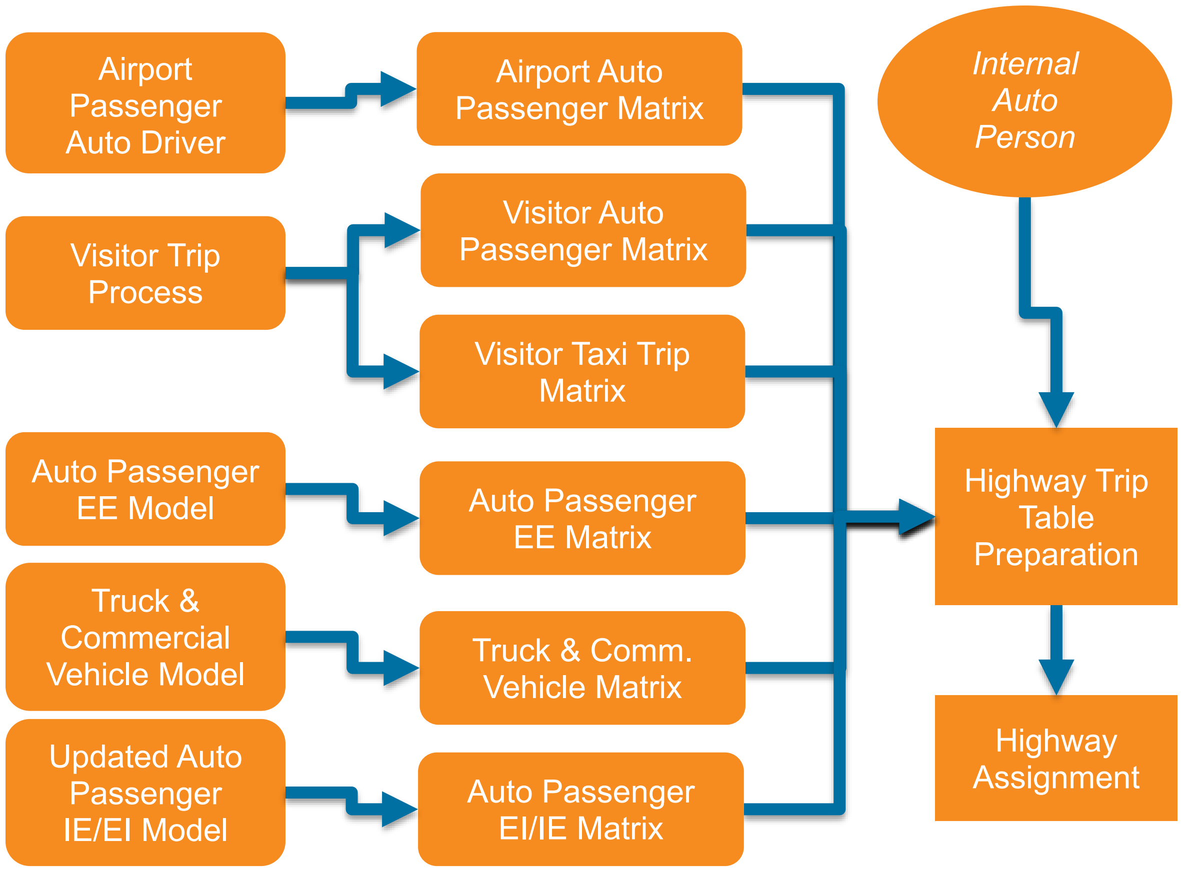


FIGURE 10: AUXILIARY HIGHWAY MODEL PROCESS

Airport Passengers, Visitors, Taxi Trips, and Auto-Person External-External Trips

In the cases of airport passengers, visitors, taxi trips, and auto-person external-external trips, there are input trip tables in the model input folder that are used as inputs in the Generation 2 Model as well. These were prepared by MWCOG staff in an exogenous process. In this case, updating the model for future years would require MWCOG to continue the current process to prepare new matrices for future years. Note that, in the case of the taxi trips, the input trip tables represent special-generator markets that are not well represented by the household travel survey data, and do not significantly overlap with the resident ridehailing travel outputs from ActivitySim (which includes taxi and TNC).

External-Internal (EI) and Internal-External (IE) Auto Trips Model

Big Data (also known as passively collected origin-destination data, such as the AirSage data used in the Gen2 Model) have been used to develop separate EI and IE models. The process is shown in Figure 11.

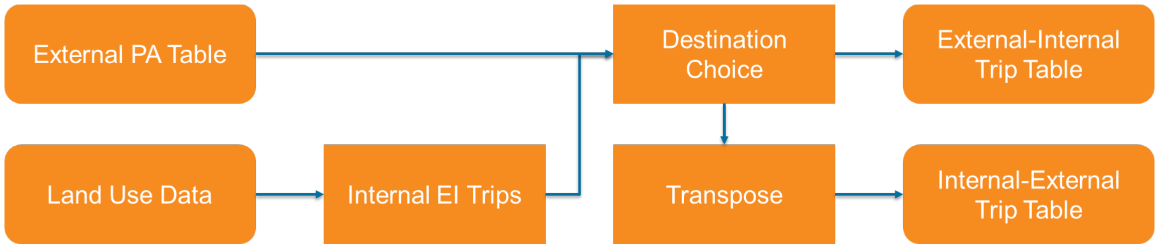


FIGURE 11: AUTO PERSON EXTERNAL-INTERNAL AND INTERNAL-EXTERNAL TRIP FLOWCHART

In this model, internal-external trips are made by residents of the metropolitan Washington region from households to external stations and external-internal trips are made by non-residents (e.g., out-of-region commuters) who travel into the metropolitan Washington region. Inputs to this model included:

- Forecasts of volumes at external stations and a percentage of those trips that are IE at each external station, by trip purpose
- Land Use data
    - For home-based IE trips, total households by TAZ
    - For home-based EI trips, employment by type by TAZ
    - For NHB trips (both directions), total households and employment by type by TAZ
- Drive-alone midday time, distance, and toll cost skims, which are used to calculated a generalized cost matrix: time + y * ((distance * auto operating cost)+tolls) where y is a term that converts cost to minutes

The inputs to the calibration of the auto passenger internal-external and external-internal models were the AirSage data that MWCOG procured from AirSage in 2014, which are shown in Figures 2-7. These maps are based on the AirSage data provided by MWCOG[43] and the external PA model input file from the Generation 2 Model, which has auto person trips by purpose and direction. Trip purposes include Home-based Work (HBW), Home-based Shopping (HBS) and Home-based Other (HBO), and Non-home-based (NHB), and directions are IE and EI, referring to internal-external and external-internal, respectively. In all cases, there is a definite pattern showing higher numbers of external trips in the areas north and west to the TPB's modeling area (i.e., Baltimore and Virginia), which was considered in both the trip generation and attraction functions as well as the destination choice impedance functions. For this short-term approach, the EI model operated with autos only (and may continue as such for the long term), and the IE model used a factor to convert person trips (AirSage) to auto trips (based on external PA traffic counts).

Once the trip generation and attraction models were calibrated, a destination choice model was calibrated to the AirSage EI and IE trip tables. The model forms are shown in Equation 4 and

[43] R. Milone and M. Seifu, Year 2014 AirSage Data Time-of-Day Analysis, Memorandum to DTP Technical Staff, 21 April 2020.

Equation 5. The destination choice utility functions were based on similar factors as the trip generation data (zonal data and generalized cost), and it was adjusted during model calibration.

EQUATION 4: INTERNAL-EXTERNAL DESTINATION CHOICE MODEL

$$IETrips_{ij} = Attractions_i * \frac{\exp(\beta_{time} * time_{ij} + \beta_{cost} * CostPerMile * Distance_{ij})}{\sum \exp(\beta_{time} * time_{ij} + \beta_{cost} * CostPerMile * Distance_{ij})}$$

EQUATION 5: EXTERNAL-INTERNAL DESTINATION CHOICE MODEL

$$EITrips_{ij} = Productions_i * \frac{\exp(\beta_{time} * time_{ij} + \beta_{cost} * CostPerMile * Distance_{ij})}{\sum \exp(\beta_{time} * time_{ij} + \beta_{cost} * CostPerMile * Distance_{ij})}$$

The external-internal models are singly-constrained, meaning that one iteration is run with no attraction balancing since the rate of internal attractions of external day trips to COG is unknown for any given zone. For internal-external models, the models are applied from external to internal and as singly-constrained models to ensure that they matched the external PA counts and the resulting trip table were transposed to prepare them for assignment.

The external-internal and internal-external models were calibrated to trip generation and attraction by zone and trip length frequency compared to the AirSage data. As part of the completion of this model, matching the external PA counts are documented[44].

6.9 EXTERNAL AND VISITOR TRANSIT TRAVEL MODEL

The external transit model is input to the Gen3 Model as static trip tables. These trip tables primarily represent the transit entering the region via MARC commuter rail with a smaller portion using Amtrak Rail. These tables were developed using survey data from the WMATA Metrorail Survey.

The survey data was processed to determine the home location of riders and then determined using GIS if respondents were external users that reported a home location within 75 miles of the model region or visitors, who reported a home location outside of 75 miles and did not transfer from MARC or Amtrak. Additional information can be found in the Gen3 Model Phase 1 Development Report[45].

The trip tables are input from the transit input folder. These tables are unchanged via the model processes, if growth is expected for external and visitor transit the tables would need to be updated via an exogenous process.

---

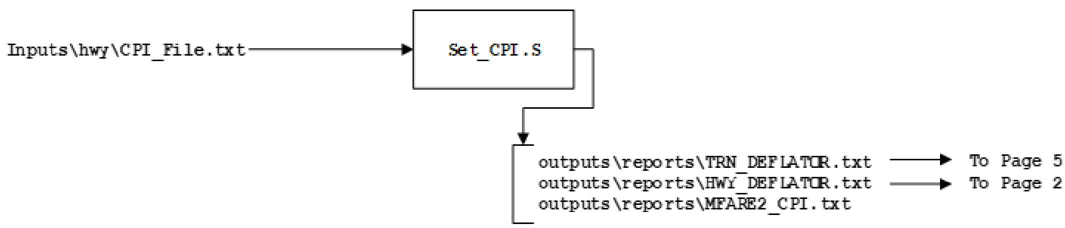
[44] RSG, Inc. Gen3 Auxiliary Travel Model Implementation in the Phase 2 Development. Memorandum. August 18, 2022. https://app.box.com/s/epegylkxr9vvyvay0h3cjbqn53ydypvr.
[45] RSG, Inc. and BMG. Gen3 Data Development. 12/29/21. https://app.box.com/s/xe5vb28daox1aqtw895iy2r5ocy584w8.

7.0 APPENDIX

7.1 DETAILED MODEL FLOWCHARTS

TITLE:	**Flowchart for the TPB activity-based, regional travel demand forecasting model**
COMPANY: **COG/TPB**	Authors COG/TPB Staff
DATE: 09/27/2023	PG: 1 OF 14
FILENAME:	Gen3_Phase_2_flowchart_v09.vsd

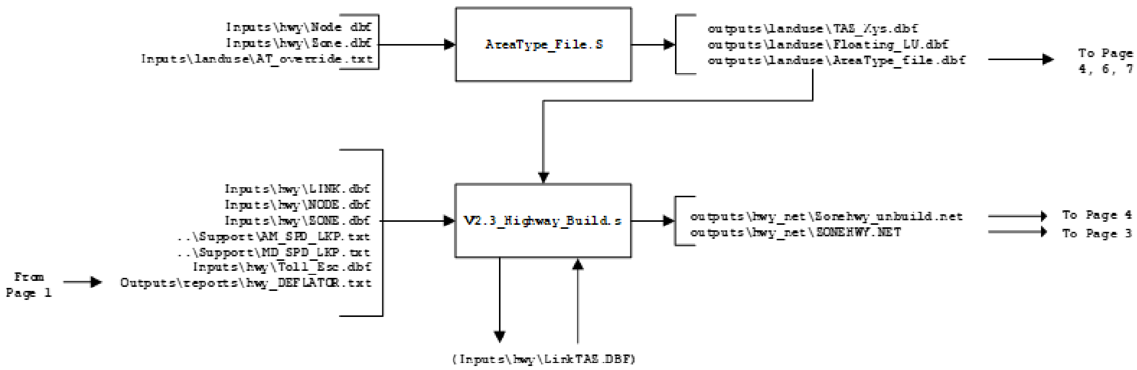
Set_CPI.bat



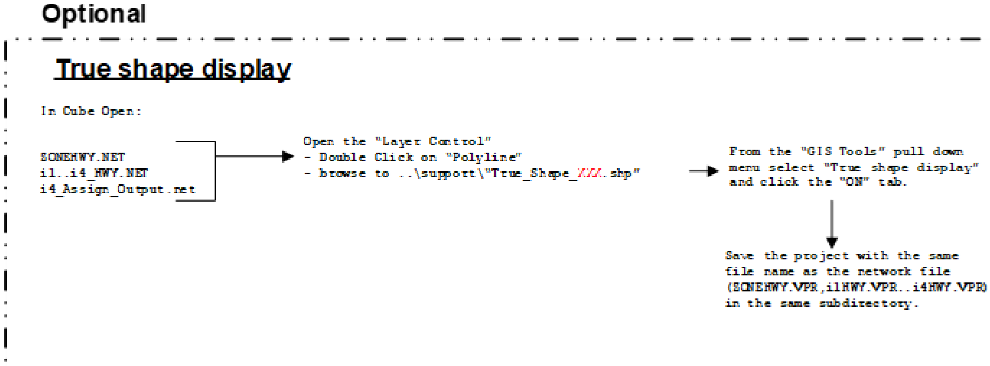
Report Files Generated by Set_CPI.bat:
Set_CPI.rpt
Set_Factors.rpt

TITLE: Flowchart for the TPB activity-based, regional travel demand forecasting model	
COMPANY: COG/TPB	Authors COG/TPB Staff
DATE: 09/27/2023	PG: 2 OF 14
FILENAME: Gen3_Phase_2_flowchart_v09.vsd	

PP_Highway_Build.bat



Optional



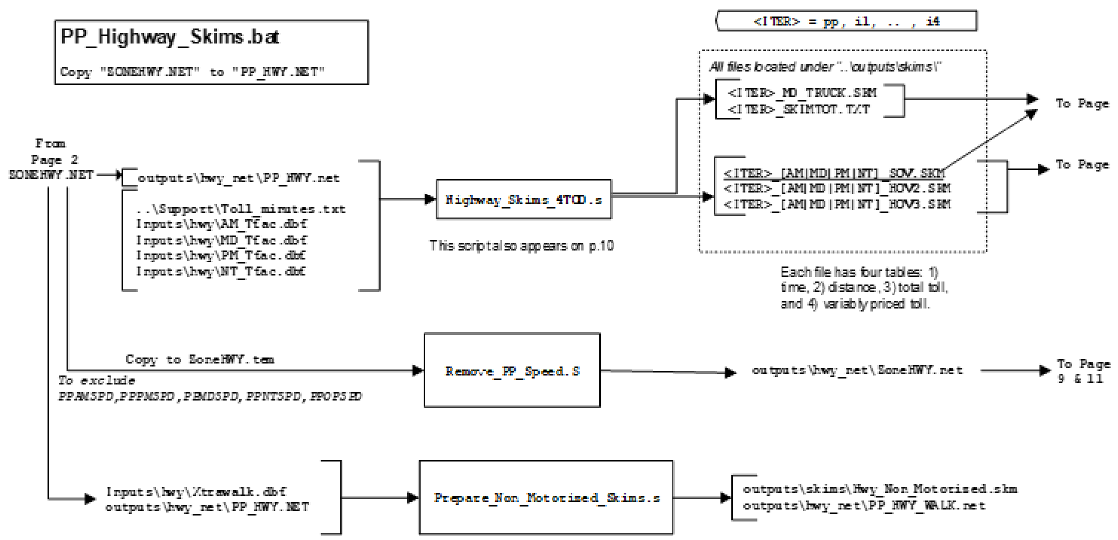
Report Files Generated by PP_Highway_Build.bat:

AreaType_File.rpt
V2.3_highway_build.rpt

TITLE:	**Flowchart for the TPB activity-based, regional travel demand forecasting model**
COMPANY: **COG/TPB**	Authors COG/TPB Staff
DATE: 09/27/2023	PG: 3 OF 14
FILENAME: Gen3_Phase_2_flowchart_v09.vsd	

PP_Highway_Skims.bat

(See also Page 9 for highway skimming process used in speed feedback iterations 1-4)

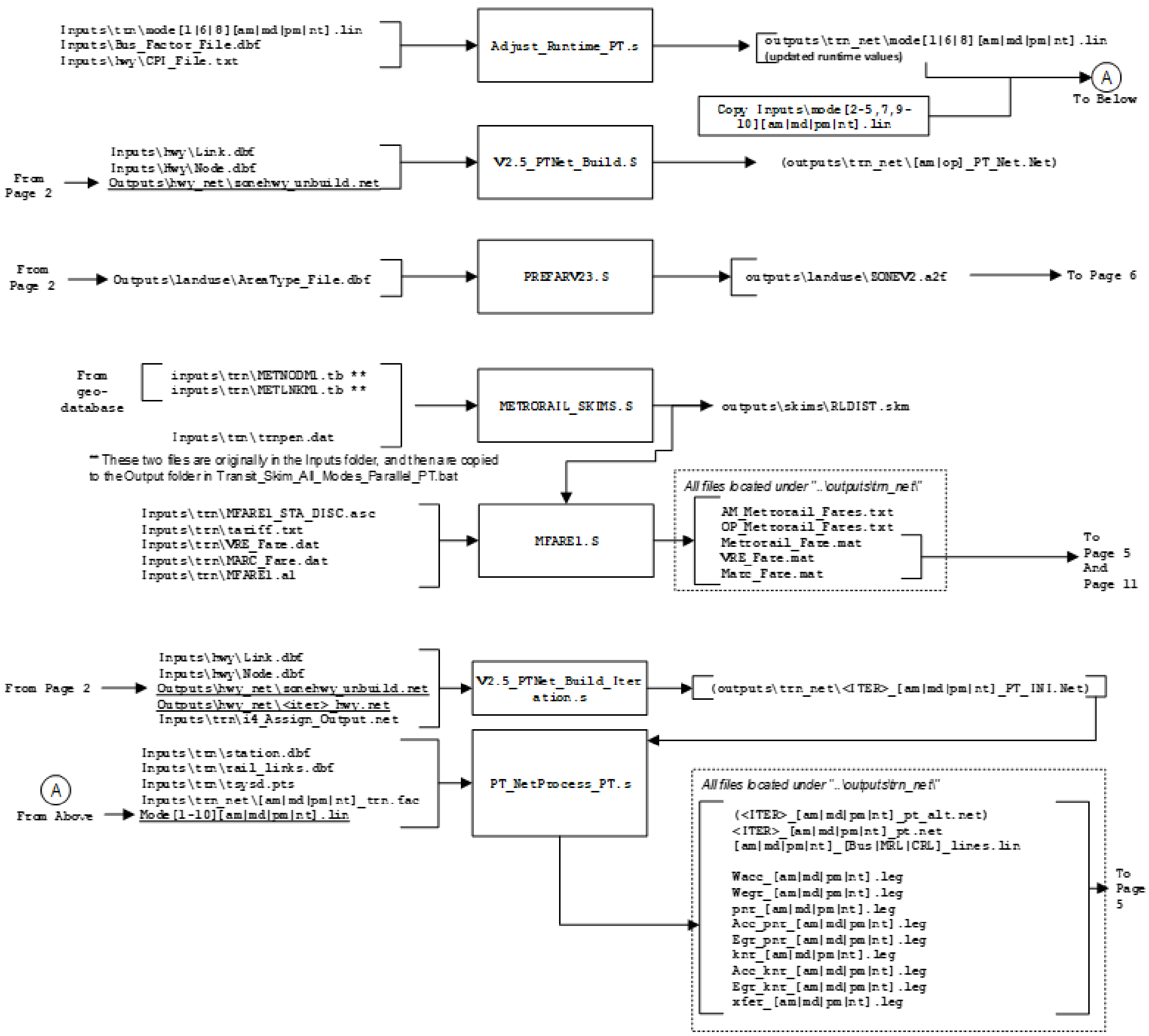


Report Files Generated by PP_Highway_Skims.bat:

pp_Highway_skims_4TOD.rpt
pp_Joinskims.rpt
pp_Modnet.rpt
pp_Highway_skims_mod_4TOD.rpt
CheckStationAccess.rpt
pp_Remove_PP_Speed.rpt
Prepare_non_motorized_skims.rpt

TITLE:	**Flowchart for the TPB activity-based, regional travel demand forecasting model**
COMPANY: **COG/TPB**	Authors COG/TPB Staff
DATE: 09/27/2023	PG: 4 OF 14
FILENAME: Gen3_Phase_2_flowchart_v09.vsd	

Transit_Fare_PT.bat
Transit_Skims_All_Modes_Parallel_PT.bat



Report Files Generated by Transit_Fare_PT.bat and Transit_Skims_All_Modes_Parallel_PT.bat

adjust_Runtime.rpt
v2.5_PTNet_Build.rpt
walkacc.rpt
<iter>_prefarV23.rpt
<iter>_metrorailskims.rpt
<iter>_MFare1.rpt

V2.5_PTNet_Build_Iteration.rpt
PT_netProcess_PT.rpt
<iter>_transit_skims_ab.rpt
<iter>_transit_skims_bm.rpt
<iter>_transit_skims_cr.rpt
<iter>_transit_skims_mr.rpt

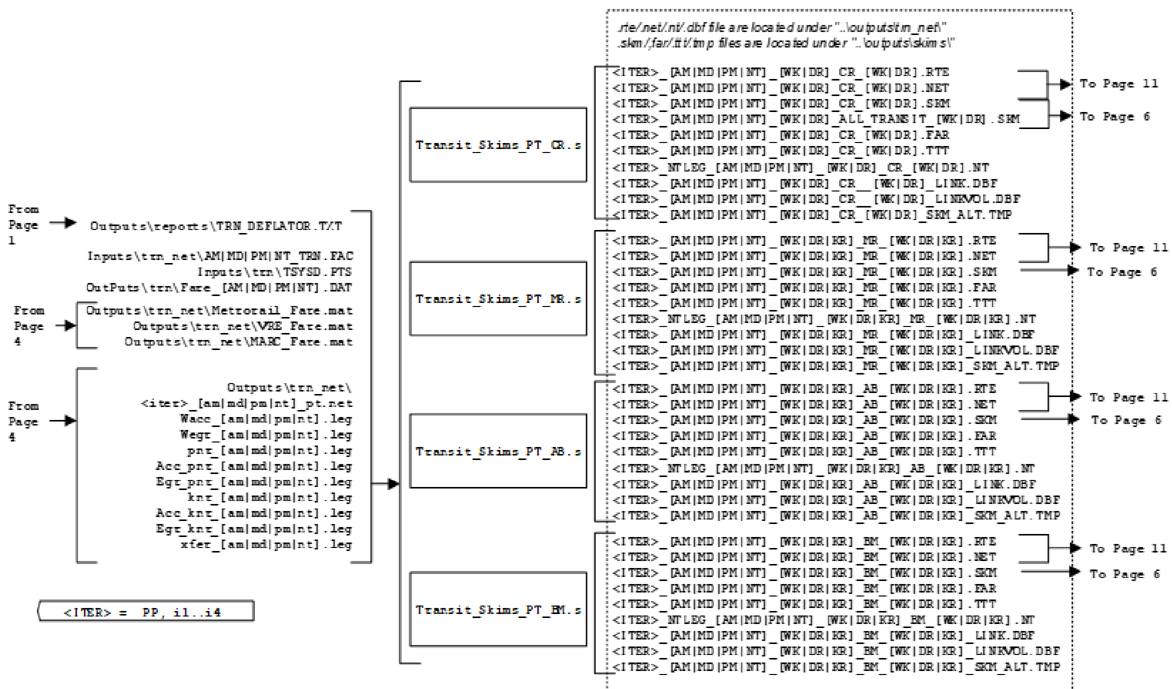
TITLE: Flowchart for the TPB activity-based, regional travel demand forecasting model

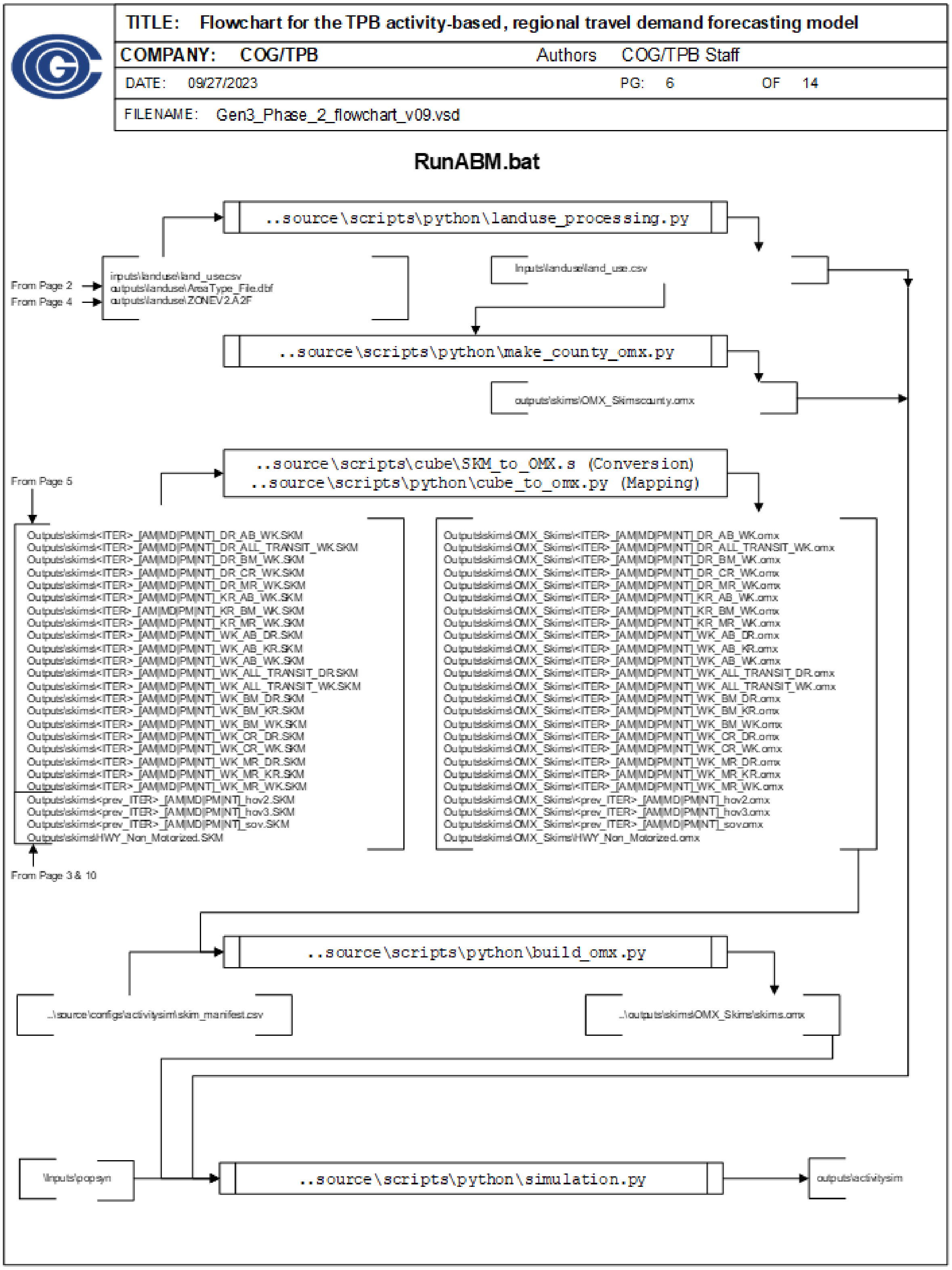
COMPANY: COG/TPB Authors COG/TPB Staff

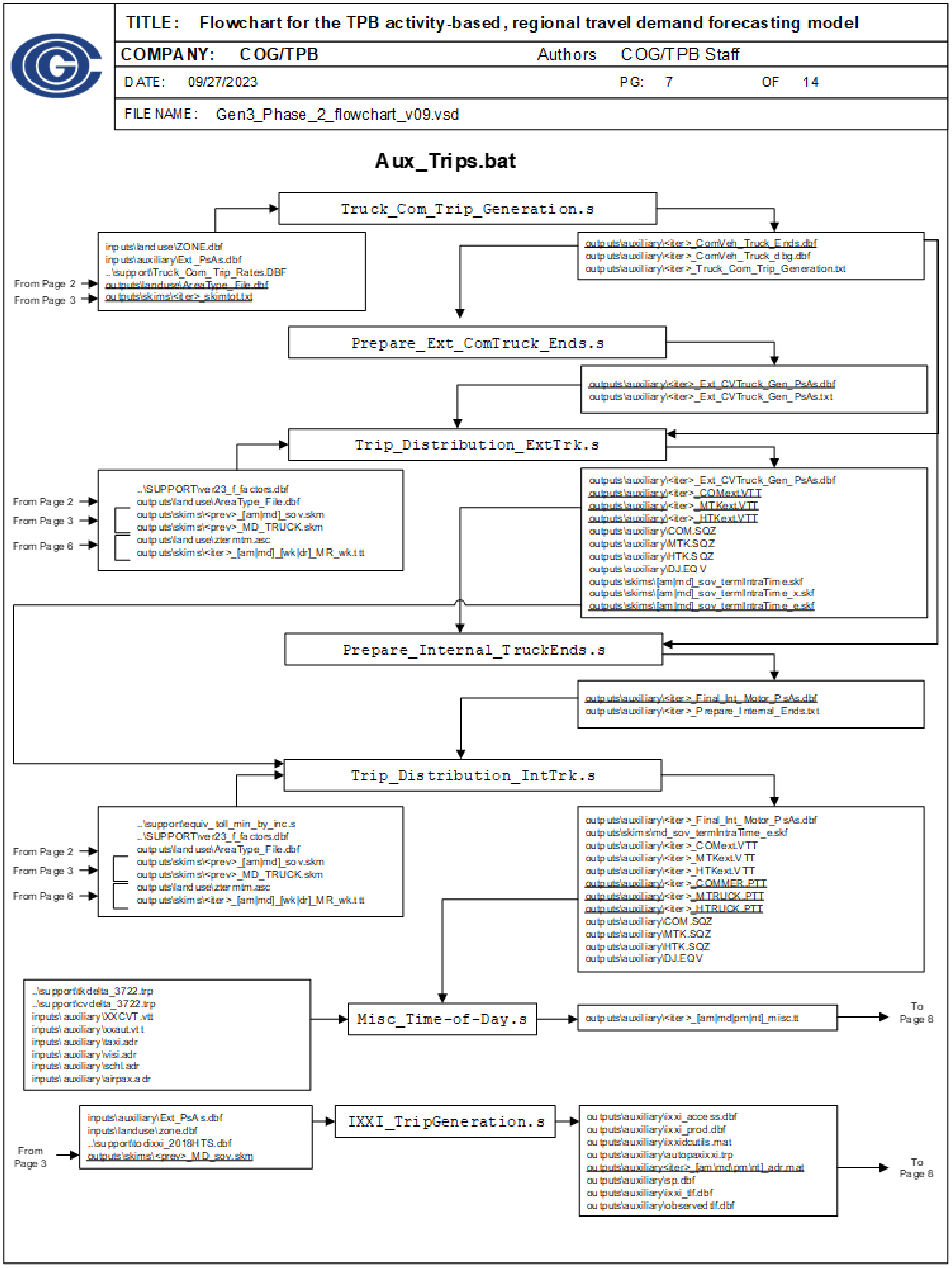
DATE: 09/27/2023 PG: 5 OF 14

FILENAME: Gen3_Phase_2_flowchart_v09.vsd

Transit_Skim_LineHaul_Parallel_PT.bat

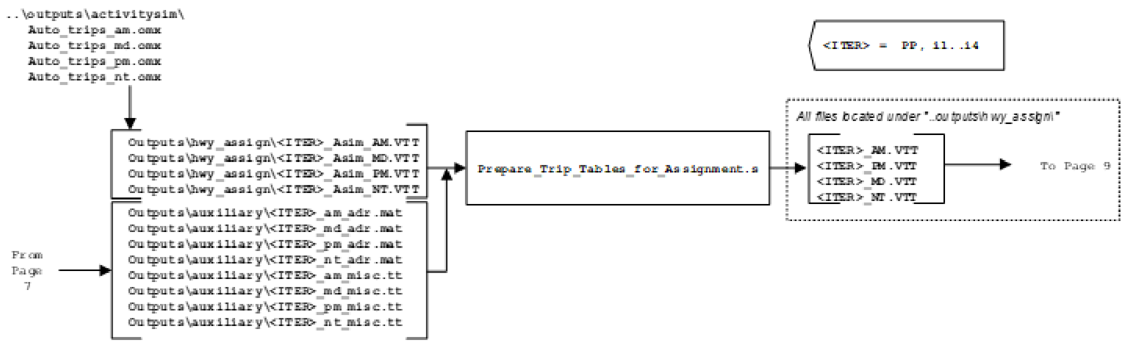



TITLE: Flowchart for the TPB activity-based, regional travel demand forecasting model
COMPANY: COG/TPB
Authors COG/TPB Staff
DATE: 09/27/2023
PG: 6
OF 14
FILENAME: Gen3_Phase_2_flowchart_v09.vsd
RunABM.bat
..source\scripts\python\landuse_processing.py
From Page 2
From Page 4
inputs\landuse\land_use.csv
outputs\landuse\AreaType_File.dbf
outputs\landuse\ZONEV2.A2F
Inputs\landuse\land_use.csv
..source\scripts\python\make_county_omx.py
outputs\skims\OMX_Skimscounty.omx
..source\scripts\cube\SKM_to_OMX.s (Conversion)
..source\scripts\python\cube_to_omx.py (Mapping)
From Page 5
Outputs\skims\<ITER>_[AM|MD|PM|NT]_DR_AB_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_DR_ALL_TRANSIT_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_DR_BM_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_DR_CR_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_DR_MR_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_KR_AB_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_KR_BM_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_KR_MR_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_AB_DR.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_AB_KR.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_AB_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_ALL_TRANSIT_DR.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_ALL_TRANSIT_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_BM_DR.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_BM_KR.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_BM_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_CR_DR.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_CR_WK.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_MR_DR.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_MR_KR.SKM
Outputs\skims\<ITER>_[AM|MD|PM|NT]_WK_MR_WK.SKM
Outputs\skims\<prev_ITER>_[AM|MD|PM|NT]_hov2.SKM
Outputs\skims\<prev_ITER>_[AM|MD|PM|NT]_hov3.SKM
Outputs\skims\<prev_ITER>_[AM|MD|PM|NT]_sov.SKM
Outputs\skims\HWY_Non_Motorized.SKM
From Page 3 & 10
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_DR_AB_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_DR_ALL_TRANSIT_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_DR_BM_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_DR_CR_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_DR_MR_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_KR_AB_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_KR_BM_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_KR_MR_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_AB_DR.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_AB_KR.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_AB_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_ALL_TRANSIT_DR.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_ALL_TRANSIT_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_BM_DR.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_BM_KR.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_BM_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_CR_DR.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_CR_WK.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_MR_DR.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_MR_KR.omx
Outputs\skims\OMX_Skims\<ITER>_[AM|MD|PM|NT]_WK_MR_WK.omx
Outputs\skims\OMX_Skims\<prev_ITER>_[AM|MD|PM|NT]_hov2.omx
Outputs\skims\OMX_Skims\<prev_ITER>_[AM|MD|PM|NT]_hov3.omx
Outputs\skims\OMX_Skims\<prev_ITER>_[AM|MD|PM|NT]_sov.omx
Outputs\skims\OMX_Skims\HWY_Non_Motorized.omx
..source\scripts\python\build_omx.py
..\source\configs\activitysim\skim_manifest.csv
..\outputs\skims\OMX_Skims\skims.omx
\inputs\popsyn
..source\scripts\python\simulation.py
outputs\activitysim


TITLE: Flowchart for the TPB activity-based, regional travel demand forecasting model
COMPANY: COG/TPB
Authors COG/TPB Staff
DATE: 09/27/2023
PG: 7 OF 14
FILE NAME: Gen3_Phase_2_flowchart_v09.vsd
Aux_Trips.bat
Truck_Com_Trip_Generation.s
inputs\landuse\ZONE.dbf
inputs\auxiliary\Ext_PsAs.dbf
..\support\Truck_Com_Trip_Rates.DBF
outputs\landuse\AreaType_File.dbf
outputs\skims\<iter>_skimtot.txt
From Page 2
From Page 3
outputs\auxiliary\<iter>_ComVeh_Truck_Ends.dbf
outputs\auxiliary\<iter>_ComVeh_Truck_dbg.dbf
outputs\auxiliary\<iter>_Truck_Com_Trip_Generation.txt
Prepare_Ext_ComTruck_Ends.s
outputs\auxiliary\<iter>_Ext_CVTruck_Gen_PsAs.dbf
outputs\auxiliary\<iter>_Ext_CVTruck_Gen_PsAs.txt
Trip_Distribution_ExtTrk.s
..\SUPPORT\ver23_f_factors.dbf
outputs\landuse\AreaType_File.dbf
outputs\skims\<prev>_[am|md]_sov.skm
outputs\skims\<prev>_MD_TRUCK.skm
outputs\landuse\ztermtm.asc
outputs\skims\<iter>_[am|md]_[wk|dr]_MR_wk.ttt
From Page 2
From Page 3
From Page 6
outputs\auxiliary\<iter>_Ext_CVTruck_Gen_PsAs.dbf
outputs\auxiliary\<iter>_COMext.VTT
outputs\auxiliary\<iter>_MTKext.VTT
outputs\auxiliary\<iter>_HTKext.VTT
outputs\auxiliary\COM.SQZ
outputs\auxiliary\MTK.SQZ
outputs\auxiliary\HTK.SQZ
outputs\auxiliary\DJ.EQV
outputs\skims\[am|md]_sov_termIntraTime.skf
outputs\skims\[am|md]_sov_termIntraTime_x.skf
outputs\skims\[am|md]_sov_termIntraTime_e.skf
Prepare_Internal_TruckEnds.s
outputs\auxiliary\<iter>_Final_Int_Motor_PsAs.dbf
outputs\auxiliary\<iter>_Prepare_Internal_Ends.txt
Trip_Distribution_IntTrk.s
..\support\equiv_toll_min_by_inc.s
..\SUPPORT\ver23_f_factors.dbf
outputs\landuse\AreaType_File.dbf
outputs\skims\<prev>_[am|md]_sov.skm
outputs\skims\<prev>_MD_TRUCK.skm
outputs\landuse\ztermtm.asc
outputs\skims\<iter>_[am|md]_[wk|dr]_MR_wk.ttt
From Page 2
From Page 3
From Page 6
outputs\auxiliary\<iter>_Final_Int_Motor_PsAs.dbf
outputs\skims\md_sov_termIntraTime_e.skf
outputs\auxiliary\<iter>_COMext.VTT
outputs\auxiliary\<iter>_MTKext.VTT
outputs\auxiliary\<iter>_HTKext.VTT
outputs\auxiliary\<iter>_COMMER.PTT
outputs\auxiliary\<iter>_MTRUCK.PTT
outputs\auxiliary\<iter>_HTRUCK.PTT
outputs\auxiliary\COM.SQZ
outputs\auxiliary\MTK.SQZ
outputs\auxiliary\HTK.SQZ
outputs\auxiliary\DJ.EQV
..\support\tkdelta_3722.trp
..\support\cvdelta_3722.trp
inputs\auxiliary\XXCVT.vtt
inputs\auxiliary\xxaut.vtt
inputs\auxiliary\taxi.adr
inputs\auxiliary\visi.adr
inputs\auxiliary\schl.adr
inputs\auxiliary\airpax.adr
Misc_Time-of-Day.s
outputs\auxiliary\<iter>_[am|md|pm|nt]_misc.tt
To Page 8
inputs\auxiliary\Ext_PsAs.dbf
inputs\landuse\zone.dbf
..\support\todixxi_2018HTS.dbf
outputs\skims\<prev>_MD_sov.skm
From Page 3
IXXI_TripGeneration.s
outputs\auxiliary\ixxi_access.dbf
outputs\auxiliary\ixxi_prod.dbf
outputs\auxiliary\ixxidcutils.mat
outputs\auxiliary\autopaxixxi.trp
outputs\auxiliary\<iter>_[am|md|pm|nt]_adr.mat
outputs\auxiliary\sp.dbf
outputs\auxiliary\ixxi_tlf.dbf
outputs\auxiliary\observed tlf.dbf
To Page 8

TITLE:	**Flowchart for the TPB activity-based, regional travel demand forecasting model**		
COMPANY:	**COG/TPB**	Authors	COG/TPB Staff
DATE:	09/27/2023	PG: 8	OF 14
FILE NAME:	Gen3_Phase_2_flowchart_v09.vsd		

Highway_Assignment_Prep.bat

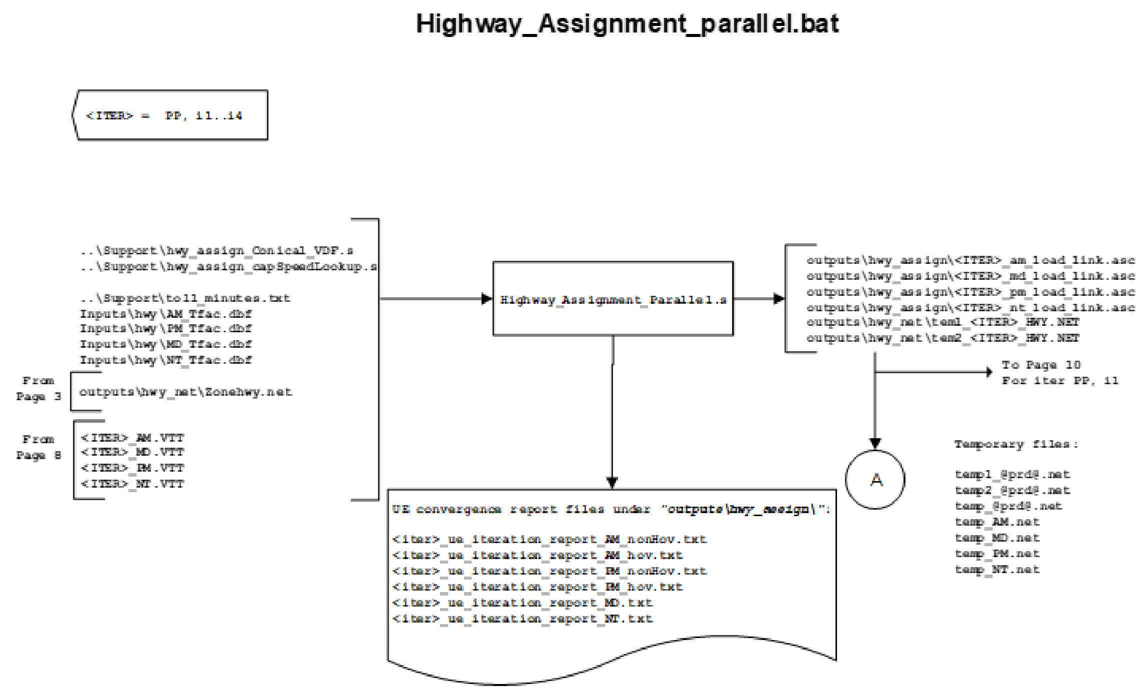


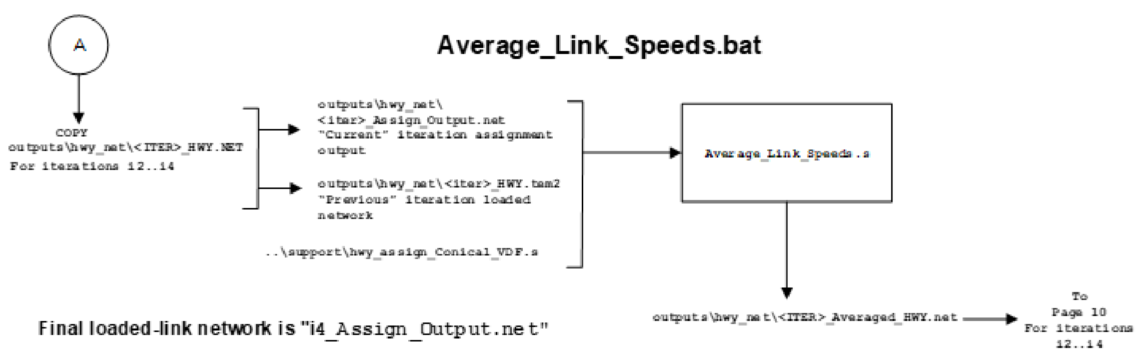
TITLE: Flowchart for the TPB activity-based, regional travel demand forecasting model

COMPANY: COG/TPB   Authors COG/TPB Staff

DATE: 09/27/2023   PG: 9 OF 14

FILENAME: Gen3_Phase_2_flowchart_v09.vsd



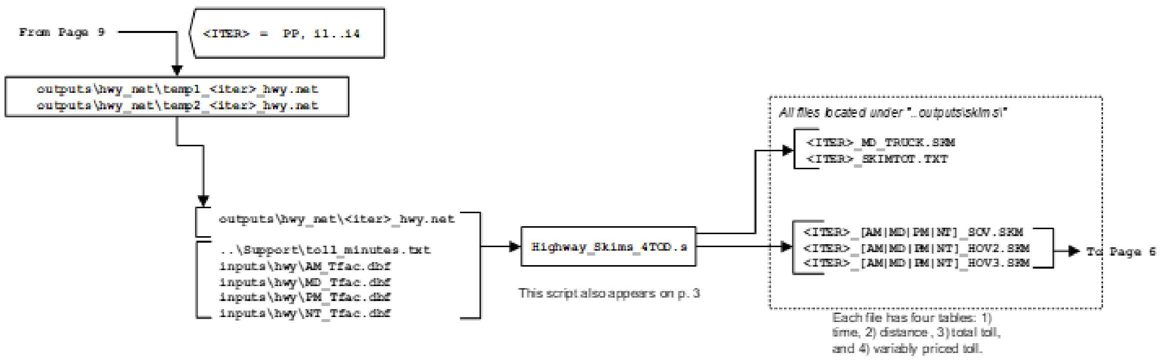


Report Files Generated by Highway_Assignment.bat:
<ITER>_Highway_assignment.rpt
Average_Link_Speeds.rpt

TITLE: Flowchart for the TPB activity-based, regional travel demand forecasting model	
COMPANY: COG/TPB	Authors COG/TPB Staff
DATE: 09/27/2023	PG: 10 OF 14
FILE NAME: Gen3_Phase_2_flowchart_v09.vsd	

Highway_Skims.bat

(See also page 3 for highway skimming process used in the "pump prime" speed feedback iteration)



Report Files Generated by Highway_Skims.bat:

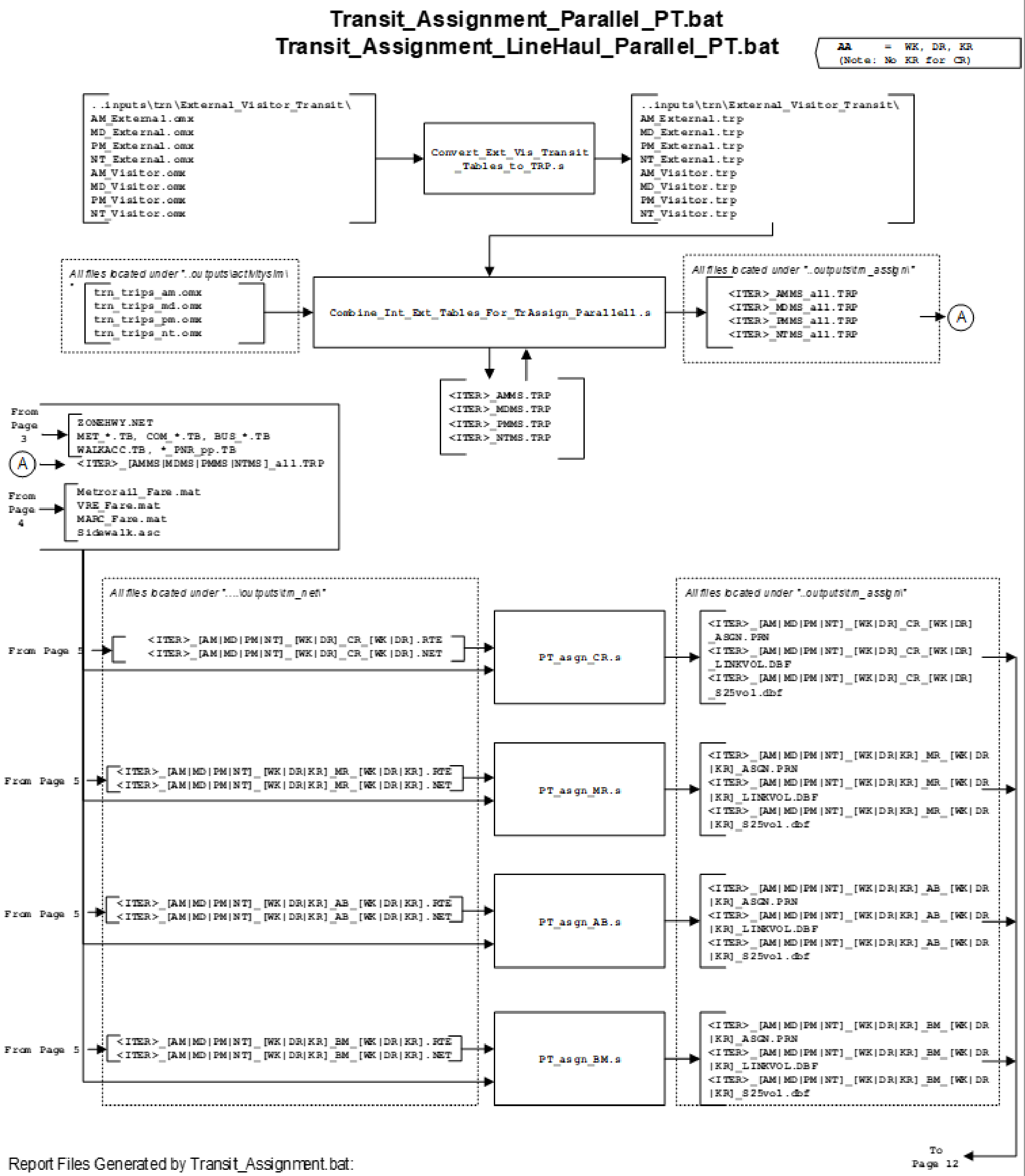
<ITER>_Highway_skims.rpt
<ITER>_Joinskims.rpt
<ITER>_Modnet.rpt
<ITER>_Highway_skims_mod.rpt

TITLE: Flowchart for the TPB activity-based, regional travel demand forecasting model

COMPANY: COG/TPB Authors COG/TPB Staff

DATE: 09/27/2023 PG: 11 OF 14

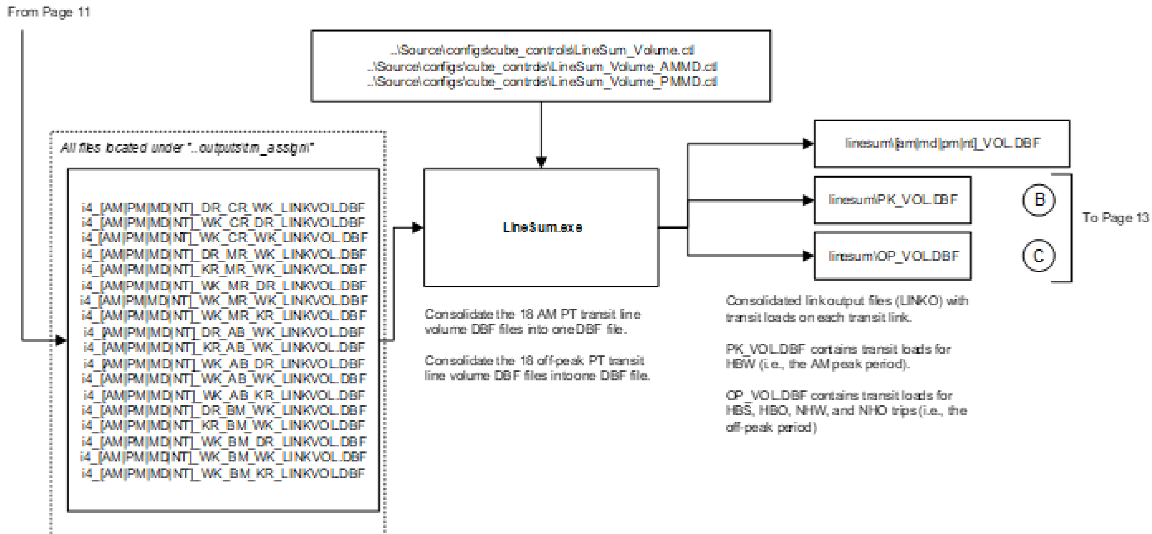
FILENAME: Gen3_Phase_2_flowchart_v09.vsd



Report Files Generated by Transit_Assignment.bat:
Combine_Tables_For_TrAssign_Parallel.rpt
Transit_Assignment.rpt, Transit_Assgn_CR.rpt, Transit_Assgn_MR.rpt, Transit_Assgn_AB.rpt, Transit_Assgn_BM.rpt

TITLE:	Flowchart for the TPB activity-based, regional travel demand forecasting model		
COMPANY:	COG/TPB	Authors	COG/TPB Staff
DATE:	09/27/2023	PG: 12	OF 14
FILENAME:	Gen3_Phase_2_flowchart_v09.vsd		

Transum.bat (page 1 of 2)



Link output files (LINKO) and node output files (NODEO) are produced in the transit assignment.

NODEO files include the X,Y coordinates.

LINKO files include:

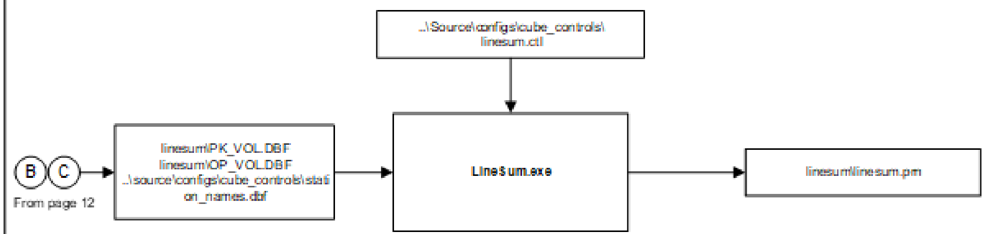
A — A-node of link
B — B-node of link
TIME — A-B time (hundredths of minutes)
MODE — Mode of link (1-255)
COLOR — User designated drawing color
STOP_A — 1 = A is a stop node
STOP_B — 1 = B is a stop node
DIST — A-B distance
NAME — Name of line on this link
FREQ — Service frequency (min)
PLOT — Always = 0

The following are included due to transit assignment
SEQ — Link sequence in the line
OWNER — Line owner (1st ten characters)
AB_VOL — Volume
AB_BRDA — Number of trip boardings a A
AB_XITA — Number of exits at A
AB_BRDB — Number of boardings at B
AB_XITB — Number of exits at B
(last 5 variables are also repeated for B-A direction)

See p. 1020 of Citilabs, Inc., "Cube Voyager Reference Guide, Version 6.4.1." September 30, 2015.

TITLE:	Flowchart for the TPB activity-based, regional travel demand forecasting model
COMPANY: COG/TPB	Authors COG/TPB Staff
DATE: 09/27/2023	PG: 13 OF 14
FILE NAME:	Gen3_Phase_2_flowchart_v09.vsd

Transum.bat (page 2 of 2)



station_names.dbf is an equivalency between node/station number and the node name/station name. It is created from station.dbf using set_factors.s (p. 2).

LINESUM summarizes transit line volume data stored in a PT loaded link DBF file. It can be used to create the following summaries:

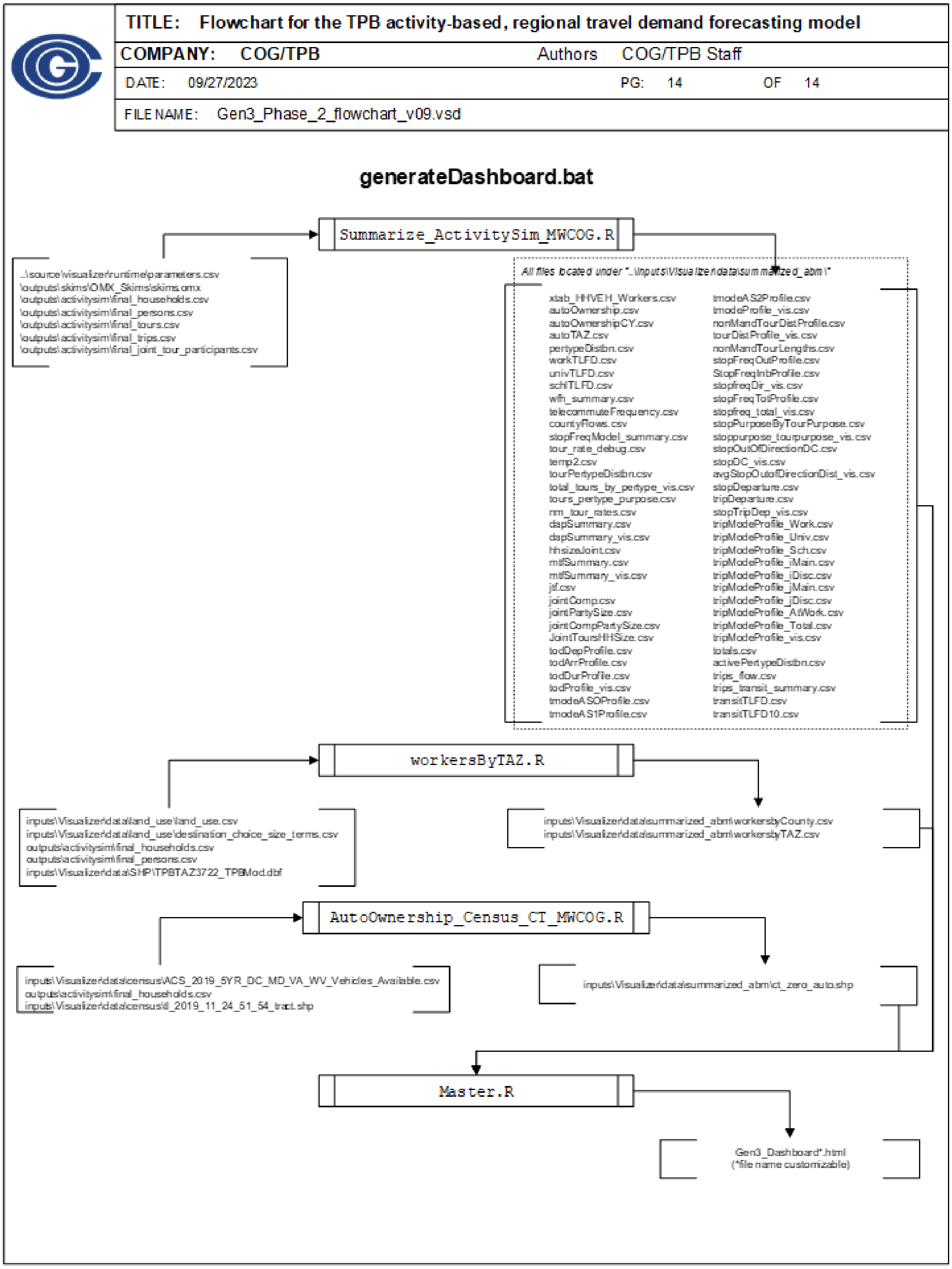
* Boarding/alighting information
* Station access information
* Link-based summaries
  (i.e., between stations).
* Transit route/line summaries

In this case, we are using the "ACCESS" keyword, so the program summarizes the number of riders accessing a station.

Transit assignment is in P/A format.

Number of passengers accessing one or more stations, i.e., number of passengers boarding and alighting each station. This can be disaggregated by access mode, but, in this case, it is not.

	---- Peak ----		-- Offpeak ---		---- Daily ---		
Stop	Arrive	Depart	Arrive	Depart	Arrive	Depart	
8001	23143	1508	1610	621	24753	2129	Shady Grove
8002	10395	2022	1026	718	11421	2740	Rockville
8003	3934	2983	770	1394	4704	4377	Twinbrook
8004	2931	2976	615	1413	3546	4389	White Flint
8005	8117	197	1979	716	10096	913	Grosvenor
8006	4491	7324	955	1805	5446	9129	Medical Center
8007	11213	12786	3916	5519	15129	18305	Bethesda
8008	9492	5630	3230	3340	12722	8970	Friendship Heights
8009	7904	3640	2780	1856	10684	5496	Tenleytown
8010	4845	2721	2041	1610	6886	4331	Van Ness-UDC


TITLE: Flowchart for the TPB activity-based, regional travel demand forecasting model
COMPANY: COG/TPB
Authors COG/TPB Staff
DATE: 09/27/2023
PG: 14 OF 14
FILE NAME: Gen3_Phase_2_flowchart_v09.vsd
generateDashboard.bat
Summarize_ActivitySim_MWCOG.R
..\source\visualizer\runtime\parameters.csv
\outputs\skims\OMX_Skims\skims.omx
\outputs\activitysim\final_households.csv
\outputs\activitysim\final_persons.csv
\outputs\activitysim\final_tours.csv
\outputs\activitysim\final_trips.csv
\outputs\activitysim\final_joint_tour_participants.csv
All files located under "..\inputs\Visualizer\data\summarized_abm\"
xtab_HHVEH_Workers.csv
autoOwnership.csv
autoOwnershipCY.csv
autoTAZ.csv
pertypeDistbn.csv
workTLFD.csv
univTLFD.csv
schlTLFD.csv
wfh_summary.csv
telecommuteFrequency.csv
countyFlows.csv
stopFreqModel_summary.csv
tour_rate_debug.csv
temp2.csv
tourPertypeDistbn.csv
total_tours_by_pertype_vis.csv
tours_pertype_purpose.csv
nm_tour_rates.csv
dapSummary.csv
dapSummary_vis.csv
hhsizeJoint.csv
mtfSummary.csv
mtfSummary_vis.csv
jtf.csv
jointComp.csv
jointPartySize.csv
jointCompPartySize.csv
JointToursHHSize.csv
todDepProfile.csv
todArrProfile.csv
todDurProfile.csv
todProfile_vis.csv
tmodeAS0Profile.csv
tmodeAS1Profile.csv
tmodeAS2Profile.csv
tmodeProfile_vis.csv
nonMandTourDistProfile.csv
tourDistProfile_vis.csv
nonMandTourLengths.csv
stopFreqOutProfile.csv
StopFreqInbProfile.csv
stopfreqDir_vis.csv
stopFreqTotProfile.csv
stopfreq_total_vis.csv
stopPurposeByTourPurpose.csv
stoppurpose_tourpurpose_vis.csv
stopOutOfDirectionDC.csv
stopDC_vis.csv
avgStopOutofDirectionDist_vis.csv
stopDeparture.csv
tripDeparture.csv
stopTripDep_vis.csv
tripModeProfile_Work.csv
tripModeProfile_Univ.csv
tripModeProfile_Sch.csv
tripModeProfile_iMain.csv
tripModeProfile_iDisc.csv
tripModeProfile_jMain.csv
tripModeProfile_jDisc.csv
tripModeProfile_AtWork.csv
tripModeProfile_Total.csv
tripModeProfile_vis.csv
totals.csv
activePertypeDistbn.csv
trips_flow.csv
trips_transit_summary.csv
transitTLFD.csv
transitTLFD10.csv
workersByTAZ.R
inputs\Visualizer\data\land_use\land_use.csv
inputs\Visualizer\data\land_use\destination_choice_size_terms.csv
outputs\activitysim\final_households.csv
outputs\activitysim\final_persons.csv
inputs\Visualizer\data\SHP\TPBTAZ3722_TPBMod.dbf
inputs\Visualizer\data\summarized_abm\workersbyCounty.csv
inputs\Visualizer\data\summarized_abm\workersbyTAZ.csv
AutoOwnership_Census_CT_MWCOG.R
inputs\Visualizer\data\census\ACS_2019_5YR_DC_MD_VA_WV_Vehicles_Available.csv
outputs\activitysim\final_households.csv
inputs\Visualizer\data\census\tl_2019_11_24_51_54_tract.shp
inputs\Visualizer\data\summarized_abm\ct_zero_auto.shp
Master.R
Gen3_Dashboard*.html
(*file name customizable)

7.2 USING THE GEN3 MODEL FOR AV OWNERSHIP

This section is a supplement to section 5.4 and is aimed at adjusting the model for automated vehicle scenario studies. There are two options in this section - one is for simple studies, and the other is for more detailed studies. A simple study is where there is a regional target percentage of AV-owning households. A more detailed study is where there are detailed targets - whether by income, location, or any other variable used in ActivitySim. In both cases, there is a group of mode choice and vehicle selection updates that should be adjusted.

Simple Studies

A simple study can be undertaken by changing two files, av_ownership.yaml file and av_ownership_coeffs.csv.

In av_ownership.yaml, set `AV_OWNERSHIP_TARGET_PERCENT` to a the target percentage of households in the region that should own AVs. Then, set the `AV_OWNERSHIP_TARGET_PERCENT_TOLERANCE` to an allowable tolerance. For example, if the target is 10% and the tolerance is 1%, the model will iterate until the regional share is between 9% and 11%. The number of iterations is controlled by `AV_OWNERSHIP_ITERATIONS`. Finally, check in av_ownership_coeffs.csv to ensure that the coefficient named in the YAML file as `AV_OWNERSHIP_COEFFICIENT_CONSTANT` is *not* set to -999 (if it is, 0 is an appropriate value to start with).

Complex Studies

For a complex study, the individual constants will be adjusted. It is expected that the adjustments will happen in outside software, so this section is an outline of how things should be updated in the ActivitySim model configuration files. At least three of the four AV ownership model files will need to be updated for a complex study.

The first update is av_ownership.yaml. In this file, `AV_OWNERSHIP_TARGET_PERCENT`, `AV_OWNERSHIP_TARGET_PERCENT_TOLERANCE`, `AV_OWNERSHIP_ITERATIONS`, and `AV_OWNERSHIP_COEFFICIENT_CONSTANT` will all need to be commented out (using a hashtag). This will keep ActivitySim from attempting to match a regional target.

The second update is in av_ownership.csv, the model spec. This will define how the segmentation is defined. This file has four fields, an identifier, a description, an expression, and a constant to add to the utility for being an AV-owning household, and a 0 for the utility of a household not owning an AV. The expression must result in the value of 1 or True and needs to follow the expression format used in Pandas. An example is below, which uses income segments (setup in annotate_households.csv) to determine if a constant is applied to a household. In the case of multiple items, expressions must be in parentheses and joined with either an ampersand (&) for ‘and’ or a pipe (|) for ‘or’. In the case of complex expressions, they can be added to av_ownership_preprocessor.csv (see section 5.4 for more details).

```
util_inc1,Utility for income group 1,income_segment == 1,coef_av_inc1,0
util_inc2,Utility for income group 2,income_segment == 2,coef_av_inc2,0
```

```
util_inc3,Utility for income group 3,income_segment == 3,coef_av_inc3,0
util_inc4,Utility for income group 4,income_segment == 4,coef_av_inc4,0
```

The third change is the coefficients file. The coefficients listed in the model specification will need to be added to the file. This file is formatted with the coefficient name (which must match the model specification), the value, and an F or T for if the coefficient should NOT be adjusted[46]. An example is below.

```
coef_av_inc1,0.8779621242338954,F
coef_av_inc2,-0.13572485598037048,F
coef_av_inc3,-0.06581044921537069,F
coef_av_inc4,3.099819356052244,F
```

AV Mode Choice Updates

The tour and trip mode choice models will reflect a reduced burden on the in-vehicle time, parking cost, and terminal time for all motorized modes (single-occupant vehicle, 2-person shared-ride, and 3 or more person shared-ride). These adjustments are listed in Table 3938. Additionally, if an AV is available for the tour and trip, the minimum age to drive is reduced, using a default of 13 years old. All four of these adjustments are included in the Gen3_Model\source\configs\activitysim\configs\constants.yaml ActivitySim configuration file.

TABLE 39: AV MODE CHOICE ADJUSTMENT DEFAULTS

ITEM	CONSTANT NAME (USED IN CONSTANTS.YAML)	ADJUSTMENT DEFAULT
In-Vehicle Time Multiplier	autoIVTFactorAV	0.75
Parking Cost Multiplier	autoParkingCostFactorAV	0.5
Terminal Time Multiplier	autoTerminalTimeFactorAV	0.65
Minimum Driving Age (years old)	minAgeDriveAloneAV	13

ActivitySim 1.2.1 is required for this model to operate correctly. The main benefit of this update is that ActivitySim 1.2.1 allows users to relax the requirement that vehicle type data exist for every possible alternative, including those that may not exist in the future[47].

[46] This is observed by Larch for estimating models. Otherwise it must be built in to scripts that are used to adjust constants.

[47] See https://github.com/ActivitySim/activitysim/issues/587 for a detailed explanation.

Vehicle Type Data

The vehicle type data included in the model includes some basic AV assumptions. In the case that the assumptions need to be updated, a Jupyter Notebook is included in the Gen3 Model to assist in creating vehicle type data at Gen3_Model\source\scripts\python\construct_veh_type_data.ipynb. This notebook allows users to set up a new vehicle allocation file for future years reflecting AVs. This notebook uses information setup in Gen3_Model\source\configs\activitysim\configs\av_extrapolation.yaml to determine the variables, such as body types and fuel types, to extrapolate. For AV body types, the body type names must end with "-AV" (case sensitive). In addition, the NumMakes, NumModels, MPG, Range, NewPrice, and auto_operating_cost must include sections for AV body types. For example, if only Hybrid (PEV) and Battery-only (BEV) fuel types will be available for only car body types that can be AVs, the body_types section needs to be updated to include Car-AV, and NumMakes, NumModels, MPG, Range, NewPrice, and auto_operating_cost need to be updated to include entries for Car-AV_PEV and Car-AV_BEV. Some suggested values are included in av_extrapolation.yaml, but they should be reviewed prior to use, particularly as AV adoption increases. The av_extrapolation.yaml file included on the MWCOG's Gen3 Model GitHub repository includes AV body types for car, van, SUV, and pickup truck.

The av_extrapolation.yaml file includes setup variables for the minimum fit year (min_fit_year, the first year to output, currently set to 1998) and the maximum year to extrapolate out to (max_extension, currently set to 2050). The next three array variables are for the variables to extrapolate (variables_to_extrapolate), which are set to include the number of makes, number of models, fuel efficiency in miles per gallon, range, new price, and auto operating cost (indicated as NumMakes, NumModels, MPG, Range, NewPrice, and auto_operating_cost, respectively). The next array is body types (body_types), and lists the body types that will be used in the extrapolation. When using AVs, the body type MUST end in '-AV' (case sensitive). By default, this array includes Car, Car-AV, Van, Van-AV, SUV, SUV-AV, Pickup, Pickup-AV, and Motorcycle. The final array is for fuel types (fuel_types), and it includes gas, diesel, hybrid, PEV, and BEV. In the current implementation (as of 6/28/2023), hybrid refers to non-plug-in hybrid, PEV is plugin hybrid, and BEV is battery plug-in EV (not hybrid/no internal combustion engine).

The next six array groups in the file are named according to the variables to extrapolate, and each value (e.g. NumMakes) is split into a variable named with the model, an underscore, and a fuel type (e.g. Car_Gas, Car-AV_Gas, etc.). Each model+fuel type has an interpolation method, years (which can be an array of multiple, such as [2030, 2050], or a single value), and values (if the years is an array, the values need to be an array of the same size). The interpolation methods are listed in Table 40.

By default, this file is output to vehicle_type_data_extended.csv, and that will need to be reflected in the vehicle type choice model setup.

A series of plots are generated from running the construct_veh_type_data.ipynb file showing the predicted variables (NumMakes, NumModels, MPG, Range, NewPrice, and auto_operating_cost) through 2050. Some example plots can be found in Figure 12.

TABLE 40: AV INTERPOLATION METHODS

METHOD NAME	METHOD DESCRIPTION
assert_constant	Asserts a constant value through the future year
assert_values	Asserts the future-year value and linearly extrapolates to that value
average	Uses the average value from the input vehicle table through the future year
linear	Linear extrapolation through the future year using observed data from the input vehicle table through the figure year
percentage_change	Percent change extrapolation (Vf = Vs * (1 + pct_change)^y)

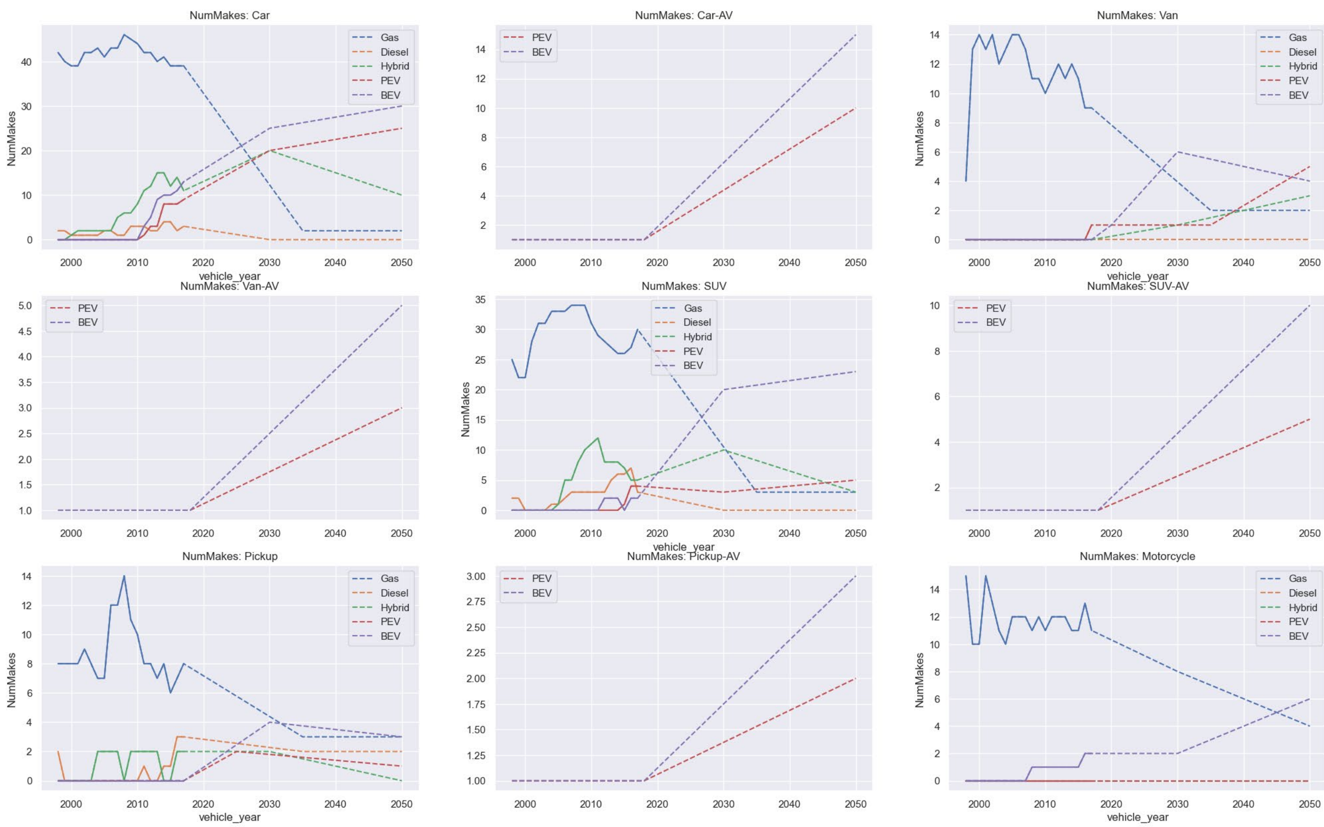


FIGURE 12: EXAMPLE PLOT FROM CONSTRUCT_VEH_TYPE_DATA.IPYNB

Vehicle Type Choice Setup

The vehicle type choice setup file, vehicle_type_choice.yaml also needs to be updated if the vehicle type assumptions have been updated. This requires updating the VEHICLE_TYPE_DATA_FILE as indicated below:

```
VEHICLE_TYPE_DATA_FILE: vehicle_type_data_extended.csv
```

In the current Gen3 Model, the vehicle_type_data.csv file used in the vehicle type choice model has already been extended to include vehicle type data between 1998 and 2050. Thus, there is no need to further extend it unless the vehicle type assumptions need to be updated.

7.3 EXAMPLE SCRIPTS

Model Crosstab Script

This script imports the person and trips tables from ActivitySim and returns a table of trips by person type. The first two lines import the pandas library and the os library. The next line sets the MODEL_FOLDER variable, which is used to simplify the code. The following two lines read the person and trips CSV files. The next line (`trips_ptype = …`) sets an index on the persons table to person id and joins the trip table that is grouped by person id and aggregated to output the count of trips. Then, this is grouped by `ptype` and aggregated to sum the trips. An example output follows the code.

```
import pandas as pd
import os

MODEL_FOLDER = r"E:\Gen3_Model\2018_base\outputs\activitysim"

person = pd.read_csv(os.path.join(MODEL_FOLDER, "final_persons.csv"))
trips = pd.read_csv(os.path.join(MODEL_FOLDER, "final_trips.csv"))

trips_ptype = person.set_index("person_id").join(trips.groupby("person_id").a
gg(trips = ('trip_id', 'count'))).groupby('ptype').agg(trips = ('trips', 'sum
'))
trips_ptype
```

TABLE 41: EXAMPLE SCRIPT OUTPUT - TRIPS BY PERSON TYPE CROSSTAB

PTYPE	TRIPS
1	2,329,534
2	290,270
3	187,006
4	446,242
5	369,753
6	110,169
7	512,477
8	200,079

Model Calibration Scripts

The following set of scripts is similar to many scripts used to calibrate ActivitySim model steps. This is written for the AV model but can be applied to many of the binary and multinomial logit models in ActivitySim.

Imports and Constants

The libraries used in most of the calibration code are listed in Table 42 along with what they are used for.

TABLE 42: PYTHON LIBRARIES USED IN CALIBRATION

LIBRARY NAME	LIBRARY DESCRIPTION
Pandas[48]	An open-source data analysis and manipulation tool
Os[49]	Provides many operating system functions
Numpy[50]	An open-source package for scientific computing

[48] https://pandas.pydata.org/
[49] https://docs.python.org/3/library/os.html
[50] https://numpy.org/

Matplotlib[51]	A package to produce visualizations
Seaborn[52]	A Python data visualization library based on Matplotlib

```
import pandas as pd
import os
import numpy as np
from matplotlib.pyplot import plot, show, draw, figure, cm
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib.colors import rgb2hex
```

In addition to the imports, the calibration scripts need some constants for the file location of the ActivitySim outputs, the sample rate, and the damping factor. Frequently, these are used in code as all-uppercase variables, such as below:

```
MODEL_FOLDER = "C:\\projects\\Gen3_Model\\2018_base\\outputs\\activitysim"
SAMPLE_RATE = 1.0
DAMPING = 0.75
```

File Reading and Manipulation

The next task in calibration is to read the validation data and ActivitySim output into memory.

```
census_data = pd.read_csv("ACS_2019_5YR_DC_MD_VA_WV_Vehicles_Available.csv")
model_households = pd.read_csv(os.path.join(MODEL_OUTPUT, "final_households.c
sv"))
```

In the case above, the files being read in are CSV files. Pandas can read Excel files, DBF files can be read by using the SimpleDBF Python library, and other file types can usually be read with an additional library if not with Pandas.

Since ActivitySim Output is disaggregate and most validation data is aggregate, ActivitySim output will need to be grouped. In some cases, validation data may also need to be grouped.

Example Script

These scripts are expected to be run in a Jupyter Notebook and are broken up into groups. The first group imports the libraries needed for the Notebook to work. This group also sets up paths, constants, and reads in the existing coefficients and model outputs.

```
import pandas as pd
import os
import numpy as np
from matplotlib.pyplot import plot, show, draw, figure, cm
import matplotlib.pyplot as plt
import seaborn as sns
```

[51] https://matplotlib.org/
[52] https://seaborn.pydata.org/

```
from matplotlib.colors import rgb2hex

MODEL_OUTPUT = r"C:\PATH"
MODEL_CONFIGS = r" C:\PATH "

TARGET_AV_PCT = {1: 0.05, 2: 0.1, 3: 0.25, 4: 0.85} # Income group : percent
COEF_DICT = {1: 'coef_av_inc1', 2: 'coef_av_inc2', 3: 'coef_av_inc3', 4: 'coe
f_av_inc4'}
# NOTE: This expects to see coef_av_inc1, coef_av_inc2, coef_av_inc3, and coe
f_av_inc4 in the coefficients file.
DAMPING = 0.75
av_coeffs = pd.read_csv(os.path.join(MODEL_CONFIGS, 'av_ownership_coeffs.csv'
))
model_hh = pd.read_csv(os.path.join(MODEL_OUTPUT, 'final_households.csv'))
```

This second group summarizes the household file by av ownership and compares it to the target percentages listed in the first code group. An example output table follows the

```
model_av_summary = model_hh[model_hh['av_ownership'] == True].groupby(['incom
e_segment', 'av_ownership']).agg(owns_av = ('av_ownership', 'count'))
model_av_summary['model'] = model_av_summary[['owns_av']] / model_hh.groupby(
'income_segment').agg(owns_av = ('income_segment', 'count'))
model_av_summary['target'] = model_av_summary.index.get_level_values('income_
segment').map(TARGET_AV_PCT)
model_av_summary
```

TABLE 43: EXAMPLE SCRIPT OUTPUT - AV COMPARISON

INCOME_SEGMENT	AV_OWNERSHIP	OWNS_AV	MODEL	TARGET
1	True	632	0.044753	0.05
2	True	555	0.101910	0.10
3	True	1020	0.252538	0.25
4	True	4804	0.754753	0.85

The third code block shows the data as a chart, suitable for saving or copying and pasting into a document. This is optional. Should this code not be used, the matplotlib and seaborn imports in the first code block can be omitted. Following the code is an example of the resulting chart.

```
plot_data = model_av_summary.reset_index().melt(id_vars = 'income_segment', v
alue_vars = ['model', 'target'], value_name = 'percent')
fig = plt.figure(figsize=(20, 6))
plot_idx = 111
```

```
plt.subplot(plot_idx)
sns.barplot(data = plot_data, x = 'income_segment', y = 'percent', hue = 'var
iable')
plt.title(f"AV Ownership", fontsize=18)
plt.xticks(fontsize=16, rotation = 90)
plt.yticks(fontsize=16)
plt.ylabel('Percent', fontsize=16)
plt.xlabel('Income Segment', fontsize=16)
```

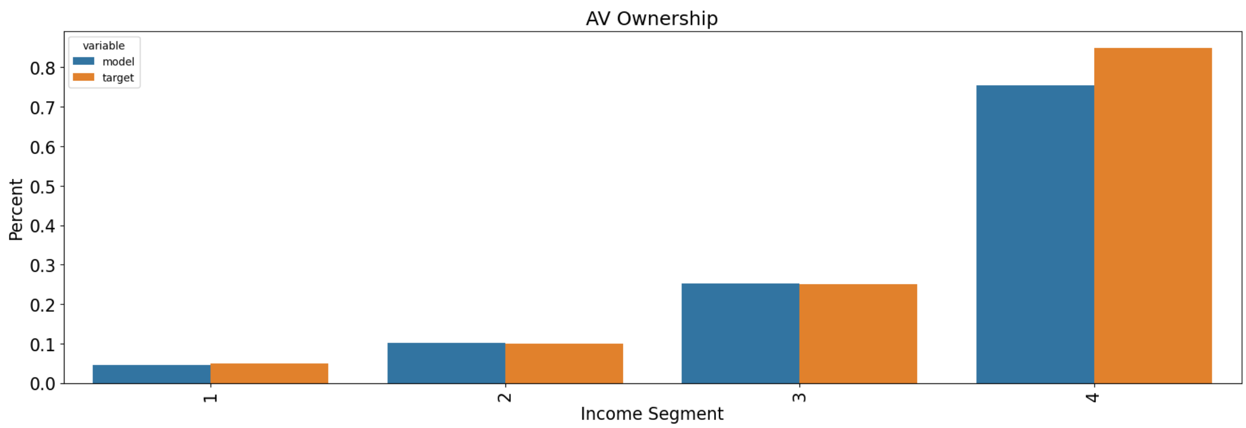


FIGURE 13: EXAMPLE CHART OUTPUT - AV COMPARISON

This fourth code block computes an adjustment to the coefficients. This uses a simple method to determine the new coefficient as shown in Equation 6. In this equation, the %Target and %Model can be replaced by actual sample numbers, keeping in mind that the model must be able to attain the sample number (for example, having a target number of households that exceeds the number of households in the model will cause the constant to be continually increased).

EQUATION 6: CONSTANT CALIBRATION EQUATION

$$K_{new} = K_{old} + damping * ln\left(\frac{\%Target}{\%Model}\right)$$

Where:

K_{new} is the new constant
K_{old} is the old constant
damping is the damping factor
%Target is the percent of households being targeted for that class
%Model is the percent of households the model is currently predicting for that class

```
model_av_summary['income_group'] = model_av_summary.index.get_level_values('i
ncome_segment')
model_av_summary['coefficient_name'] = model_av_summary.index.get_level_value
s('income_segment').map(COEF_DICT)
model_av_summary2 = model_av_summary.merge(av_coeffs.rename(columns = {'value
': 'input_coef'}), how = 'left', on = 'coefficient_name')
model_av_summary2['coef_adjust'] = np.log(model_av_summary2['target'] / model
_av_summary2['model'])
model_av_summary2['value'] = model_av_summary2['input_coef']
model_av_summary2.loc[model_av_summary2['constrain'] == 'F', 'value'] = model
_av_summary2['input_coef'] + DAMPING * model_av_summary2['coef_adjust']
model_av_summary2[['income_group', 'owns_av', 'model', 'target', 'coefficient
_name', 'input_coef', 'constrain', 'coef_adjust', 'value']]
```

This final code block will update the coefficient file and output the new coefficient file. This will overwrite the existing file without prompting the user, so there is a comment on the very first line to alert the user as to what they are doing.

```
# ⚠ THIS WILL CHANGE THE COEFFICIENT FILE!!!
av_coeffs.set_index('coefficient_name', inplace = True)
av_coeffs.update(model_av_summary2[['coefficient_name', 'value']].set_index('
coefficient_name'))
av_coeffs.to_csv(os.path.join(MODEL_CONFIGS, 'av_ownership_coeffs.csv'))
```

One caveat with this process is that it expects that the user will run the model after writing the new coefficients. There is no check to see if the coefficients file is newer than the output file, and running these script blocks multiple times without running the model after the final group will cause this script to re-adjust the coefficients.

7.4 MISCELLANEOUS

The items in this section are important to the functionality of the model, but not to the results of the model.

Changing the Disk Space Threshold

To ensure that the model has sufficient disk space to ensure the model can complete, the model script checks disk space and will not proceed if the server has less than 600 GB disk space. This is changed by opening Gen3_Model\source\scripts\python\ check_free_space.py and changing the DISK_SPACE_THRESHOLD variable near the top of the file.

55 Railroad Row

White River Junction, VT 05001

802.295.4999

www.rsginc.com

White River Junction & Burlington, VT | Arlington, VA | Chicago, IL | Portland, OR | San Diego, CA

RSG promotes sustainable business practices that minimize negative impacts on the environment. We print all proposals and reports on recycled paper that utilizes a minimum of 30% postconsumer waste. RSG also encourages recycling of printed materials (including this document) whenever practicable. **For more information on RSG's sustainability practices, please visit www.rsginc.com.**