# ModeChoice (version 6.0.6)

The ModeChoice program is used to:

1. Apply a nested-logit or multi-nomial mode choice model to zone-based trip tables.
   1. Read trip tables and one or more mode-specific skim files in TPPlus or TransCAD format.
   2. Trip tables can be subdivided by income or some other traveler attribute.
   3. Mode choice constants can vary by geographic market segments and traveler attribute.
   4. Short and long walk accessibility to transit by origin and destination zone is used to determine mode shares by combined access markets.
   5. A user script interface is available for manipulating the skim and zonal attributes used for each origin-destination choice.
2. Apply an iterative model calibration process to match mode choice results to calibration targets.
   1. Calibration targets can be defined by mode, market segment, and traveler attribute.
   2. Exit criteria are control by a maximum number of iterations and a RMSE convergence value.
   3. Input constants with minimum and maximum constraints are used as seed values for the calibration process.
   4. The output calibration file can be used for mode choice applications or as input to additional calibration iterations.
3. Generate reports summarizing the mode shares, market segments, and calibration results.
4. Generate text files summarizing the mode shares, market segments, and calibration results.
5. Generate model results in the input format required by FTA’s SUMMIT program.
6. Generate production and attraction zone mode split summary files.

ModeChoice is a console-based program that runs in a command window on either Windows or Linux. The command syntax is:

## ModeChoice [-flag] [control\_file]

The control\_file is the directory path and file name of a text file that contains the control strings expected by the program. If a file name is not provided, the program will prompt the user to enter a file name. The program automatically creates a printout file based on the control\_file name. If the file name includes an extension (e.g., “.ctl”), the extension is removed and “.prn” is added. The printout file will be created in the current working directory and will overwrite an existing file with the same name.

Additional information about program syntax, command lines flags, control keys and comments can be found in the Execution Service document.

A typical ModeChoice control file is shown below. These keys can be defined in a variety of different ways to perform the tasks listed above.

TITLE Metrolina HBW PEAK Mode Choice

PROJECT\_DIRECTORY D:\Metrolina\2013

NUMBER\_OF\_THREADS 1

TRIP\_TABLE\_FILE TripTables\HBW\_PEAK\_TRIPS.mtx

TRIP\_TABLE\_FORMAT TRANSCAD

SKIM\_FILE\_1 AutoSkims\SPMAT\_auto.mtx

SKIM\_FORMAT TRANSCAD

SKIM\_FILE\_2 Skims\TR\_NonMotorized.MTX

SKIM\_FORMAT TRANSCAD

SKIM\_FILE\_3 Skims\PK\_WKTRAN\_SKIMS.MTX

SKIM\_FORMAT TRANSCAD

SKIM\_FILE\_4 Skims\PK\_DRVTRAN\_SKIMS.MTX

SKIM\_FORMAT TRANSCAD

SKIM\_FILE\_5 Skims\PK\_DROPTRAN\_SKIMS.MTX

SKIM\_FORMAT TRANSCAD

ZONE\_FILE TAZ\_ATYPE.ASC

ZONE\_FORMAT TRANSCAD:TEXT

NEW\_TRIP\_TABLE\_FILE ModeSplit\HBW\_PEAK\_MS.mtx

NEW\_TRIP\_TABLE\_FORMAT TRANSCAD

SELECT\_TRIP\_TABLES INCOME1, INCOME2, INCOME3, INCOME4

MODE\_CONSTANT\_FILE ModeSplit\INPUTS\HBW\_PEAK\_Constant.txt

MODE\_BIAS\_FILE ModeSplit\INPUTS\HBW\_PEAK\_Bias.txt

MODE\_CHOICE\_SCRIPT ModeSplit\INPUTS\Mode\_Choice\_Script.txt

SEGMENT\_MAP\_FILE ModeSplit\INPUTS\Controls\Segment\_Map.txt

ORIGIN\_MAP\_FIELD CBD\_FLAG

DESTINATION\_MAP\_FIELD CBD\_FLAG

CALIBRATION\_TARGET\_FILE ModeSplit\INPUTS\Targets\HBW\_Target\_PEAK.txt

CALIBRATION\_SCALING\_FACTOR 1.0

MAX\_CALIBRATION\_ITERATIONS 10

CALIBRATION\_EXIT\_RMSE 0.1

NEW\_MODE\_CONSTANT\_FILE ModeSplit\Results\HBW\_PEAK\_Constant\_New.txt

##NEW\_CALIBRATION\_DATA\_FILE ModeSplit\Results\HBW\_PEAK\_Data.txt

##ADJUST\_FIRST\_MODE\_CONSTANTS FALSE

##REPORT\_AFTER\_ITERATIONS 3

TRIP\_PURPOSE\_LABEL Home-Based Work

TRIP\_PURPOSE\_NUMBER 1

TRIP\_TIME\_PERIOD 1

##SELECT\_ORIGIN\_ZONES 10251

##SELECT\_DESTINATION\_ZONES 10821

PRIMARY\_MODE\_CHOICE AUTO, TRANSIT, WALKBIKE

MODE\_CHOICE\_NEST\_1 AUTO = SOV, HOV

MODE\_CHOICE\_NEST\_2 HOV = POOL2, POOL3

MODE\_CHOICE\_NEST\_3 TRANSIT = WALKTRAN, DRIVETRAN, DROPTRAN

MODE\_CHOICE\_NEST\_4 WALKTRAN = WALKPREM, WALKBUS

MODE\_CHOICE\_NEST\_5 DRIVETRAN = DRIVEPREM, DRIVEBUS

MODE\_CHOICE\_NEST\_6 DROPTRAN = DROPPREM, DROPBUS

MODE\_CHOICE\_NEST\_7 WALKBIKE = WALK, BIKE

NESTING\_COEFFICIENT\_1 0.655

NESTING\_COEFFICIENT\_2 0.458

NESTING\_COEFFICIENT\_3 0.655

NESTING\_COEFFICIENT\_4 0.458

NESTING\_COEFFICIENT\_5 0.458

NESTING\_COEFFICIENT\_6 0.458

NESTING\_COEFFICIENT\_7 0.655

MODEL\_NAMES\_1 INCOME1, INCOME2, INCOME3, INCOME4

VEHICLE\_TIME\_VALUES\_1 -0.02202

WALK\_TIME\_VALUES\_1 -0.05680

DRIVE\_ACCESS\_VALUES\_1 -0.02202

WAIT\_TIME\_VALUES\_1 -0.03303

TRANSFER\_TIME\_VALUES\_1 -0.04404

PENALTY\_TIME\_VALUES\_1 -0.02202

COST\_VALUES\_1 -0.00329, -0.00165, -0.00089, -0.00070

USER\_VALUES\_1 -0.03290, -0.01650, -0.00890, -0.00700

DIFFERENCE\_VALUES\_1 -0.01542

MODE\_ACCESS\_MARKET\_1 SOV, POOL2, POOL3, DRIVEBUS, DRIVEPREM, DROPBUS, DROPPREM, WALKBUS, WALKPREM, WALK, BIKE

MODE\_ACCESS\_MARKET\_2 SOV, POOL2, POOL3, DRIVEBUS, DRIVEPREM, DROPBUS, DROPPREM

MODE\_ACCESS\_MARKET\_3 SOV, POOL2, POOL3

ACCESS\_MARKET\_NAME\_1 Can Walk to Transit at the Origin and Destination

ACCESS\_MARKET\_NAME\_2 Must Drive at the Origin and Can Walk to Transit the Destination

ACCESS\_MARKET\_NAME\_3 Must Drive

NEW\_TABLE\_MODES\_1 SOV = Drive Alone

NEW\_TABLE\_MODES\_2 POOL2 = Carpool 2

NEW\_TABLE\_MODES\_3 POOL3 = Carpool 3

NEW\_TABLE\_MODES\_4 WALKPREM = Wk-Premium

NEW\_TABLE\_MODES\_5 WALKBUS = Wk-Bus

NEW\_TABLE\_MODES\_6 DRIVEPREM = Dr-Premium

NEW\_TABLE\_MODES\_7 DRIVEBUS = Dr-Bus

NEW\_TABLE\_MODES\_8 DROPPREM = DropOff-Premium

NEW\_TABLE\_MODES\_9 DROPBUS = DropOff-Bus

NEW\_TABLE\_MODES\_10 WALK = Walk

NEW\_TABLE\_MODES\_11 BIKE = Bike

NEW\_MODE\_SUMMARY\_FILE ModeSplit\Results\2013\_HBW\_PEAK\_Summary.txt

NEW\_MARKET\_SEGMENT\_FILE ModeSplit\Results\2013\_HBW\_PEAK\_Segment.txt

##NEW\_MODE\_SEGMENT\_FILE ModeSplit\Results\HBW\_Mode\_Seg.txt

##NEW\_FTA\_SUMMIT\_FILE ModeSplit\Results\HBW\_Summit.bin

NEW\_PRODUCTION\_FILE ModeSplit\Results\2013\_HBW\_PEAK\_Productions.txt

NEW\_ATTRACTION\_FILE ModeSplit\Results\2013\_HBW\_PEAK\_Attractions.txt

MODECHOICE\_REPORT\_1 MODE\_CHOICE\_SCRIPT

MODECHOICE\_REPORT\_2 MARKET\_SEGMENT\_REPORT

MODECHOICE\_REPORT\_3 MODE\_SUMMARY\_REPORT

##MODECHOICE\_REPORT\_4 CALIBRATION\_REPORT

##MODECHOICE\_REPORT\_5 TARGET\_DATA\_REPORT

MODECHOICE\_REPORT\_6 MODE\_VALUE\_SUMMARY

MODECHOICE\_REPORT\_7 SEGMENT\_VALUE\_SUMMARY

##MODECHOICE\_REPORT\_8 MODE\_CHOICE\_STACK

##MODECHOICE\_REPORT\_9 ACCESS\_MARKET\_SUMMARY

##MODECHOICE\_REPORT\_10 LOST\_TRIPS\_REPORT

## Execution Service Keys

The ModeChoice program inherits the following control keys from Execution Service.

### TITLE (optional, text)

### REPORT\_DIRECTORY (optional, path)

### REPORT\_FILE (optional, output file)

### REPORT\_FLAG (optional, flag, FALSE)

### PAGE\_LENGTH (optional, integer, 65, >= 0)

### PROJECT\_DIRECTORY (optional, text)

### DEFAULT\_FILE\_FORMAT (optional, text, TAB\_DELIMITED)

## Selection Service Keys:

The ModeChoice program inherits the following control keys from Selection Service.

### SELECT\_ORIGIN\_ZONES (optional, list, ALL, > 0)

### SELECT\_DESTINATION\_ZONES (optional, list, ALL, > 0)

## Input-Output File Keys:

### TRIP\_TABLE\_FILE (required, input file)

This key specifies the name and location of the input person trip file that will be read by the program and distributed to various travel modes. The full path and file name for the input file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. The file may contain multiple tables representing trip purposes, income groups, or auto ownership levels.

### TRIP\_TABLE\_FORMAT (optional, text, TPPLUS)

If the Trip file includes a definition file, this key is ignored. If a definition file is not provided, this key identifies the file format so that the program can create a definition file. The format options include TRANSCAD, CUBE, TPPLUS, TRANPLAN, BINARY, FIXED\_COLUMN, COMMA\_DELIMITED, SPACE\_DELIMITED, TAB\_DELIMITED, CSV\_DELIMITED, DBASE, and SQLITE3.

### NEW\_TRIP\_TABLE\_FILE (required, output file)

This key specifies the name and location of the output trip file were the mode choice results are stored. The full path and file name for the output file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. The file will contain multiple tables representing trip assigned to each mode.

### NEW\_TRIP\_TABLE\_FORMAT (optional, text, TPPLUS)

This key is used to override the default file format for creating data files. The options include TRANSCAD, CUBE, TPPLUS, TRANPLAN, BINARY, FIXED\_COLUMN, COMMA\_DELIMITED, SPACE\_DELIMITED, TAB\_DELIMITED, CSV\_DELIMITED, DBASE, and SQLITE3.

### SELECT\_TRIP\_TABLES (optional, text, ALL)

This key is used to select table names from the input trip file for processing. If the key is ALL or not provided, all tables in the input trip file will be processed. The key is interpreted as a comma or space delimited list of text labels that must match the table names found in the trip file.

### SKIM\_FILE\_# (required, input file)

This key specifies the name and location of one or more input data files containing one or more travel attributes (e.g., time, distance and cost) between zones. The full path and file name for the input file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. The \_# is used to identify multiple input files. Each input file typically represents the travel attributes of a given travel mode.

### SKIM\_FORMAT (optional, text, TPPLUS)

If the Skim file includes a definition file, this key is ignored. If a definition file is not provided, this key identifies the file format so that the program can create a definition file. The format options include TRANSCAD, CUBE, TPPLUS, TRANPLAN, BINARY, FIXED\_COLUMN, COMMA\_DELIMITED, SPACE\_DELIMITED, TAB\_DELIMITED, CSV\_DELIMITED, DBASE, and SQLITE3. The skim format applies to all input skim files.

### ZONE\_FILE (required, input file)

This key specifies the name and location of the input zone file that defines attributes of the trip origin and destination. The full path and file name for the input file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. The file may contain area types, terminal time values, parking costs, density measures, and other zone-related attributes required by the mode choice model.

### ZONE\_FORMAT (optional, text, TAB\_DELIMITED)

If the Zone file includes a definition file, this key is ignored. If a definition file is not provided, this key identifies the file format so that the program can create a definition file. The format options include TEXT, BINARY, FIXED\_COLUMN, COMMA\_DELIMITED, SPACE\_DELIMITED, TAB\_DELIMITED, CSV\_DELIMITED, DBASE, ARCVIEW, and SQLITE3.

### MODE\_CONSTANT\_FILE (optional, input file)

This key specifies the name and location of the input file that defines mode specific constants. The full path and file name for the input file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. This file is typically the output file of the model calibration process. It is assumed to be a standard TRANSIMS data file with an accompanying definition file. A detailed description of the file structure is found in the Model Structure section of this document.

### MODE\_BIAS\_FILE (optional, input file)

This key specifies the name and location of the input file that defines mode specific bias values. Unlike the mode constant file, the mode bias file is typically specified as input to the calibration and application process. The full path and file name for the input file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. It is assumed to be a standard TRANSIMS data file with an accompanying definition file. A detailed description of the file structure is found in the Model Structure section of this document.

### MODE\_CHOICE\_SCRIPT (required, input file)

This key specifies the name and location of the input file that defines a user program or script for the mode choice model. The full path and file name for the input file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. The model script sets the values used in the utility function for each mode using various tables from the skim files and the attributes of the origin or destination zone record. Details about the scripting language available to a model user can be found in the User Program guide. Details about the file and field names expected by the ModeChoice program are found in the Model Structure section of this document.

## Market Segment Keys:

### SEGMENT\_MAP\_FILE (optional, input file)

This key specifies the name and location of the input file that defines market segment numbers. The full path and file name for the input file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. This file maps an attribute of the origin zone and an attribute of the destination zone to a market segment code. It is assumed to be a standard TRANSIMS data file with an accompanying definition file. A detailed description of the file structure is found in the Model Structure section of this document.

### ORIGIN\_MAP\_FIELD (optional, text, SEGMENT)

This key identifies the field name in the zone file that will be used to associate a given origin zone with a specific market segment. The key is only read if a segment map file is provided.

### DESTINATION\_MAP\_FIELD (optional, text, SEGMENT)

This key identifies the field name in the zone file that will be used to associate a given destination zone with a specific market segment. The key is only read if a segment map file is provided.

## Model Structure Keys:

### TRIP\_PURPOSE\_LABEL (optional, text, Peak HBW)

This is a user-defined text string that is printed at the top of output reports for general information.

### TRIP\_PURPOSE\_NUMBER (optional, integer, 1, 1..100)

This is a user-defined code number that identifies the trip purpose within the model script. The value is passed to the model script through the variable TRIP.PURPOSE.

### TRIP\_TIME\_PERIOD (optional, integer, 1, 1..100)

This is a user-defined code number that identifies the time period within the model script. The value is passed to the model script through the variable TRIP.PERIOD.

### PRIMARY\_MODE\_CHOICE (required, text)

This key defines the mode labels assigned to the primary mode choice. It includes a list of two or more user-defined text strings separated by a comma or space (e.g., AUTO, TRANSIT). The names defined here are used to build attribute fields recognized by the model script (e.g., AUTO.TIME) and define relationships and processing options specified using other control keys. They are also used by output reports to identify the mode. All mode names much be unique and follow the standard variable name syntax define in the User Program guide.

### MODE\_CHOICE\_NEST\_# (optional, text)

A nested logit model structure is specified using one or more of these keys. Each key associates a mode label defined in a higher level key with a comma or space delimited list of user-defined text strings representing the nested mode names. For example, if the primary mode choice includes AUTO as one of its mode labels, the key

MODE\_CHOICE\_NEST\_1 AUTO = SOV, HOV

subdivides the AUTO mode into two submodes labeled SOV and HOV. The new mode names defined by this key are used to build attribute fields recognized by the model script (e.g., SOV.TIME) and define relationships and processing options specified using other control keys. They are also used by output reports to identify the mode. Any number of nested records can be defined. The only restriction is that the mode labels specified to the left of the equal sign must have been defined in the primary mode list or the mode list of a nest key with a lower key number (i.e., \_#).

### NESTING\_COEFFICIENT\_# (optional, decimal, 0.5, 0.0..1.0)

In a nested model, the model script should define the attributes of each mode that is not nested. The attributes and utility of the nested mode are defined by the logsum of its nested members. This logsum is combined with other modes in a higher level mode split using a nesting coefficient. This key defines the nesting coefficient for the mode nest with the same key number (i.e., \_#). The key may also be specified without a key number (i.e., NESTING\_COEFFICIENT). In this case, the key value is applied to all nests. The default coefficient is 0.5.

### MODEL\_NAMES\_# (optional, text)

This key specifies the names of one or more models for which mode choice is to be run. The model names are usually from the list of SELECT\_TRIP\_TABLES. The # is the index of the first model and the index for subsequent models listed will be incremented in steps of 1. If no # is specified, the index defaults to 1.

## Utility Parameter Keys:

The utility of a given mode is defined as the sum of a series of mode attributes multiplied by attribute weights (e.g., utility = a \* x + b \* y + c). The following keys define the weights for attributes set by the model script. Additional information about how these weights are applied can be found in the Model Structure section of this document. A separate value can be specified for each of the MODEL\_NAMES listed. The # refers to the model number for which the first of the listed parameter keys is specified. The subsequent parameter keys apply to the subsequent models listed in MODEL\_NAMES\_#. If only one utility parameter key value is specified, the same value is applied to all the models.

### VEHICLE\_TIME\_VALUES\_# (optional, decimal, -0.02, 0, -0.04..-0.01)

The vehicle time value is the weight generally associated with in-vehicle travel time. The value of this key is multiplied by the mode attribute field called TIME. TIME is typically defined in minutes.

### WALK\_TIME\_VALUES\_# (optional, decimal, 0, 0, -1.0..-0.01)

The walk time value is the weight generally associated with walk access or out-of-vehicle time. The value of this key is multiplied by the mode attribute field called WALK. WALK is typically defined in minutes.

### DRIVE\_ACCESS\_VALUES\_# (optional, decimal, 0, 0, -1.0..-0.01)

The drive access value is the weight generally associated with drive access time to transit for park-n-ride or kiss-n-ride modes. The value of this key is multiplied by the mode attribute field called AUTO. AUTO may be defined as time in minutes or distance in miles.

### WAIT\_TIME\_VALUES\_# (optional, decimal, 0, 0, -1.0..-0.01)

The wait time value is the weight generally associated with waiting for the first transit vehicle. It could also represent the total waiting time for all transit boardings. The value of this key is multiplied by the mode attribute field called WAIT. WAIT is typically defined in minutes.

### LONG\_WAIT\_VALUES\_# (optional, decimal. 0, 0, -1.0..-0.01)

Most models calculate waiting time as a function of the headway of the available transit routes between two stops. One half of the headway is typically used for frequent transit service. If the transit option has infrequent service, the mode choice model may include an average wait time that travelers arrive prior to each scheduled departure (using the WAIT field) plus a penalty factor that attempts to capture the disutility associated with infrequent service. The long wait value key can be used to include an additional penalty for infrequent service. The value of this key is multiplied by the mode attribute field called LWAIT. LWAIT is typically defined in minutes, but may represent other values as well.

### TRANSFER\_TIME\_VALUES\_# (optional, decimal, 0, 0, -1.0..-0.01)

If a transit trip requires a transfer between transit routes, the second wait may be valued by the traveler differently from the waiting time for the first transit route. The transfer time value is the weight generally associated with waiting to board the second or third transit route. The value of this key is multiplied by the mode attribute field called XWAIT. XWAIT is typically defined in minutes.

### TRANSFER\_COUNT\_VALUES\_# (optional, decimal, 0, 0, -1.0..-0.01)

The transfer count value is the disutility associated with the number of transfer included in the transit path. The key value is multiplied by the mode attribute field called XFER. XFER is the number of transfers.

### PENALTY\_TIME\_VALUES\_# (optional, decimal, 0, 0, -1.0..-0.01)

The penalty time value can be used to include additional time-related attributes in the utility function. This value of this key is multiplied by the mode attribute field TPEN. TPEN may be defined in minutes or simply be a 0/1 flag value that defines when to include the penalty for a given mode.

### TERMINAL\_TIME\_VALUES\_# (optional, decimal, 0, 0, -1.0..-0.01)

The terminal time value is the weight generally associated with the extra time required to complete the trip at a given origin or destination. It may represent the time required to park the car and walk to the destination for an auto trip or additional access disutility for transit trips. The value of this key is multiplied by the mode attribute field called TERM. TERM is typically defined in minutes.

### DIFFERENCE\_VALUES\_# (optional, decimal, 0, 0, -1.0..-0.01)

The difference value is the weight generally associated with the difference in travel times between modes (e.g., between HOV and SOV trips). The value of this key is multiplied by the mode attribute field called DIFF. DIFF is typically defined in minutes.

### DISTANCE\_VALUES\_# (optional, decimal, 0, 0, -1.0..-0.01)

The distance value is the weight generally associated with the trip length between given origin or destination. The value of this key is multiplied by the mode attribute field called DIST. DIST is typically defined in miles.

### COST\_VALUES\_# (optional, decimal, 0, 0, -5.0..0.0)

The cost value is the weight generally associated with travel cost. The value of this key is multiplied by the mode attribute field called COST. The COST field will typically include transit fares and/or parking and auto operating costs in cents.

Note there is only one COST field, but there can be multiple cost values. Each cost value is mapped directly to one of the tables in the input trip file. The first cost value (i.e., COST\_VALUE\_1) is applied to the trips included in the first trip table in the trip file, or if the SELECT\_TRIP\_TABLES key is provided, the first table included in the selection list. The cost of the trip does not change as a function of the trip table being processed. In other words, the path is assumed to be the same for each trip table being processed.

Multiple cost values are typically used to include different “values of time” or in this case values of cost in the mode choice model. If, for example, the input trip file contains multiple tables representing income groups, the modeler might wish to make the relative impact of transit fares or parking costs higher for low income travelers than for high income travelers. Assuming income increases with table number, the cost value #1 would be a larger negative number than the cost value #2. This makes the disutility of low income trips higher as costs increases and as a consequence reduces the importance of travel time differences in the mode choice. This can be interpreted as a lower value of time for low income travelers and a higher value of time of high income travelers.

### USER\_VALUES\_# (optional, decimal, 0, 0, -5.0..0.0)

The user value is a weight the user can apply to a given attribute based on the target trip table. The value of this key is multiplied by the mode attribute field called USER.

Note there is only one USER field, but there can be multiple user values. Each user value is mapped directly to one of the tables in the input trip file. The first user value (i.e., USER\_VALUE\_TABLE\_1) is applied to the trips included in the first trip table in the trip file, or if the SELECT\_TRIP\_TABLES key is provided, the first table included in the selection list. The user attribute of the trip does not change as a function of the trip table being processed. In other words, the path is assumed to be the same for each trip table being processed.

Multiple user values are typically used to include different values of the user attribute in the mode choice model. If, for example, the input trip file contains multiple tables representing income groups, the modeler might wish to make the relative impact of the user attribute higher for low income travelers than for high income travelers. Assuming income increases with table number, the user value #1 would be a larger negative number than the user value #2. This makes the disutility of low income trips higher as the user attribute increases and as a consequence reduces the importance of travel time differences in the mode choice.

## Output Specification Keys:

### MODE\_ACCESS\_MARKET\_# (optional, text, , SOV, SR2, SR3...)

In addition to travel markets based on origin-destination combinations and traveler attributes such as income or auto ownership, separate mode choice calculations can be made for each subset of the origin and destination zone that has access to various transit services. In its simplest form this is based on the percentage of a zone that can walk to a transit stop within a specified distance (e.g., 0.5 mile). Separate access markets could also be defined for longer walks to premium transit services or for drive access markets.

Each access market is defined using a mode access market key. The key number (\_#) corresponds to a field in the model script (e.g., TRIP.ACCESS1) that defines the share of the total trips that are included in each access market. The sum of access field values should total 1.0 in order to distribute all of the trips to a single access market.

The mode access market key includes a comma or space delimited list of the mode names that are available to travelers within the corresponding access market. The mode names are defined by the PRIMARY\_MODE\_CHOICE and MODE\_CHOICE\_NEST\_# keys. Mode names may be used in all access markets or only a few access markets. If the model is nested, only the non-nested modes should be included in the access market keys.

The following is a very simple example of how access markets are defined and used. Let’s assume a very simple model with two modes (AUTO and TRANSIT). In order to choose the TRANSIT mode, the traveler’s origin and destination must be within walking distance of a transit stop. This results in two access markets – the share of the trips between the origin and destination that have walk access to transit at both ends of the trip and the share of trips that don’t have a transit option. If we assume the input zone file includes a transit walk coverage field called COVERAGE, the model script will define the access travel markets using a calculation like:

TRIP.ACCESS1 = ORG.COVERAGE \* DES.COVERAGE //---- walk to transit market ----

TRIP.ACCESS2 = 1.0 – TRIP.ACCESS1 //---- no transit option ----

In this case, two mode access market keys would be included in the control file:

MODE\_ACCESS\_MARKET\_1 AUTO, TRANSIT

MODE\_ACCESS\_MARKET\_2 AUTO

The program will split the input trips into two access markets and apply the mode choice calculations to each market based on the modes that are available within that market. In this case, all trips in market #2 will choose AUTO and a percentage of the trips in market #1 will choose AUTO. Note that the disutility of the AUTO choice is the same for both access markets, but the number of mode options available to the traveler vary between access markets.

### NEW\_TABLE\_MODES\_# (optional, text)

Mode choice calculations are made for each access market and mode name defined using the PRIMARY\_MODE\_CHOICE and MODE\_CHOICE\_NEST\_# keys. Each mode name and access market can be saved to the output trip file in a separate table or combined into one or more aggregate tables. The mode names and access markets assigned to each output table are defined using this key. The key specifies a comma or spaced delimited list of mode names followed by an equal sign and the name given to the output table. For example, the key

NEW\_TABLE\_MODES\_1 SOV = Drive Alone

would create a table called “Drive Alone” in the new trip file and save the mode shares from the SOV mode to this table. If the key does not include an equal sign, the name of the output table is equal to the name of the mode (e.g., SOV).

In the example shown above, the SOV trips from each access market are summed and saved as a single value. If the user wishes to save the SOV trips for a given access market into a separate output table, the access market code number can be appended to the mode name to select the subset of SOV trips generated by the access market. This is done using the following syntax:

NEW\_TABLE\_MODES\_1 SOV.1 = SOV Access Market 1

The syntax that combines multiple modes into an aggregate output table would look like this:

NEW\_TABLE\_MODES\_1 SOV, SR2, SR3 = Auto Trips

This command sums the trips assigned to SOV, SR2, and SR3 modes for all access markets and outputs the total value in a table called “Auto Trips”.

### OUTPUT\_TRIP\_FACTOR (optional, decimal, 1.0, 1.0..1000.0)

This factor is applied to the trips assigned to each mode prior to writing the value to the output matrix file. It is primarily used to compensate or correct for the number of significant digits written to the matrix. All internal calculations use double floating point numbers.

## Summary Data File Keys:

### NEW\_MODE\_SUMMARY\_FILE (optional, output file)

In addition to writing the mode choice summary report to the printout file, the user may choose to create a tab delimited file that can be processed or formatted by Excel for documentation and analysis purposes. The full path and file name for the output file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. An example of the type of file created by this key is shown below.

### NEW\_MARKET\_SEGMENT\_FILE (optional, output file)

If a segment map file is provided, the mode choice results can also be summarized by geographic market segments. The user may choose to create a tab delimited file that can be processed or formatted by Excel for documentation and analysis purposes. The full path and file name for the output file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. Selected records from a market segment file are shown below.



### NEW\_MODE\_SEGMENT\_FILE (optional, output file)

If a segment map file is provided, the mode choice results can also be summarized by mode and market segment in a matrix format. The user may choose to create a tab delimited file that can be processed or formatted by Excel for documentation and analysis purposes. The full path and file name for the output file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. An example of the type of file (18 modes by 2 market segments) created by this key is shown below.



### NEW\_FTA\_SUMMIT\_FILE (optional, output file)

The ModeChoice program can summarize and write the data needed as input for the FTA SUMMIT program used to generate User Benefits data for the FTA New Starts process. The full path and file name for the output file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. The file is in binary format, ready for input to SUMMIT.

### NEW\_PRODUCTION\_FILE (optional, output file)

The program can summarize and write mode choice results for each origin zone into an output file that can be processed or formatted by Excel for documentation and analysis purposes. The full path and file name for the output file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. Selected records from a production file are shown below.

### NEW\_PRODUCTION\_FORMAT (optional, text, TAB\_DELIMITED)

This key is used to override the default file format for creating data files. The options include BINARY, FIXED\_COLUMN, COMMA\_DELIMITED, SPACE\_DELIMITED, TAB\_DELIMITED, CSV\_DELIMITED, DBASE, and SQLITE3.

### NEW\_ATTRACTION\_FILE (optional, output file)

The program can summarize and write mode choice results for each destination zone into an output file that can be processed or formatted by Excel for documentation and analysis purposes. The full path and file name for the output file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. Selected records from an attraction file are shown below.

### NEW\_ATTRACTION\_FORMAT (optional, text, TAB\_DELIMITED)

This key is used to override the default file format for creating data files. The options include BINARY, FIXED\_COLUMN, COMMA\_DELIMITED, SPACE\_DELIMITED, TAB\_DELIMITED, CSV\_DELIMITED, DBASE, and SQLITE3.

## Model Calibration Keys:

The ModeChoice program can be operated in production mode or calibration mode. In production mode the choice model is applied once and the outputs are generated. In calibration mode, the choice model is applied and the differences between the calibration targets and the mode specific results are evaluated. The estimation errors are used to adjust the mode specific constants and the mode choice model is applied again. The process continues until the maximum number of iterations is reached or the model converges.

The constants used in the ModeChoice program are based on the McFadden structure of a nested logit model. As such all coefficients and constants used in the disutility function (except nest-specific constants) are specified in terms of their equivalent top-level, main mode choice values. Prior to computing lower level disutilities, all coefficients and constants are divided by the product of all pertinent nest coefficients to create an equivalent lower-level constant. In coding the constant file for the ModeChoice program it is important to note that this model assumes the McFadden formulation and that some popular model estimation programs output constants according to the Daly formulation in which constants are specified in terms of their bottom-level value.

### CALIBRATION\_TARGET\_FILE (optional, input file)

This key specifies the name and location of the calibration target file. The full path and file name for the calibration file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. This key must be defined for mode constant calibration to take place. The target file is a tab delimited text file with fields and rows that correspond to the model structure, market segments, and trip table names. A sample target file for models without market segments is shown below. This is followed by an example of a target file with market segments. The Min\_Const and Max\_Const fields define the maximum and minimum constant values the calibration process can generate. Once this value is reached, no additional adjustments to the mode constant area made.





### CALIBRATION\_SCALING\_FACTOR (optional, decimal, 1, 1.0..5.0)

This factor is applied to the calibration targets prior to the calibration process. It is used to adjust the calibration targets to a different analysis year or to adjust the trip units.

### MAX\_CALIBRATION\_ITERATIONS (optional, integer, 20, 1..1000)

The calibration process adjusts the mode specific constants over multiple iterations. This key defines the maximum number of iterations that may be performed before the process exits.

### CALIBRATION\_EXIT\_RMSE (optional, decimal, 5, .0..50.0)

The calibration process will exit after a maximum number of iterations or the root mean squared errors between the mode targets and the estimated trips are less than the value of this key.

### NEW\_MODE\_CONSTANT\_FILE (optional, output file)

After the last calibration iteration, the program writes the final set of mode specific constants in the same format as the input calibration target file. The full path and file name for the new mode constant file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key.

### NEW\_CALIBRATION\_DATA\_FILE (optional, output file)

During the calibration process, the program writes the estimated mode shares to the output printout file. The user may also choose to write the results of each iteration to a tab delimited text file that can be processed or formatted by Excel for documentation and analysis purposes. The full path and file name for the output file is constructed by appending the value of this key to the value of the PROJECT\_DIRECTORY key. A few samples lines from a calibration data file are shown below.

## Report Options:

### MODECHOICE\_REPORT\_# (optional, text)

This key is typically used to select reports written to the printout file. The “#” at the end of the report keyword represents the report number (e.g., MODECHOICE\_REPORT\_1). The key can be provided with additional numbers to specify additional reports. The reports are typically generated in numerical order (i.e., 1, 2, 3…).

The string parameter associated with a report keyword is limited to the following options:

MODE\_CHOICE\_SCRIPT

MODE\_CHOICE\_STACK

MODE\_SUMMARY\_REPORT

MARKET\_SEGMENT\_REPORT

CALIBRATION\_REPORT

TARGET\_DATA\_REPORT

MODE\_VALUE\_SUMMARY

SEGMENT\_VALUE\_SUMMARY

The above reports are printed in the “\*.prn” file that is generated in the same directory as the control file used to run the program. Each of above reports is described below:

#### Mode Choice Script

The mode choice script report is a simple listing of the processing logic included in the file identified by the Mode Choice Script key. An extract from a mode choice script report is shown below.

#### Mode Choice Stack

The mode choice stack report shows how the processing logic included in the file identified by the Mode Choice Script key is converted to compiled commands. An extract from a mode choice stack report is shown below.



#### Mode Summary Report

The mode summary report shows the trips allocated to each mode and from each trip table and the overall mode shares and table shares. An example of the mode summary report is shown below.

#### Market Segment Report

The market segment report has the same layout as the mode summary report, but lists the trips and mode shares for each market segment. In this case market segments can be defined as part of the mode choice model or simply included to summarize travel markets of special interest. An example of the market segment report is shown below.

#### Calibration Report

The calibration report shows the difference between the target value and the estimated number of trips after each iteration of the convergence process. It shows the difference, percent error, and the current set of mode-specific constants. The Minutes column to the rights translates the mode constant to an equivalent number of minutes in a vehicle. Large values imply that a significant portion of the explanatory power of the model is defined by the mode constant. The bottom of the report shows the absolute error, average error, percent root mean squared error (RMSE), and r-squared values for the overall model. The RMSE value is one of the stopping criteria for the iterative process.



#### Target Data Report

The target data report lists the calibration target trips by mode and sub-mode included in the file referenced by the mode value summary report shows the weighted average value of each attribute provided by the Calibration Target File key. An example of the target data report is shown below.



#### Mode Value Summary

The mode value summary report shows the weighted average value of each attribute provided by the mode script for each travel mode. This report is helpful in reviewing the attributes to ensure they are in the expected range. An example of the mode value summary report is shown below.



#### Segment Value Summary

The segment value summary report has the same layout as the mode value summary report, but lists the model attributes and mode shares for each market segment. In this case market segments can be defined as part of the mode choice model or simply included to summarize travel markets of special interest. An example of the segment value report is shown below.

## Model Structure

This section describes how the control keys, key values, and supporting data files work together to define various model structures.

##### Mode Constant File

At a minimum the mode constant file needs to include a mode and constant field. It interprets fields named “mode”, “m”, “mod”, and “modes” as the mode field and “constant” or “const” as the constant field. The text strings found in each row of the mode field must equal the mode labels defined by the primary and nested mode choice keys. For example, if the model structure includes the following keys:

PRIMARY\_MODE\_CHOICE AUTO, TRANSIT

MODE\_CHOICE\_NEST\_1 AUTO = SOV, HOV

The mode field values should be AUTO, TRANSIT, SOV, or HOV.

If the model includes market segment constants, a market segment field should be included in the mode constant file using the field name “segment”, “seg”, “market”, or “s”. A new set of mode specific constants should be provided for each market segment.

If the input trip file includes multiple tables or the select trip tables key includes multiple table names, the mode constant file may include a constant value corresponding to each table name. These are typically mode specific constants related to income groups or auto ownership levels. A sample mode constant file for a nested logic model with four income groups (INCOME1..INCOME4) and one market segment is shown below:



In a nested logit model, the constants for the modes at the higher level of the nest determine (along with mode and user attribute values and coefficients) the split among the higher level modes. The constants for modes at the lower level of the nest determine the share of the lower level mode out of the higher level mode share. For example, the constants for SOV and HOV modes in the above example determine the share of SOV and HOV modes out of the DRIVE mode share, which in turn is determined by the constant for DRIVE mode.

##### Mode Bias File

The mode bias values can be used to incorporate factors that are not explicitly captured by mode or user attributes or mode constants. Such factors may include auto availability or varied perception of a mode across market segments (e.g. income categories). In nested logit models, the bias value may be specified only for the modes at the lowest nest level.

At a minimum the mode bias file needs to include a mode and bias field. It interprets fields named “mode”, “m”, “mod”, and “modes” as the mode field and “bias” as the bias field. The text strings found in each row of the mode field must equal the mode labels defined by the primary and nested mode choice keys. For example, if the model structure includes the following keys:

PRIMARY\_MODE\_CHOICE AUTO, TRANSIT

MODE\_CHOICE\_NEST\_1 AUTO = SOV, HOV

The mode field values should be AUTO, TRANSIT, SOV, or HOV.

If the model includes market segment constants, a market segment field should be included in the mode bias file using the field name “segment”, “seg”, “market”, or “s”. A new set of mode specific bias should be provided for each market segment.

If the input trip file includes multiple tables or the select trip tables key includes multiple table names, the mode bias file may include a bias value corresponding to each table name. These are typically mode specific bias related to income groups or auto ownership levels. They only apply to non-nested modes. A sample mode constant file for a nested logic model with four income groups (INCOME1..INCOME4) and one market segment is shown below:

##### Segment Map File

The segment map file expects to find three fields in the data file. It interprets field names “origin”, “org”, “o”, or “i” as the origin field; “destination”, “des”, “d”, or “j” as the destination field; and “segment”, “seg”, “market”, or “s” as the market segment field. If the file does not include a file header, the program assumes the fields are in origin, destination, market segment order. Each field is assumed to be an integer number. The numbers found in the origin and destination fields are values assigned to the origin map field and the destination map field for a given zone in the zone file.

A typical segment map file is shown below. This file uses seven origin and destination area type codes to define 12 market segments.

|  |  |  |
| --- | --- | --- |
| ORG | DES | SEGMENT |
| 1 | 1 | 1 |
| 1 | 2 | 2 |
| 1 | 3 | 2 |
| 1 | 4 | 3 |
| 1 | 5 | 3 |
| 1 | 6 | 4 |
| 1 | 7 | 4 |
| 2 | 1 | 9 |
| 2 | 2 | 10 |
| 2 | 3 | 11 |
| 2 | 4 | 11 |
| 2 | 5 | 11 |
| 2 | 6 | 12 |
| 2 | 7 | 12 |
| 3 | 1 | 1 |
| 3 | 2 | 2 |
| 3 | 3 | 3 |
| 3 | 4 | 3 |
| 3 | 5 | 3 |
| 3 | 6 | 4 |
| 3 | 7 | 4 |
| 4 | 3 | 7 |
| 4 | 4 | 7 |
| 4 | 5 | 7 |
| 4 | 6 | 8 |
| 4 | 7 | 8 |
| 5 | 1 | 9 |
| 5 | 2 | 10 |
| 5 | 3 | 11 |
| 5 | 4 | 11 |
| 5 | 5 | 11 |
| 5 | 6 | 12 |
| 5 | 7 | 12 |
| 6 | 1 | 5 |
| 6 | 2 | 6 |
| 6 | 3 | 7 |
| 6 | 4 | 7 |

##### Mode Choice Script Interface

The program creates a number of custom file records to set values for use or update by the user-defined mode choice script. The zone data file, for example, is split into an “ORG” file and a “DES” file to give the user access to the zone-based attributes at the origin and destination ends of the trip. Each of the data fields included in the input zone file is copied to the ORG and DES files provided to the mode choice script. For example, if the zone file includes an AREA\_TYPE field, the command ORG.AREA\_TYPE will return the value of the AREA\_TYPE field corresponding to the origin zone number.

The program also adds fields to the input trip file to expand functionality. Normally the trip file will include the origin and destination zone numbers and the number of trips. The program adds a “PURPOSE”, “PERIOD”, “SEGMENT”, and several “ACCESS#” fields to the trip file. The value of the Trip Purpose Number control key is written to the “PURPOSE” field and the value of Trip Time Period control key is written to the “PERIOD” field. If a segment map is provided, the segment value generated by the segment map is written to the “SEGMENT” field. An access field is created for each of the Mode Access Market keys defined in the control file. The user script should calculate the market share for each access market and store that share in the corresponding TRIP.ACCESS# field.

The program also creates an attribute or value file for each of the user-defined modes. The file names correspond to the mode names defined by the Primary Mode Choice and Mode Choice Nest keys. For example, the keys

PRIMARY\_MODE\_CHOICE AUTO, TRANSIT

MODE\_CHOICE\_NEST\_1 AUTO = SOV, HOV

create four files for use by the modeling script. The file names will be “AUTO”, “TRANSIT”, “SOV” and “HOV”. Each of these files will include a program-defined series of field names. These field names are listed below along with the control key that is applied to the value of each field when calculating the mode utility.

|  |  |  |
| --- | --- | --- |
| **Description** | **Field Name** | **Control Key** |
| In-vehicle travel time | TIME | VEHICLE\_TIME\_VALUES |
| Walk access time | WALK | WALK\_TIME\_VALUES |
| Drive access time | AUTO | DRIVE\_ACCESS\_VALUES |
| Initial waiting time | WAIT | WAIT\_TIME\_VALUES |
| Long initial wait time | LWAIT | LONG\_WAIT\_VALUES |
| Transfer wait time | XWAIT | TRANSFER\_TIME\_VALUES |
| Additional travel time | TPEN | PENALTY\_TIME\_VALUES |
| Terminal Time | TERM | TERMINAL\_TIME\_VALUES |
| Time or other difference | DIFF | DIFFERENCE\_VALUES |
| Trip length | DIST | DISTANCE\_VALUES |
| Travel cost | COST | COST\_VALUES |
| User attribute | USER | USER\_VALUES |
| Transfer penalty | XFER | TRANSFER\_COUNT\_VALUES |
| Mode-specific bias | BIAS | 1 |
| Pedestrian Environment Factor | PEF | 1 |
| CBD constant | CBD | 1 |
| Mode constant value | CONSTANT | 1 |
| Table-specific constants | [table names] | 1 |
| Table-specific flag | NO\_[table names] | 0 |

Given the mode example listed above, the user can include in-vehicle travel time, terminal time, and cost in the utility equation by setting the following variables in the model script.

AUTO.TIME = SKIM2.TIME

AUTO.COST = 10 \* SKIM2.DISTANCE + SKIM2.TOLL + DES.PARK\_COST

AUTO.TERM = ORG.TERM\_TIME + DES.TERM\_TIME

The program will than calculate the mode utility by exponentiating the following equation:

Auto [x] = AUTO.TIME \* VEHICLE\_TIME\_VALUES[x] + AUTO.TERM \* TERMINAL\_TIME\_VALUES [x] + AUTO.COST \* COST\_VALUES [x] + AUTO.CONSTANT[x] + AUTO.BIAS[x]

Where [x] corresponds to each subdivision of trips defined in the input trip table. This is typically a distribution of trips by income group or auto ownership.

Note that the CONSTANT and BIAS fields do not need to be set by the model script. These values are pre-set with the constant and bias values included in the Mode Constant and Mode Bias Files. The program does, however, permit the user to modify these constant values for a particular interchange to implement special market segment adjustments. The constant values are re-initialized each time the script is executed.

The NO\_[table name] fields are used as a flag to exclude a given table from the mode choice model of a given mode. If this value is non-zero for a given table, no trips from that table will be assigned to the corresponding mode. For example, if one table represents trips from zero-car households, none of the trips from this table should be assigned to the drive alone mode.

The Auto [x] utility is used in the mode share calculation for each access market. The access market share (TRIP.ACCESS#) is multiplied by the number of trips in the corresponding trip table [x] to define the number of trips in the access market. These trips are then distributed to the modes included in the corresponding Mode Access Market key based on the model structure defined by the Primary Mode Choice and Mode Choice Nest keys. The sum of the mode utilities in the lowest level nest are calculated first. The log of this sum is factored by the nesting constant and added to the sum of the mode utilities included in the next higher nest. This continues until the primary mode choice is reached. The primary mode choice model splits the access market trips into the primary modes based on the ratio of the mode utility to the sum of the mode utilities. The shares assigned to each nested mode are calculated in the same way.

##### Simple Example

This section describes how to configure the software to implement a simple mode choice model involving a primary choice between auto and transit with a nested transit choice between walk and drive access. The trip table only includes total home-based work person trips. Three skim files are provided. The first is an auto skim, the second is the walk to transit skim and the third is the drive to transit skim.

The control file includes the following keys.

TRIP\_FILE TRIPS\HBW.MTX

SKIM\_FILE\_1 SKIMS\AUTO.SKM

SKIM\_FILE\_2 SKIMS\WALK.SKM

SKIM\_FILE\_3 SKIMS\DRIVE.SKM

ZONE\_FILE ZONE.TXT

NEW\_TRIP\_FILE TRIPS\HBW\_MS.MTX

MODE\_CONSTANT\_FILE HBW\_Constant.txt

MODE\_BIAS\_FILE HBW\_Bias.txt

MODE\_CHOICE\_SCRIPT Mode\_Choice\_Script.txt

PRIMARY\_MODE\_CHOICE AUTO, TRANSIT

MODE\_CHOICE\_NEST\_1 TRANSIT = WALK, DRIVE

NESTING\_COEFFICIENT 0.5

VEHICLE\_TIME\_VALUE -0.02

WALK\_TIME\_VALUE -0.04

DRIVE\_ACCESS\_VALUE -0.03

WAIT\_TIME\_VALUE -0.05

MODE\_ACCESS\_MARKET\_1 AUTO, WALK, DRIVE

MODE\_ACCESS\_MARKET\_2 AUTO, DRIVE

MODE\_ACCESS\_MARKET\_3 AUTO

NEW\_TABLE\_MODES\_1 AUTO

NEW\_TABLE\_MODES\_2 WALK = Walk Access

NEW\_TABLE\_MODES\_3 DRIVE = Drive Access

The mode constant file would look something like the following:

|  |  |
| --- | --- |
| MODE | CONSTANT |
| AUTO | 0.051099 |
| TRANSIT | -0.163902 |
| WALK | 2.684534 |
| DRIVE | -4.415451 |

The mode bias file would look something like the following:

|  |  |
| --- | --- |
| MODE | CONSTANT |
| AUTO | -0.5 |
| TRANSIT | 0 |
| WALK | 0 |
| DRIVE | -0.5 |

At a minimum, the mode choice script should include the following:

//---- transit access markets ----

TRIP.ACCESS1 = ORG.WALK\_COVERAGE \* DES.WALK\_COVERAGE

TRIP.ACCESS2 = (1 - ORG.WALK\_COVERAGE) \* DES.WALK\_COVERAGE

TRIP.ACCESS3 = 1 - TRIP.ACCESS1 - TRIP.ACCESS2

//---- auto ----

AUTO.TIME = SKIM1.TIME

//---- walk access to transit ----

IF (SKIM2.TIME > 0) THEN

WALK.TIME = SKIM2.TIME

WALK.WALK = SKIM2.WALK

WALK.WAIT = SKIM2.WAIT

ENDIF

//---- drive access to transit ----

IF (SKIM3.TIME > 0) THEN

DRIVE.TIME = SKIM3.TIME

DRIVE.WALK = SKIM3.WALK

DRIVE.DRIVE = SKIM3.DRIVE

DRIVE.WAIT = SKIM3.WAIT

ENDIF

Note that the mode names defined by the Primary Mode Choice and Mode Choice Nest keys in the control file are the same as the mode names in the mode constant file and the mode choice script. The mode utilities will be calculated as:

Auto = exp (-0.02 \* SKIM1.TIME + 0.051099 + -0.5)

Walk = exp (-0.02 \* SKIM2.TIME + -0.04 \* SKIM2.WALK + -0.05 \* SKIM2.WAIT + 2.684534)

Drive = exp (-0.02 \* SKIM3.TIME + -0.04 \* SKIM2.WALK + -0.03 \* SKIM3.DRIVE +

-0.05 \* SKIM3.WAIT + -4.415451 + -0.5)

Transit = exp (0.5 \* log (Walk + Drive) + -0.163902)

The model includes three mode access markets based on the percentages of the origin and destination zone that are within walking distance of transit service. The first market has walk access at the origin and destination ends of the trip. As a result, all three modes are an option in this market segment. The second market is not within walking distance at the origin end, but is within walking distance at the destination end of the trip. In this case, a walk access transit trip is not an option, but a drive access transit trip is a possibility. The third market is not within walking distance at both ends of the trip, so the only option is auto.

The mode shares for access market #1 are calculated as:

Auto Share = Auto / (Auto + Transit)

Transit Share = Transit / (Auto + Transit)

Walk Share = Transit Share \* Walk / (Walk + Drive)

Drive Share = Transit Share \* Drive / (Walk + Drive)

The mode shares for access market #2 are calculated as:

Auto Share = Auto / (Auto + Drive)

Drive Share = Drive / (Auto + Drive)

All of the trips in access market #3 are assigned to the auto mode.