Mosaic Tool Version 2 Documentation

Prepared for

Oregon Department of Transportation

November, 2014

Prepared by

Contents

List of Tables

List of Figures

Introduction

The purpose of this report is to provide detailed technical documentation on the estimation of specific indicators within Version 2.0 of the Mosaic tool.

It is structured as a series of documentation sheets, organized by category of transportation system performance. Each documentation sheet includes:

- A definition of the specific indicator;
- A description of the conceptual framework used in the estimation of the indicator, where applicable;
- A Structure & Logic (S&L) diagram describing how the indicator is estimated; and
- A table summarizing the data and assumptions used in the estimation of the indicator.

Before the first documentation sheet, a chapter on Mosaic's comparison process describes the methods and calculations used in the Benefit-Cost Analysis (BCA) and Multi-Objective Decision Analysis (MODA) components of Mosaic.

Supporting technical information and source code (written in Visual Basic for Applications, or VBA) can be found in the appendix.

Comparison Process

This chapter of the documentation explains how the Benefit-Cost Analysis results are calculated (Section 1) and how the outcomes of MODA are produced and summarized (Section 2).

Additional information on BCA methods and calculations are available in a number of online publications, including:

- The Green Book, Appraisal and Evaluation in Central Government, July 2011, https://www.gov.uk/government/publications/the-green-book-appraisal-and-evaluation-in-central-governent (last accessed October 23, 2014)
- Guide to Cost-Benefit Analysis of Investment Projects, European Commission, Directorate General Regional Policy, July 2008, http://ec.europa.eu/regional_policy/sources/docgener/guides/cost/guide2008_en.pdf (last accessed October 23, 2014)

A detailed documentation of the multi-criteria analysis principles and techniques embedded within the MODA component of the Mosaic tool can be found in:

- Multi-Criteria Analysis: A Manual, UK Communities and Local Government, January 2009, London; available at https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/7612/1132618.pdf (last accessed October 23, 2014)

Benefit-Cost Analysis Calculations

Benefit-Cost Analysis (BCA) is a conceptual framework that quantifies in monetary terms as many of the costs and benefits of a plan or project as possible. Benefits are defined broadly. They represent the extent to which people to whom they accrue are made better-off, as measured by their own willingness-to-pay. In other words, central to BCA is the idea that people are best able to judge what is good for them, what improves their well-being or welfare. BCA also adopts the view that a net increase in welfare (as measured by the summation of individual welfare changes) is a good thing, even if some groups within society are made worse-off. And a plan or bundle would be rated positively (e.g., positive Net Present Value, as defined below) if the benefits to some are large enough to *potentially* compensate the losses of others. Finally, BCA is forward-looking and must consider the future welfare impacts of a plan or bundle over its entire life-cycle. Future welfare changes are weighted against today's investment costs through present valuation, i.e., discounting.

The main BCA calculations included in the Mosaic tool – other than the summation of individual benefit and cost estimates – involve discounting and the expression of summary indicators in monetary terms.

Discounting is a method used to convert future benefits or costs into a common year (present value) for comparison. In Mosaic, the Present Value (PV) of future benefits and costs is calculated with the following formula:

$$PV = \frac{V_0}{(1+r)^0} + \frac{V_1}{(1+r)^1} + \cdots + \frac{V_N}{(1+r)^N} = \sum_{t=0}^{N} \frac{V_t}{(1+r)^t}$$

Where:

- *r* is the real discount rate selected for analysis;
- *V* is the estimated value of benefits (or costs) in Year t, expressed in constant, base-year dollars;
- *t* is the year in which the benefits or costs are realized (with $Year_0$ as the base year); and
- *N* is the last year in the period of analysis (e.g., *N* = 30).

Additional information on discounting and justifications for the range of discount rates included in the Mosaic tool can be found in ODOT's Recommendation Memo on discounting, available at https://www.oregon.gov/ODOT/Planning/Documents/Mosaic-Recommendations-for-Users.pdf (last accessed October 23, 2014).

Two summary indicators are estimated within the Mosaic workbook: the Net Present Value and Benefit/Cost Ratio. They are defined below.

The Net Present Value of a plan or bundle is the present value of total monetized benefits minus the present value of total investment costs. A positive NPV suggests that the proposed bundle would generate a net gain to society and is justifiable in economic terms.

$$NPV = \sum_{t=0}^{N} \frac{(Benefits - Investment\ Costs)_t}{(1+r)^t}$$

Or:

$$NPV = PV(Benefits) - PV(Investment\ Costs)$$

The Benefit/Cost Ratio is the present value of total benefits divided by the present value of total investment costs. A Benefit/Cost Ratio greater than 1.0 suggests that the proposed bundle is justifiable in economic terms.

$$B/C = \frac{PV(Benefits)}{PV(Investment\ Costs)}$$

A variant of this generally-accepted definition of the Benefit/Cost Ratio is also available in the Mosaic tool. In this alternative specification, the Benefit/Cost Ratio is calculated as the present value of total benefits divided by the present value of net agency costs:

$$B/C_{Alt} = \frac{PV(Benefits)}{PV(Net\ Agency\ Costs)}$$

Where Net Agency Costs exclude investment costs borne by other agencies (e.g., through federal grants) and/or costs funded by user fees and other revenue sources.

They are estimated as follows:

$$Net\ Agency\ Costs = Total\ Investment\ Costs - Costs\ Borne\ by\ Other\ Agencies - Incremental\ Agency\ Revenue$$

Note that in this version of the B/C ratio, Incremental Agency Revenue is only included in the denominator (as a negative agency cost), and not in the numerator (as a positive benefit).

The following indicators can be monetized and included in the estimation of total benefits (or total costs) in Version 2.0 of the Mosaic tool:

- Monetized Benefits
    - MO.1 – Travel Time
    - MO.5 – User Costs
    - EV.2 – Changes in Business and Freight Transportation Costs
    - EV.4 – Changes in Productivity from Improved Connectivity (Agglomeration Economies)
    - ES.1 – Emissions of Criteria Air Contaminants
    - ES.3 – Life-Cycle CO2e Emissions
    - FT.3 – Total Operating Revenue
    - SA.1 – Fatal, Injury A, and Injury B (KAB) Crashes
    - SA.2 – Property Damage Only (PDO) Accidents
    - QL.1 – Health Benefits of Active Transportation
    - QL.2 – Quality of the Travel Environment
    - QL.3 – Noise Impacts
- Monetized (Investment) Costs
    - FT.1 – Capital Costs
    - FT.2 – Other Lifecycle Costs

Multi-Objective Decision Analysis Calculations

Multi-Objective Decision Analysis (MODA) is an evaluation methodology that considers multiple objectives jointly, by the attribution of a weight to each measurable objective. In contrast to BCA that focuses on total monetized benefits and costs, MODA allows users to deal with objectives measured in different units and along different scales, which cannot be aggregated through monetization.

The main calculations coded in the Mosaic tool with respect to MODA include the determination of scores (Quantitative Scoring, as defined in the Mosaic Users' Guide) and the estimation of aggregate, weighted scores within and across categories from user-specified weights (Weighting).

Quantitative Scoring with the Mosaic Tool

Users must first define an interval scale over which the scores will be defined. The end-points of the interval scale are parameterized within the Mosaic tool and can be changed from the MODEL

PARAMETERS worksheet (Rows 12 and 13). The default values are -5 and +5, with -5 representing the lowest level of transportation performance, and +5 the highest.

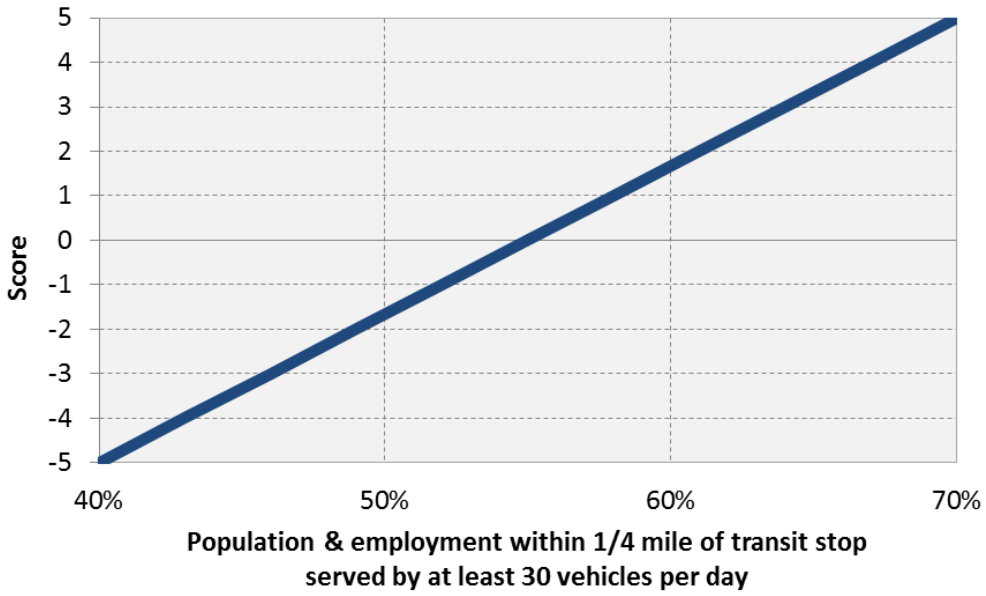
Next, the levels of performance corresponding to the extreme scores of -5 and +5 (or other end-points) must be defined. Two options are available at this stage[1]:

- **Global scoring**, where a score of -5 is assigned to represent the worst level of performance that is likely to be encountered in a comparable planning effort, and +5 to represent the best level; or
- **Local scoring**, where a score of -5 is associated with the performance level of the planning option (bundle) in the currently considered set of options which performs least well, and +5 with that which performs best.

Note: use of the global option would require that a benchmark value be defined for all MODA indicators, which is not yet available for Mosaic Version 2. Therefore, the local scoring option is the recommended approach.

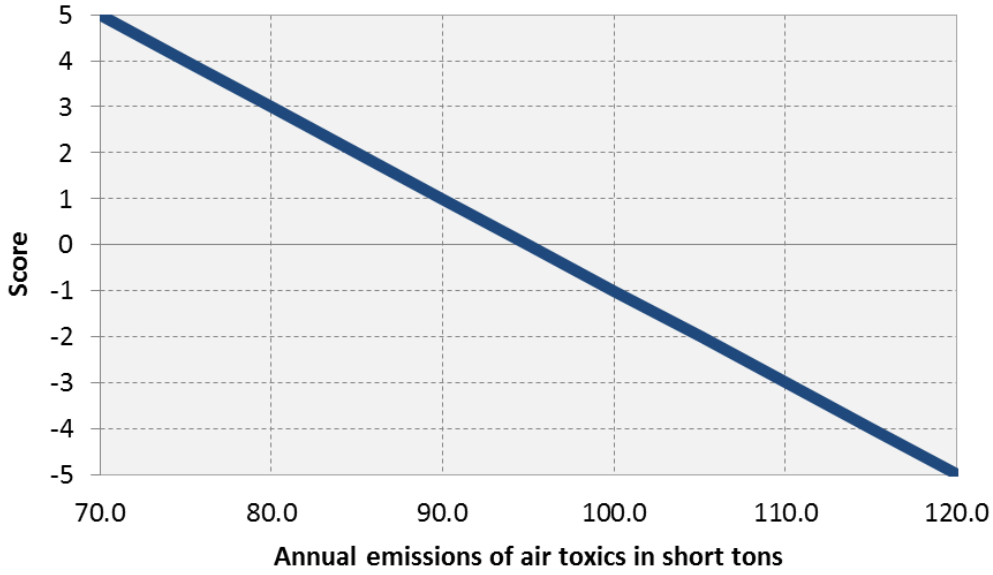
A linear scoring function is then used to translate the estimated value of a given specific indicator into a score along the -5 to +5 scale. The scores are calculated in such a way that the Base Case receives a score of zero for all specific indicators.

This is illustrated for Specific Indicator AC.4 (Population and Employment within ¼ Mile of a Transit Stop Served by at Least 30 Vehicles per Day) in the chart below.



When higher indicator values correspond to worse rather than better performance, the slope of the scoring function is reversed, as illustrated for Specific Indicator ES.2 (Air Toxics) in the figure below.

[1] This description of local and global scoring borrows from CLG (2009), page 42.



The general formula used in the Mosaic tool for calculating a score is:

$$Score_k = Min\ Score + (Performance\ Value_k - Worst\ Performance) * (Max\ Score - Min\ Score) / (Best\ Performance - Worst\ Performance)$$

Where:

- *Score* $_k$ is the score estimated from Bundle *k*;
- *Min Score* is the low end of the interval scale (set to -5 by default);
- *Min Score* is the high end of the interval scale (set to +5);
- *Performance Value* $_k$ is the estimated value of the specific indicator under Bundle *k* (e.g., Air Toxics Emission Volumes);
- *Worst Performance* is the worst level of performance, defined locally or globally; and
- *Best Performance* is the best level of performance, defined locally or globally.

All the scoring functions used in Mosaic are linear. Alternative approaches to scoring are described in CLG (2009), under "Step 4: Assessing performance levels (with scoring)".

Weighting in the Mosaic Tool

Weighted average scores are calculated within the tool based on weighting factors specified by users. Weighted average scores are estimated within each category, and for all categories combined. The general formula for estimating the overall, multi-objective score associated with a bundle is:

$$Overall\ Score_k = w_1 \times Score_{1k} + w_2 \times Score_{2k} + \ldots + w_M \times Score_{Mk}$$

Where:

- *Overall Score* $_k$ is the overall, multi-objective score estimated for Bundle *k*;
- w_i is the weight assigned to Specific Indicator *i*, with *i* = 1..*M* (for a total of *M* specific indicators, selected for MODA); and
- $Score_{ik}$ is the score associated with Specific Indicator i, calculated for Bundle *k*.

Note that the user-specified weights w_i are normalized within the tool, so that:

$$\sum_{i=1}^{M} w_i = 100\%$$

Additionally, when estimating weighted average scores by category, weights within a category (including *P* specific indicators selected for MODA) are normalized so that:

$$\sum_{i=1}^{P} \grave{w}_i = 100\%$$

The normalized weights are estimated as:

$$\grave{w}_i = \frac{w_i}{\sum_{j=1}^{P} w_j}$$

Where:

- $w_{j\ ,\ j = 1..P}$ are the weights assigned to the specific indicators selected for MODA within a given category.

Weighted average scores are reported using the following summary table template:

Category of Transportation Performance	Normalized Weights	Multi-Objective Scores		
		Bundle 1	Bundle 2	…
MOBILITY				
ACCESSIBILITY				
ECONOMIC VITALITY				
ENVIRONMENTAL STEWARDSHIP				
FUNDING / FINANCE				
SAFETY & SECURITY				
LAND USE & GROWTH MANAGEMENT				
QUALITY OF LIFE				
EQUITY				
AGGREGATE MODA SCORE				

Source: Adapted from Mosaic Tool, Version 2.0, OUTPUT TABLES worksheet, Rows 121-131

Scope of MODA within Mosaic

All the specific indicators that can be monetized (see previous section on Benefit-Cost Analysis) can also be scored and included within the calculation of a multi-objective score, but:

- The same indicator cannot be included in both MODA and the BCA within the same Mosaic application; and
- Specific indicators pertaining to investment costs (FT.1 Capital Costs, and FT.2 Other Life-Cycle Costs) must be monetized.

The following indicators can also be included in MODA, through quantitative or qualitative scoring:

MODA-Only Indicators		*Quantitative or Qualitative*	*Qualitative Only*
AC.1	Transportation Cost Index	✓	
AC.2	Population within 45 minutes between work and home	✓	
AC.3	Location of industrial jobs in relation to the regional freight network	✓	
AC.4	Population and employment within ¼ mile of a transit stop served by at least 30 vehicles per day	✓	
AC.5	Amount of multi-use paths and bike boulevards	✓	
AC.6	Sidewalk coverage	✓	
ES.2	Air Toxics	✓	
ES.4	Natural, built, and cultural resources at risk	✓	
FT.4	Share of lifecycle funds that are “new” or “recycled”	✓	
FT.5	Net impact of program on State and Local fiscal balance		✓
SA.3	Emergency Management Systems (EMS) Response Times	✓	
SA.4	Resiliency of the Network		✓
LU.1	Population and employment change and distribution		✓
EQ.1	Distribution of user benefits across population groups		✓
EQ.2	Distribution of PM emissions across population groups		✓
EQ.3	Distribution of health benefits from active transportation across population groups		✓
EQ.4	Distribution of accident rates across population groups		✓

Mobility

Documentation sheets for the following specific indicators can be found in this section:

- MO.1 – Travel Time
- MO.2 – Hours of Congestion
- MO.3 – Reliability (Recurring Delay)
- MO.4 – Reliability (Non-Recurring Delay)
- MO.5 – User Costs
- MO.6 – Mode Split
- MO.7 – VMT per Capita

Category: Mobility
(MOBILITY Worksheet)

General Indicator: Travel Time

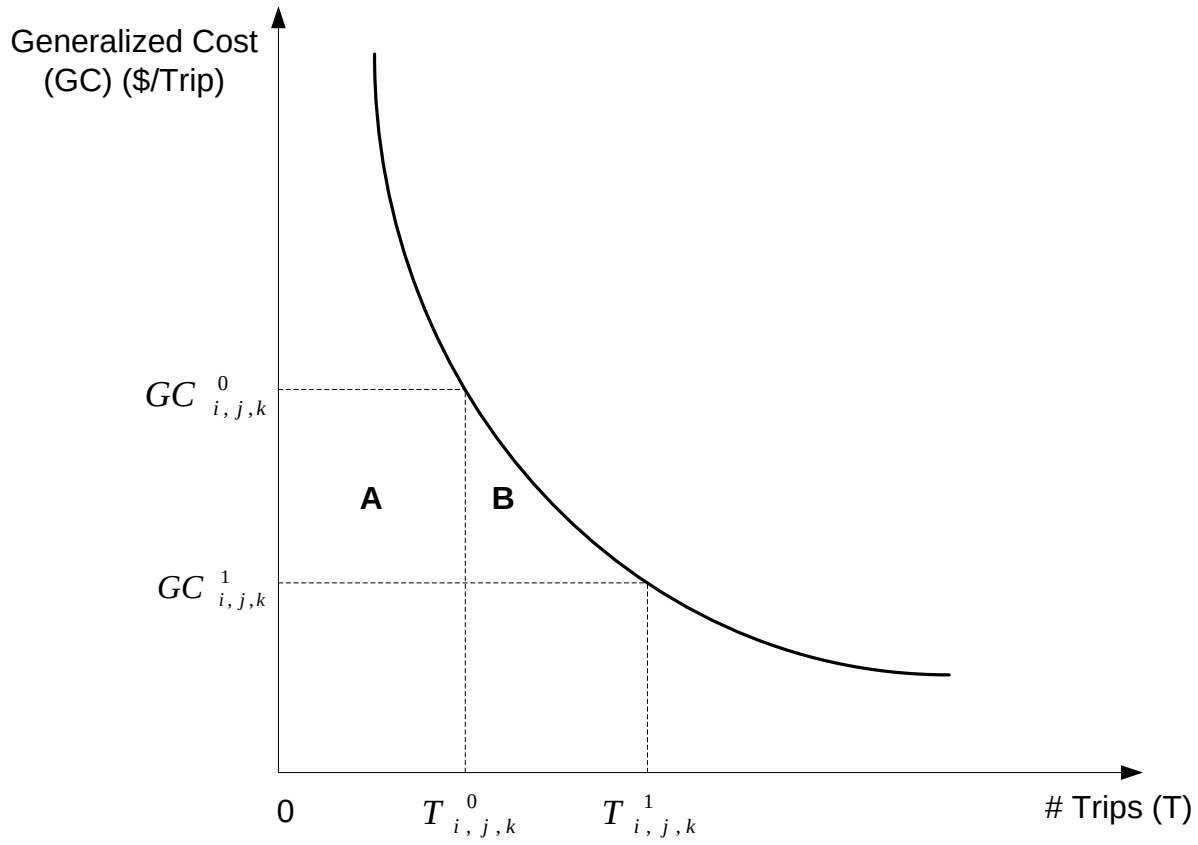
Specific Indicator: MO.1 – Travel Time

This documentation sheet describes the methodology used in the estimation of Specific Indicator MO.1, Travel Time. The indicator is defined as Travel Time Savings relative to the Base Case, and estimated as changes in "consumer surplus." The concept of consumer surplus is defined below.

Conceptual Framework

A transportation project or action typically generates a variety of mobility effects, including reductions in travel times and changes in vehicle operating costs (e.g., fuel consumption). As a result of these changes, users of the transportation network may decide to take more trips, or travel to destinations further away. The change in user benefits – or consumer surplus – resulting from the implementation of a bundle can be illustrated with a graph relating the generalized cost of travel to the demand for travel (i.e., the number of trips or vehicle miles traveled). This relationship is represented by the travel demand curve, illustrated in the figure below.

Figure MO.1.1: The Demand for Travel



Where:

$GC^0_{i,j,k}$ is the generalized cost of travel from origin *i* to destination *j* by mode *k* in the Base Case;

$GC^1_{i,j,k}$ is the generalized cost of travel from origin *i* to destination *j* by mode *k* assuming implementation of a bundle;

$T^0_{i,j,k}$ is the total number of trips from origin *i* to destination *j* by mode *k* in the Base Case; and

$T^1_{i,j,k}$ is the total number of trips from origin *i* to destination *j* by mode *k* assuming implementation of a bundle.

As illustrated in Figure MO.1.1, the travel demand curve is *downward* sloping: as the generalized cost of travel (GC) decreases, the number of trips (T) increases, and vice-versa. The generalized cost of travel refers to the costs associated with travel time, vehicle operations and parking, as well as transit fares or user charges (e.g. tolls).

Specifically, as a result of a bundle, the generalized cost of using mode *k* to travel from *i* to *j* changes from $GC^0_{i,j,k}$ to $GC^1_{i,j,k}$ and the total number of trips by mode *k* between *i* and *j* changes from $T^0_{i,j,k}$ to $T^1_{i,j,k}$.

Area A+B in the graph represents the change in consumer surplus that results from implementation of the bundle. The size of the area is estimated using the "rule of a half", a method widely used in the benefit-cost analysis of transportation investments:[2]

$$\Delta CS_{RoH} = 0.5 \sum_{i,j,k} (T^0_{i,j,k} + T^1_{i,j,k})(GC^0_{i,j,k} - GC^1_{i,j,k})$$

In Mosaic, the "rule of a half" is applied to travel data disaggregated by O-D pair, transportation mode and time-of-day (peak vs. off-peak) to estimate indicator MO.1 (Travel Time Benefits).

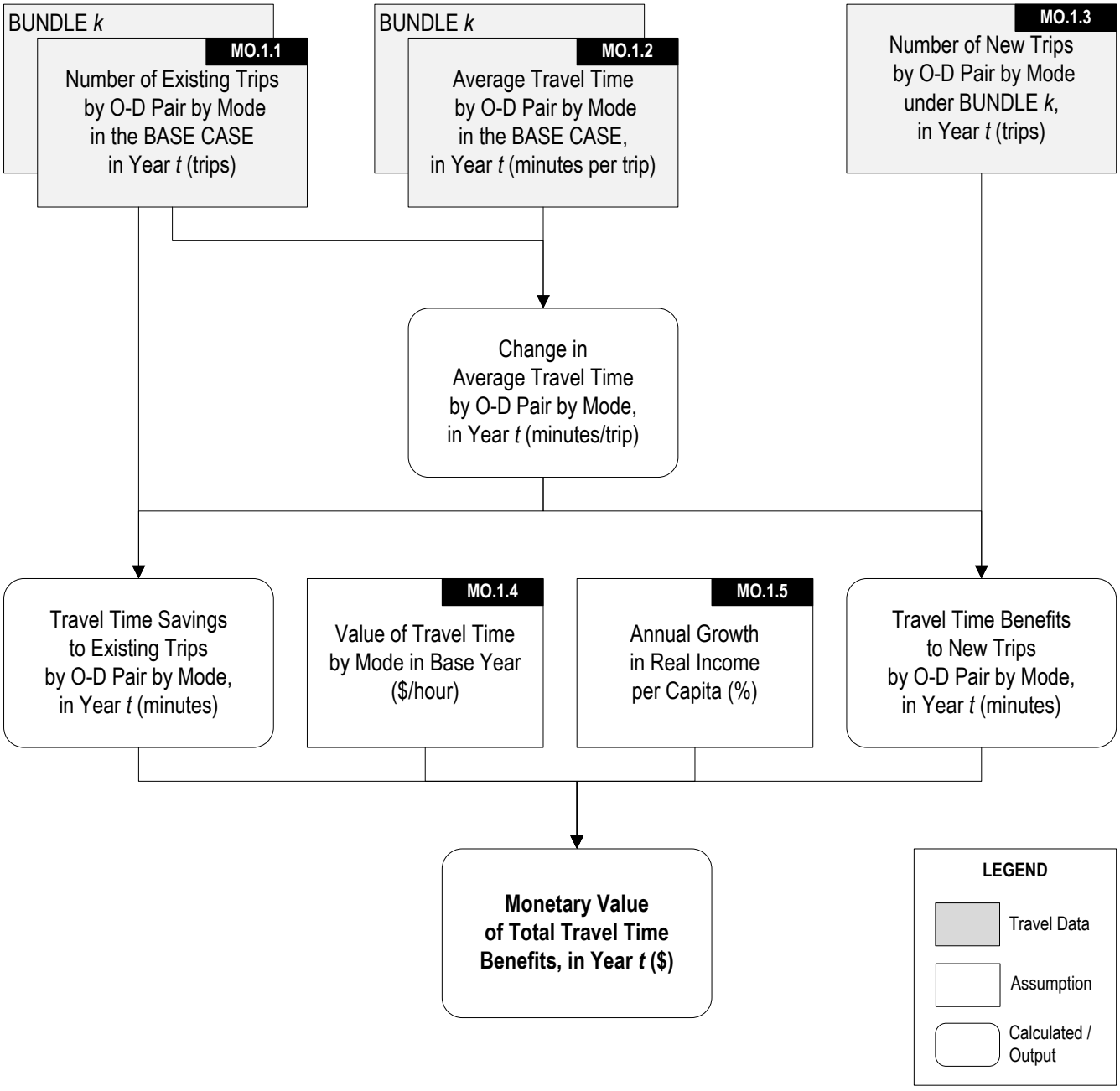
Estimation of Travel Time Benefits

The average travel time (in minutes per trip) for each O-D pair, by mode and time-of-day (peak vs. off-peak) is obtained from the travel demand model, for each bundle and for the Base Case. For the transit modes, this time would also include the average time it takes to walk to a transit stop or station, and the average time spent waiting or transferring. The travel time under each bundle (and any time savings) is monetized by means of a value of time, as specified in the MODEL PARAMETERS worksheet.

The approach to estimating travel time benefits is illustrated in Figure MO.1.2. The source code used in Version 2.0 of the Mosaic tool is provided in Appendix A.

[2] The rule of a half is an approximation to estimating user benefits when the exact shape of the travel demand curve is unknown. The rule assumes that the demand curve is a straight line between the data points describing the Base Case (point T^0GC^0) and the investment case (point T^1GC^1).

Figure MO.1.2: S&L Diagram for Estimating Travel Time Benefits



Data and Assumptions

The table below provides information on the variables used in the estimation of Specific Indicator MO.1. The input values that must be specified by Mosaic users are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table MO.1.1: Data and Assumptions for Estimating Travel Time Benefits

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***MO.1.1**	Number of Existing Trips by O-D Pair by Mode, in Year t	Trips	Yes	Yes	Must be specified by user (loaded as travel demand data)	TRAVEL DATA CALC, Rows 15-25
***MO.1.2**	Average Travel Time by O-D Pair by Mode, in Year t	Minutes per trip	Yes	Yes	Must be specified by user (loaded as travel demand data)	TRAVEL DATA CALC, Rows 60-70 (in vehicle); Rows 75-85 (access or terminal); Rows 90-100 (waiting)
MO.1.3	Number of New Trips by O-D Pair by Mode, in Year t	Trips	Yes	Yes	Calculated from travel demand data	TRAVEL DATA CALC, Rows 15-25 (calculated)
MO.1.4	Value of Travel Time by Mode in Base Year	$ per hour	No	No	Default parameter value provided in tool	MODEL PARAMETERS, Rows 28-54
MO.1.5	Annual Growth in Real Income per Capita	% per year	No	No	Default parameter value provided in tool	MODEL PARAMETERS, Rows 56-58

The values of travel time used in Mosaic are based on guidance developed by the U.S. Department of Transportation, Office of the Secretary. The guidance is available online at https:// www.transportation.gov/office-policy/transportation-policy/guidance-value-time (last accessed November 12, 2014).

The estimates included in Version 2.0 of the Mosaic tool (MODEL PARAMETERS worksheet) are based on income data and other statistics for the United States. Mosaic users can use the value of time calculator provided in the SUPPORTING DATA worksheet (Rows 12 to 106) to develop estimates specific to their study area.

Category: Mobility
(MOBILITY Worksheet)

General Indicator: Travel Time

Specific Indicator:

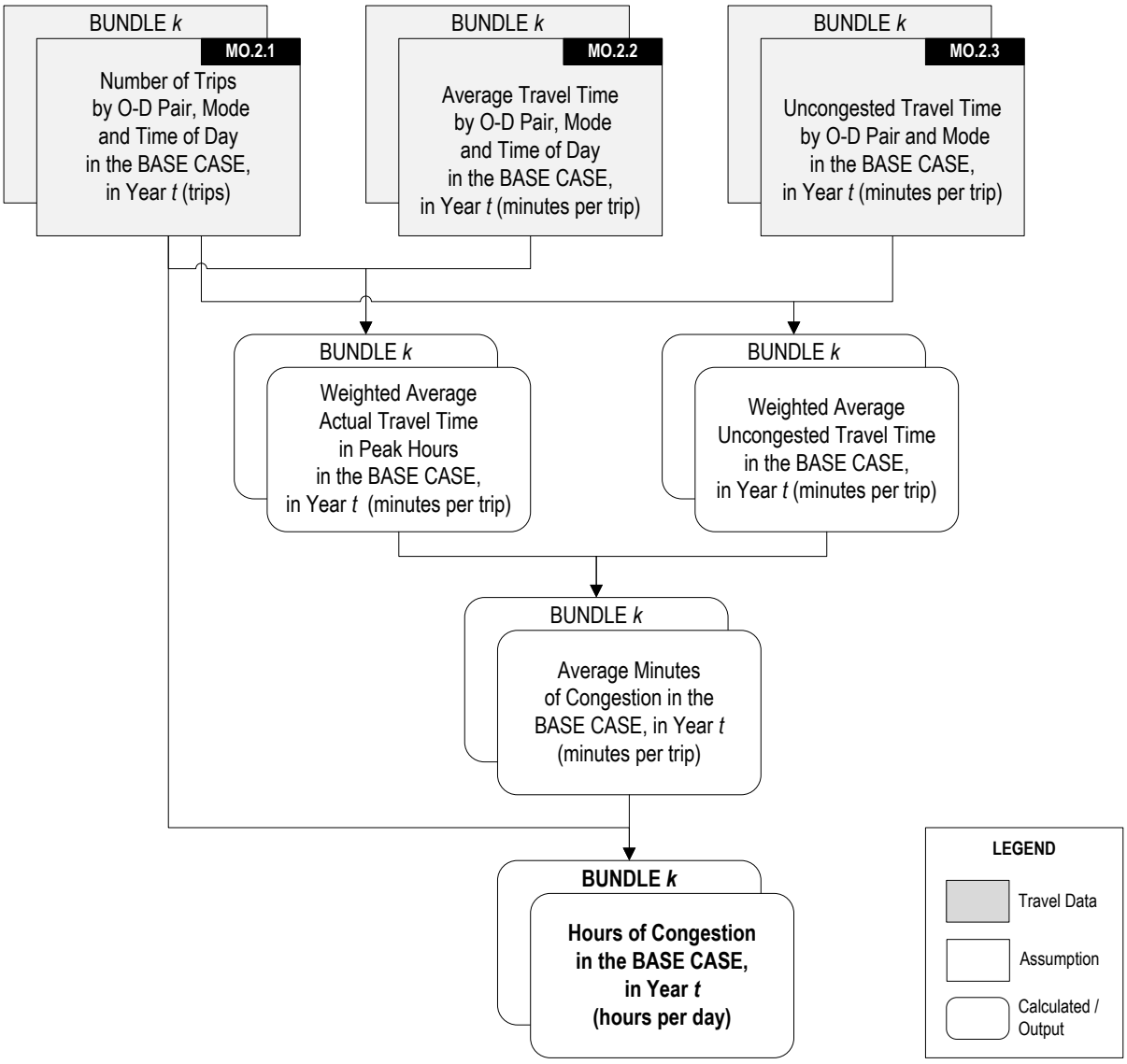
MO.2 – Hours of Congestion

This specific indicator examines the daily hours of congestion for selected modes. Because hours of congestion are a subset of travel time (measured in MO.1), this indicator is only available as Report Only in Mosaic (it cannot be monetized or weighted).

Structure & Logic Diagram

The figure below illustrates how hours of congestion are estimated in Version 2.0 of the Mosaic tool.

Figure MO.2.3: S&L Diagram for Estimating Hours of Congestion



The method illustrated in Figure MO.2.3 relies on data on trip volumes, travel times, and uncongested travel times by mode, origin-destination, and time-of-day (peak vs. off-peak) loaded into the Mosaic tool for the estimation of a number of specific indicators (see Mosaic Users' Guide, Load Travel Data and Travel Data Calculations).

Hours of congestion are estimated as the difference between Actual Travel Times and Uncongested Travel Times during the peak period. The modes considered in the estimation of MO.2 can be specified by the user in the OTHER INPUT DATA worksheet (Rows 15 to 22), as illustrated below.

DEFINITION OF MODES REPRESENTED IN DATA (MAXIMUM OF 8 MODES)		CONGESTION (MO.2) INCLUDED
MODE 1	Drive Alone	1
MODE 2	Drive Passenger	1
MODE 3	Passenger	1
MODE 4	Transit Walk	0
MODE 5	Park & Ride Bus	0
MODE 6	Walk	0
MODE 7	Bike	0
MODE 8	Truck	1

Source: Mosaic Tool, Version 2.0, OTHER INPUT DATA worksheet

Uncongested Travel Times for only one "mode" (i.e., roadway travel) can be loaded into the Mosaic tool for calculations. These travel times are used to estimate hours of congestion for all the modes selected in the OTHER INPUT DATA worksheet (i.e., Drive Alone, Drive Passenger, Passenger, and Truck in the above illustration).

Users may also choose to estimate hours of congestion *outside* the Mosaic tool and enter the results directly into the MOBILITY worksheet. That could be done by comparing peak-hour roadway segment volume-to-capacity ratios to existing multi-hour roadway counts, to compute vehicle hours above the applicable mobility standards provided in the Oregon Highway Plan Mobility Standard Guidelines Report. This report is available online, at http://www.oregon.gov/ODOT/TD/TP/Plans/Mobility.pdf (last accessed October 27, 2014).

Data and Assumptions

Table MO.2.2 on the next page provides information on the input variables used in the estimation of Specific Indicator MO.2. The input values that must be specified by users of Mosaic are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values are entered (Worksheet name in upper case, followed by Row and/or Column number).

Table MO.2.2: Data and Assumptions for Estimating Hours of Congestion

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***MO.2.1**	Number of Trips by O-D Pair, Mode, and Time of Day (peak vs. off-peak) in Year t	Trips	Yes	Yes	Must be specified by user (loaded as travel demand data)	TRAVEL DATA CALC, Rows 15-25 (daily); Rows 30-40 (peak)
***MO.2.2**	Average Travel Time by O-D Pair, Mode, and Time of Day (peak vs. off-peak) in Year t	Minutes per trip	Yes	Yes	Must be specified by user (loaded as travel demand data)	TRAVEL DATA CALC, Rows 60-70 (daily); Peak travel times used in VBA calculations only
***MO.2.3**	Uncongested Travel Time by O-D Pair and Mode, in Year t	Minutes per trip	Yes	Yes	Must be specified by user (loaded as travel demand data)	Used in VBA calculations only

Category: Mobility
(MOBILITY Worksheet)

General Indicator: Quality of Service

Specific Indicator: MO.3 – Reliability (Recurring Delay)

This documentation sheet describes how travel time reliability is assessed in Version 2.0 of the Mosaic tool.

Conceptual Framework

As explained in the SHRP 2 Project C11 technical documentation, travel time reliability relates to how travel times for a given trip and time period perform over time. Reliability can be defined as: 1) the variability in travel times that occur on a facility or a trip over the course of time; or 2) the number of trips that either fail or succeed in accordance with a pre-determined performance standard[3]. In both cases, reliability (or lack thereof) is caused by the interaction of several factors, including fluctuations in demand, traffic incidents, inclement weather, work zones, and physical capacity. These factors will produce travel times that are different from day-to-day for the same trip[4].

Reliability is quantified from the distribution of travel times (for a given facility or trip, and time-of-day) observed over a long period of time (e.g., one year). A variety of metrics can be estimated once the travel time distribution has been established, including standard statistical measures (e.g., standard deviation of travel times), percentile-based measures (e.g., 95th percentile travel time, Buffer Time Index), on-time measures (e.g., percent of trips completed within a travel time threshold), and failure measures (e.g., percent of trips that exceed a travel time threshold)[5].

To help establish the distribution of travel times, a Travel Time Index may be defined. The *mean* Travel Time Index (TTI_m) includes the effects of both recurring and incident delay, and can be expressed as follows:

$$TTI_m = 1 + FFS * (RecurringDelayRate + IncidentDelayRate)$$

Where:

- Free Flow Speed (*FFS*) is the average speed that a motorist would travel if there was no congestion or other adverse condition;
- *RecurringDelayRate* is the rate of recurring congestion, in hours per mile; and
- *IncidentDelayRate* is the average delay due to incidents, in hours per mile.

[3] This section borrows heavily from: Transportation Research Board, Second Strategic Highway Research Program (SHRP 2), Project C11: Reliability Analysis Tool, Technical Documentation and User's Guide, prepared by Cambridge Systematics and Weris, July 2013, page 3

[4] Ibid

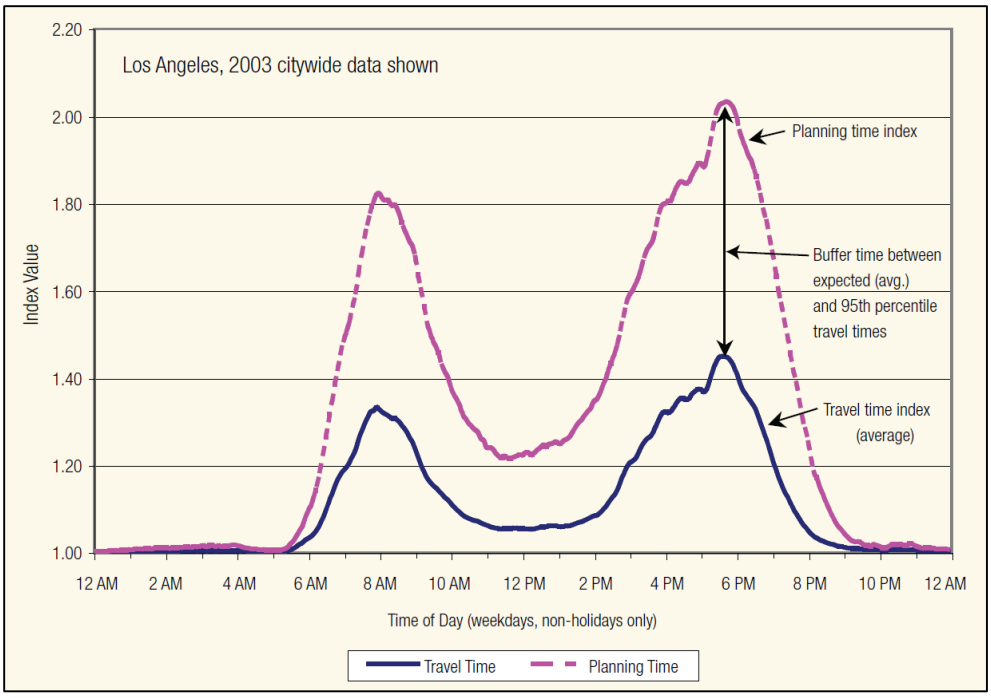
[5] Ibid

Reliability indicators can be estimated by considering different values of the Travel Time Index along its distribution. For example, the Buffer Time Index (BTI) may be calculated as the difference between the 95th percentile of the TTI distribution and the mean or average TTI value, or:

$$BTI = TTI_{95} - TTI_{m}$$

The Planning Time Index (PTI) is typically given by the 95th percentile of the TTI distribution. The BTI and PTI are illustrated in the figure below.

Figure MO.3.4: Reliability Measures Compared to Average Mobility Measures



Source: Federal Highway Administration, Travel Time Reliability Brochure, available at https://ops.fhwa.dot.gov/publications/tt_reliability/brochure/index.htm (last accessed November 6, 2014)

Structure & Logic Diagram

During the development of the Mosaic tool, a methodology for measuring recurring and non-recurring delays separately, at the system level, could not be determined. Therefore, **it is recommended that MO.3 (Reliability – Recurring Delay) and MO.4 (Reliability – Non-recurring Delay) be evaluated with qualitative scoring.**

As an *interim solution*, however, in Version 2.0 of the Mosaic tool, Specific Indicator MO.3 is populated with an estimate of average congestion delay (in minutes per trip), defined as the difference between average travel time and uncongested travel time during peak hours, as measured in the travel data loaded into the tool. It is essentially indicator MO.2 divided by the number of trips in the peak (see Figure MO.2.3). Specific Indicator MO.4 (described in the next documentation sheet) is populated with an estimate of the total buffer time (in minutes per trip), derived from a user-specified Buffer Time Index measure.

Category: Mobility
(MOBILITY Worksheet)

General Indicator: Quality of Service

Specific Indicator:

MO.4 – Reliability (Non-Recurring Delay)

This specific indicator examines the day-to-day variability in travel times due to non-recurring, incident delay associated with collisions, vehicle breakdowns, special events, or extreme weather. It is based on the concept of buffer time.

The Federal Highway Administration (FHWA) defines buffer time as the amount of extra time budgeted by travelers to ensure they reach their destination on-time 95 percent of the time (i.e., late approximately one day per month). The measure is typically divided by the average travel time or travel rate, and expressed as a percentage or index. The Buffer Time Index (BTI) is generally calculated by roadway segment, and a weighted average can be estimated using Vehicle Miles Traveled (VMT) as the weighting factor[6].

The BTI for roadway segment *s* is given by:

$$Buffer\ Time\ Index_s = \frac{95th\ Percentile\ Travel\ Rate_s - Average\ Travel\ Rate_s}{Average\ Travel\ Rate_s}$$

Where the 95th Percentile Travel Rate and Average Travel Rate are measured in miles per minute.

The BTI can also be defined as:

$$Buffer\ Time\ Index_{tr} = \frac{95th\ Percentile\ Travel\ Time_{tr} - Average\ Travel\ Time_{tr}}{Average\ Travel\ Time_{tr}}$$

Where the 95th Percentile and Average Travel Times are measured in minutes, for a given trip *tr*.

Structure & Logic Diagram

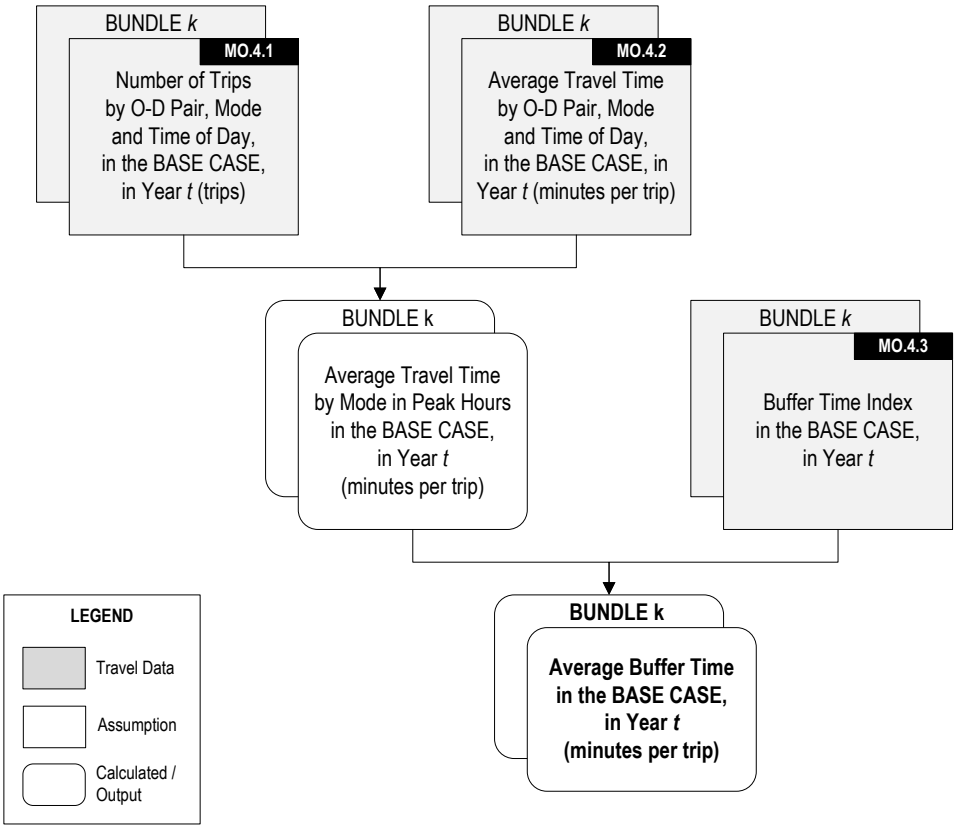
Figure MO.4.5 on the next page illustrates the estimation of indicator MO.4 in Version 2.0 of the Mosaic tool.

Estimates of the weighted average Buffer Time Index (in percent) must be developed *outside* the Mosaic tool and entered as input values, for the Base Case and each bundle being assessed and for at least one forecast year.

Within the tool, the index is simply combined with estimates of average travel time developed from travel data (See Specific Indicator MO.1) to arrive at an estimate of added time per trip (average buffer time, in minutes per trip). This estimate can then be used as a basis for scoring.

[6] See FHWA, https://ops.fhwa.dot.gov/congestion_report_04/appendix_C.htm (last accessed November 6, 2014)

Figure MO.4.5: S&L Diagram for Estimating Reliability Benefits



Data and Assumptions

The table below provides information on the input variables used in the estimation of Specific Indicator MO.4. The input values that must be specified by Mosaic users are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table MO.4.3: Data and Assumptions for Estimating Reliability Benefits

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***MO.4.1**	Number of Trips by O-D Pair, Mode, and Time of Day (peak vs. off-peak) in Year t	Trips	Yes	Yes	Must be specified by user (loaded as travel demand data)	TRAVEL DATA CALC, Rows 15-25 (daily); Rows 30-40 (peak)
***MO.4.2**	Average Travel Time by O-D Pair, Mode, and Time of Day (peak vs. off-peak) in Year t	Minutes per trip	Yes	Yes	Must be specified by user (loaded as travel demand data)	TRAVEL DATA CALC, Rows 60-70 (daily); Peak travel times used in VBA calculations only
***MO.4.3**	Buffer Time Index in Year t	Percent	Yes	Yes	Must be specified by user	OTHER INPUT DATA, Rows 112-122

Category: Mobility
(MOBILITY Worksheet)

General Indicator: Out-of-Pocket Costs

Specific Indicator:

MO.5 – User Costs

This specific indicator examines the changes in out-of-pocket costs (e.g., fuel consumption and other vehicle operating costs, transit fares and tolls) incurred by users of the transportation system as a result of a bundle. Changes in user costs are estimated in relation to the Base Case.

Structure & Logic Diagram

Figure MO.5.6 on the next page provides a graphical representation of the main variables and calculations used in the development of MO.5.

Most variables are populated with data from a travel demand model (boxes shaded in light gray in the S&L diagram). Estimates of average out-of-pocket cost, in dollars per vehicle-mile or person-mile, by mode and for a least one forecast year, must be specified by the user.

The Mosaic tool includes a series of input tables where this information can be entered. One such table is reproduced below, for illustration. Users should refer to the Mosaic Users' Guide for further instructions on these tables.

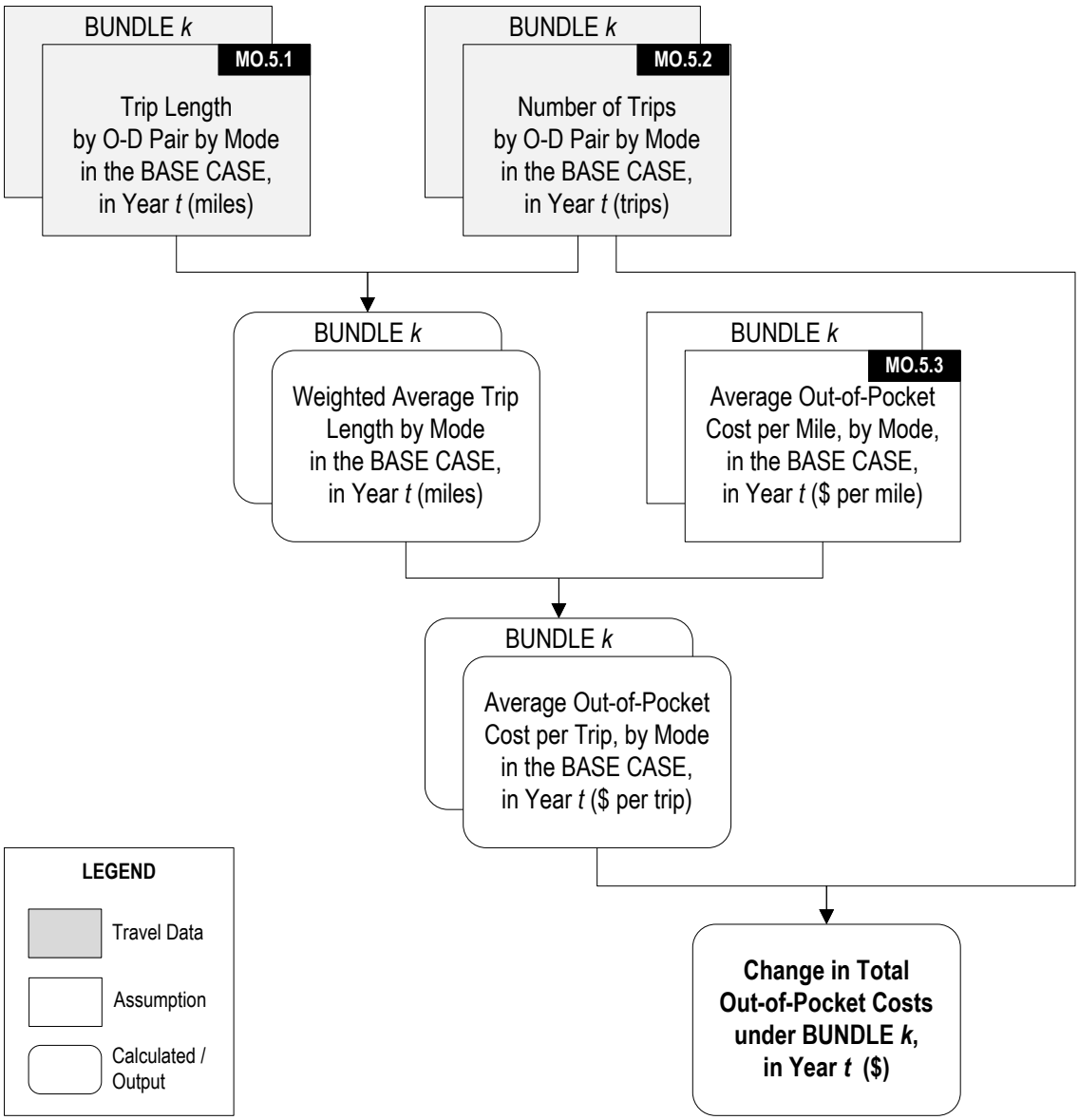
AVERAGE USER COST PER MILE , $ PER VEHICLE OR PERSON MILE, COSTS BORNE BY USERS ONLY		MODE 1 Drive Alone	MODE 2 Drive Passenger	MODE 3 Passenger	MODE 4 Transit Walk	MODE 5 Park & Ride Bus	MODE 6 Walk	MODE 7 Bike	MODE 8 Truck
		Person	Person	Person	Person	Person	Person	Person	Vehicle
2010	Current Conditions	$0.42	$0.42	$0.00	$0.18	$0.29	$0.00	$0.00	$1.43
Notes:	Auto and truck costs based on ODOT 2010 Transit Walk costs based on $1.09 per boarding and average trip length of 6.11 miles (based on Travel Demand Model output) Park & Ride Bus assumes in-bus distance of 6.11 miles, with remainder of distance (10.88 - 6.11) in personal vehicle, drive alone								
2035	Modeled Year 1								
Base Case	2035 Low Build	$0.42	$0.42	$0.00	$0.19	$0.30	$0.00	$0.00	$1.43
Bundle_1	Roadway & Capacity	$0.42	$0.42	$0.00	$0.19	$0.30	$0.00	$0.00	$1.43
Bundle_2	Transit	$0.42	$0.42	$0.00	$0.18	$0.28	$0.00	$0.00	$1.43
Bundle_3	Active Transport & Programs	$0.42	$0.42	$0.00	$0.18	$0.29	$0.00	$0.00	$1.43
Bundle_4	n/a								

Source: Mosaic Tool, Version 2.0, OTHER INPUT DATA worksheet

To allow comparison across modes, all estimates of user costs must be entered in dollars per person-mile. Estimates expressed in dollars per vehicle-mile are converted to dollars per person-mile using the estimate of cost per vehicle-mile and a mode-specific average vehicle occupancy rate, as follows:

Average User Cost per Person-Mile = Average User Cost per Vehicle-Mile / Average Vehicle Occupancy *(persons per vehicle)*

Figure MO.5.6: S&L Diagram for Estimating User Costs



Data and Assumptions

The table on the next page provides information on the input variables used in the estimation of Specific Indicator MO.5. The input values that must be specified by Mosaic users are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table MO.5.4: Data and Assumptions for Estimating User Costs

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***MO.5.1**	Trip Length by O-D Pair by Mode, in Year t	Miles	Yes	Yes	Must be specified by user (loaded as travel demand data)	TRAVEL DATA CALC, Rows 45-55
***MO.5.2**	Number of Trips by O-D Pair by Mode, in Year t	Trips	Yes	Yes	Must be specified by user (loaded as travel demand data)	TRAVEL DATA CALC, Rows 15-25
***MO.5.3**	Average Out-of-Pocket Cost per Mile, by Mode, in Year t	$ per mile	Yes	Yes/No	Must be specified by user	OTHER INPUT DATA, Rows 66, 72-82, 85-95, 98-108

Category: Mobility
(MOBILITY Worksheet)

General Indicator: Travel Characteristics

Specific Indicator:

MO.6 – Mode Split

This specific indicator examines the distribution of total trips by mode. Up to eight modes can be specified in Mosaic. Users can define indicator MO.6 by specifying which mode(s) will be included in the calculations, and how.

For example, if MO.6 is defined as the share of Single Occupancy Vehicles in total personal trips, users would enter the following information in the OTHER INPUT DATA worksheet (Rows 15 to 22):

DEFINITION OF MODES REPRESENTED IN DATA (MAXIMUM OF 8 MODES)		MODE SPLIT (MO.6)	
		NUM.	DENOM.
MODE 1	Drive Alone	1	1
MODE 2	Drive Passenger	0	1
MODE 3	Passenger	0	1
MODE 4	Transit Walk	0	1
MODE 5	Park & Ride Bus	0	1
MODE 6	Walk	0	1
MODE 7	Bike	0	1
MODE 8	Truck	0	0

Source: Mosaic Tool, Version 2.0, OTHER INPUT DATA worksheet

If, on the other hand, users wish to define MO.6 as the share of all trips made using public transportation in total personal trips, they would enter:

DEFINITION OF MODES REPRESENTED IN DATA (MAXIMUM OF 8 MODES)		MODE SPLIT (MO.6)	
		NUM.	DENOM.
MODE 1	Drive Alone	0	1
MODE 2	Drive Passenger	0	1
MODE 3	Passenger	0	1
MODE 4	Transit Walk	1	1
MODE 5	Park & Ride Bus	1	1
MODE 6	Walk	0	1
MODE 7	Bike	0	1
MODE 8	Truck	0	0

Source: Mosaic Tool, Version 2.0, OTHER INPUT DATA worksheet

Structure & Logic Diagram

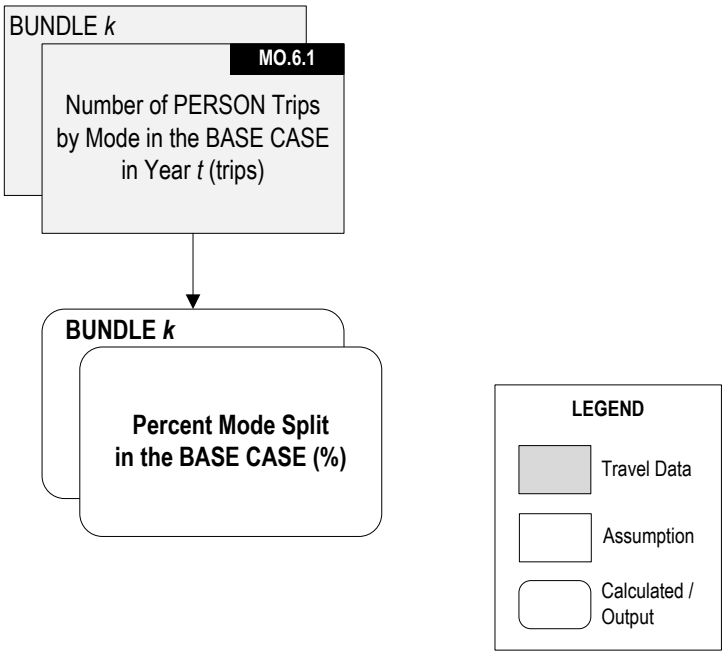
The estimation of the Mode Split indicator is illustrated in Figure MO.6.7, below. The general formula for calculating MO.6 is:

$$Mode\ Split = \frac{\sum_i^P Person\ Trips_i}{\sum_j^N Person\ Trips_j}$$

Where:

- *Person Trips* $_i$ is the number of annual trips by mode *i*, for all *P* modes selected to enter the numerator of the fraction; and
- *Person Trips* $_j$ is the number of annual trips by mode *j*, for all *N* modes selected to enter the denominator of the fraction.

Figure MO.6.7: S&L Diagram for Estimating Mode Split



Data and Assumptions

The table below provides information on the input variables used in the estimation of Specific Indicator MO.6. The input values that must be specified by Mosaic users are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table MO.6.5: Data and Assumptions for Estimating Mode Split

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***MO.6.1**	Number of Person Trips by Mode, in Year t	Trips	Yes	Yes	Must be specified by user (loaded as travel demand data)	TRAVEL DATA CALC, Rows 15-25

Category: Mobility
(MOBILITY Worksheet)

General Indicator: Travel Characteristics

Specific Indicator:

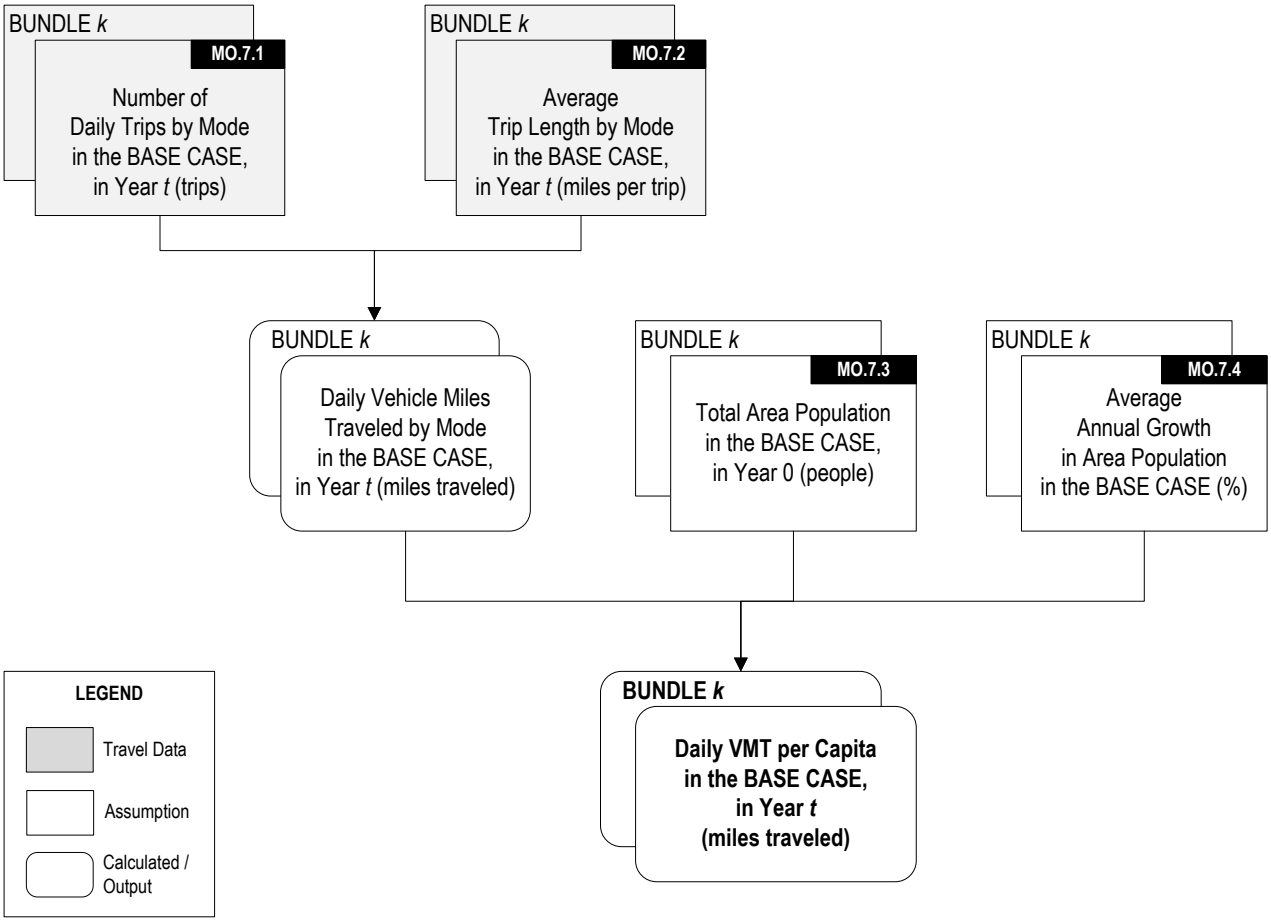
MO.7 – VMT per Capita

This specific indicator is calculated as the ratio of total Daily Vehicle Miles Traveled within a study area to the total population residing in the area.

Structure & Logic Diagram

The estimation of Daily VMT per capita is illustrated in the figure below.

Figure MO.7.8: S&L Diagram for Estimating VMT per Capita



Mosaic users can decide which modes of transportation are used in the calculation of Daily VMT. Thus, in the example below (reproduced from the OTHER INPUT DATA worksheet of the Mosaic tool), indicator MO.7 would be estimated as total Auto VMT (sum of Drive Alone + Drive Passenger + Park & Ride Bus + Truck) divided by total population.

DEFINITION OF MODES REPRESENTED IN DATA (MAXIMUM OF 8 MODES)		VHT & VMT (MO.7)
MODE 1	Drive Alone	1
MODE 2	Drive Passenger	1
MODE 3	Passenger	0
MODE 4	Transit Walk	0
MODE 5	Park & Ride Bus	1
MODE 6	Walk	0
MODE 7	Bike	0
MODE 8	Truck	1

Source: Mosaic Tool, Version 2.0, OTHER INPUT DATA worksheet

Data and Assumptions

Table MO.7.6 provides information on the input variables used in the estimation of Specific Indicator MO.7. The input values that must be specified by Mosaic users are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table MO.7.6: Data and Assumptions for Estimating VMT per Capita

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***MO.7.1**	Number of Daily Trips by Mode, in Year t	Trips	Yes	Yes	Must be specified by user (loaded as travel demand data)	TRAVEL DATA CALC, Rows 15-25
***MO.7.2**	Average Trip Length by Mode, in Year t	Miles	Yes	Yes	Must be specified by user (loaded as travel demand data)	TRAVEL DATA CALC, Rows 45-55
***MO.7.3**	Total Area Population in Year 0	Persons	Yes	No	Must be specified by user	OTHER INPUT DATA, Rows 146-156, Columns D-G
***MO.7.4**	Average Annual Growth in Area Population	%	Yes	Yes/No	Must be specified by user	OTHER INPUT DATA, Rows 146-156, Column H

Accessibility

Documentation sheets for the following specific indicators can be found in this section:

- AC.1 – Transportation Cost Index
- AC.2 – Population within 45 Minutes between Work and Home
- AC.3 – Location of Industrial Jobs in Relation to the Regional Freight Network
- AC.4 – Population and Employment within ¼ Mile of a Transit Stop Served by at Least 30 Vehicles a Day
- AC.5 – Amount of Multi-Use Paths and Bike Boulevards
- AC.6 – Sidewalk Coverage

Category: Accessibility
(ACCESSIBILITY Worksheet)

General Indicator: Proximity

Specific Indicator:

AC.1 – Transportation Cost Index

The Transportation Cost Index (TCI) measures the relative cost of accessing goods, services, and daily activities using various transportation modes. The concept is analogous to the Consumer Price Index, where the generalized cost of a basket of trips (representing different modes, geographies, and trip purposes) is estimated under different planning options.

As noted in the Mosaic Users' Guide, as of November 2014, the TCI is not yet ready for use. Users of Mosaic are advised to not use this indicator until such time that an estimation method is available. The rest of this documentation sheet describes emerging ideas and concepts.

Structure & Logic Diagram

Figure AC.1.9 on the next page provides a graphical representation of the main variables and calculations which might be required to estimate Specific Indicator AC.1. The variables involved in operations that would be performed *outside* the Mosaic workbook are represented in boxes with dotted lines. Development of the TCI is likely to involve three main steps.

1. **Define a market basket of travel destinations**

   - Identify the trip purposes (work, shopping, recreation, etc.) for which baskets of travel destinations will be defined.
   - Identify a market area that will serve as the reference for quantifying travel destinations: a reference Traffic Analysis Zone (TAZ) within an urban area, and a set of zones located around the TAZ that represent a large number of destinations.
   - Calculate the total number of travel destinations (or attractions) within the reference market area, for each trip purpose and for different income groups.

2. **Calculate travel costs to access the market basket**

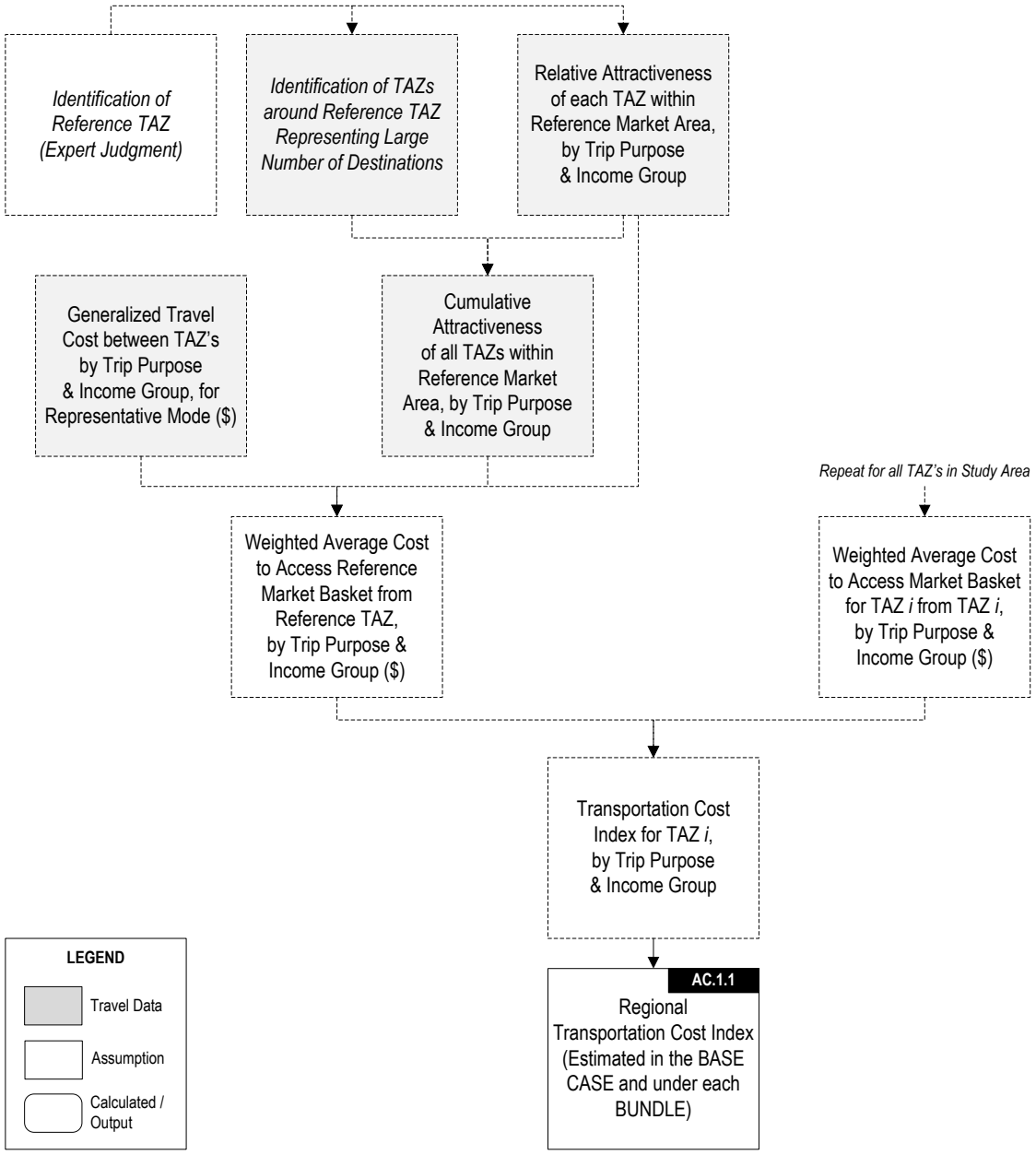
   - Calculate a weighted average cost to access each market basket, across all modes, for each TAZ and income group within the study area. The weighting factors used in the estimation of average cost should be based on the proportion of the market basket that is located within each TAZ in the market place (i.e., TAZ attractiveness divided by cumulative attractiveness).

3. **Compute transportation cost indices**

   - Calculate TCI values for each cost array by dividing the values for each TAZ by the values for the reference TAZ.

This procedure would produce TCI values by TAZ and income group, by TAZ and trip purpose, and by TAZ for all income groups and trip purposes. Additional information on developing Specific Indicator AC.1 can be found in the Transportation Planning Performance Measures, Final Report SPR 357 (October 2005), pp. 25-27.

Figure AC.1.9: S&L Diagram for Estimating the Transportation Accessibility Index



Data and Assumptions

This specific indicator (Transportation Cost Index) would be estimated *outside* the Mosaic workbook, for each bundle and for at least one forecast year. The value of the indicator would then be entered in the relevant table of the ACCESSIBILITY worksheet (Rows 12 to 22). Alternatively, a qualitative proximity score could be directly assigned to each bundle.

Category: Accessibility
(ACCESSIBILITY Worksheet)

General Indicator: Proximity

Specific Indicator:

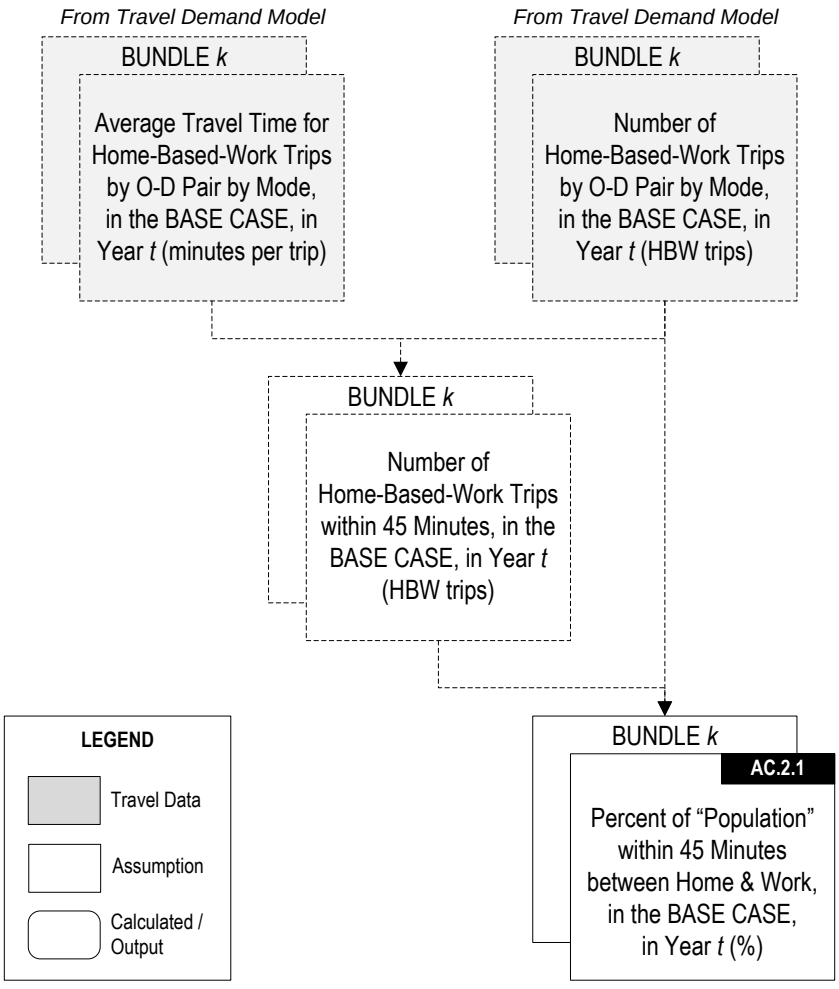
AC.2 – Population within 45 Minutes between Work and Home

This specific indicator measures the percentage of the population of a study area that is able to travel between home and work within a "reasonable" time. The definition of "reasonable" is expected to vary across study areas, and should be determined by Mosaic users. It is set to 45 minutes as a default value in Version 2.0 of Mosaic.

Structure & Logic Diagram

The figure below provides a graphical representation of the main variables and calculations required to develop Specific Indicator AC.2. The variables involved in operations performed *outside* the Mosaic tool are represented in boxes with dotted lines.

Figure AC.2.10: S&L Diagram for Estimating the Share of Population within 45 Minutes between Work and Home



An alternative approach to estimating this indicator is available in the Mosaic Users' Guide. In this approach, travel time contours are developed to determine the distance by which travelers may go within an established "reasonable time." Users then calculate the percentage of trips by origin TAZ that are within this contour and report out that percentage.

Data and Assumptions

This specific indicator (Population within 45 Minutes between Work and Home) must be estimated *outside* the Mosaic workbook, for each bundle and for at least one forecast year. The value of the indicator must then be entered in the relevant table of the ACCESSIBILITY worksheet (Rows 43 to 53).

Alternatively, a qualitative proximity score can be assigned to each bundle, using a scale of -5 to +5. Additional guidance on the estimation of AC.2 is available in the Mosaic Users' Guide (Step 4: Populating the Mosaic Tool).

Category: Accessibility
(ACCESSIBILITY Worksheet)

General Indicator: Connectivity/Ease of Connections

Specific Indicator:

AC.3 – Location of Industrial Jobs in Relation to the Regional Freight Network

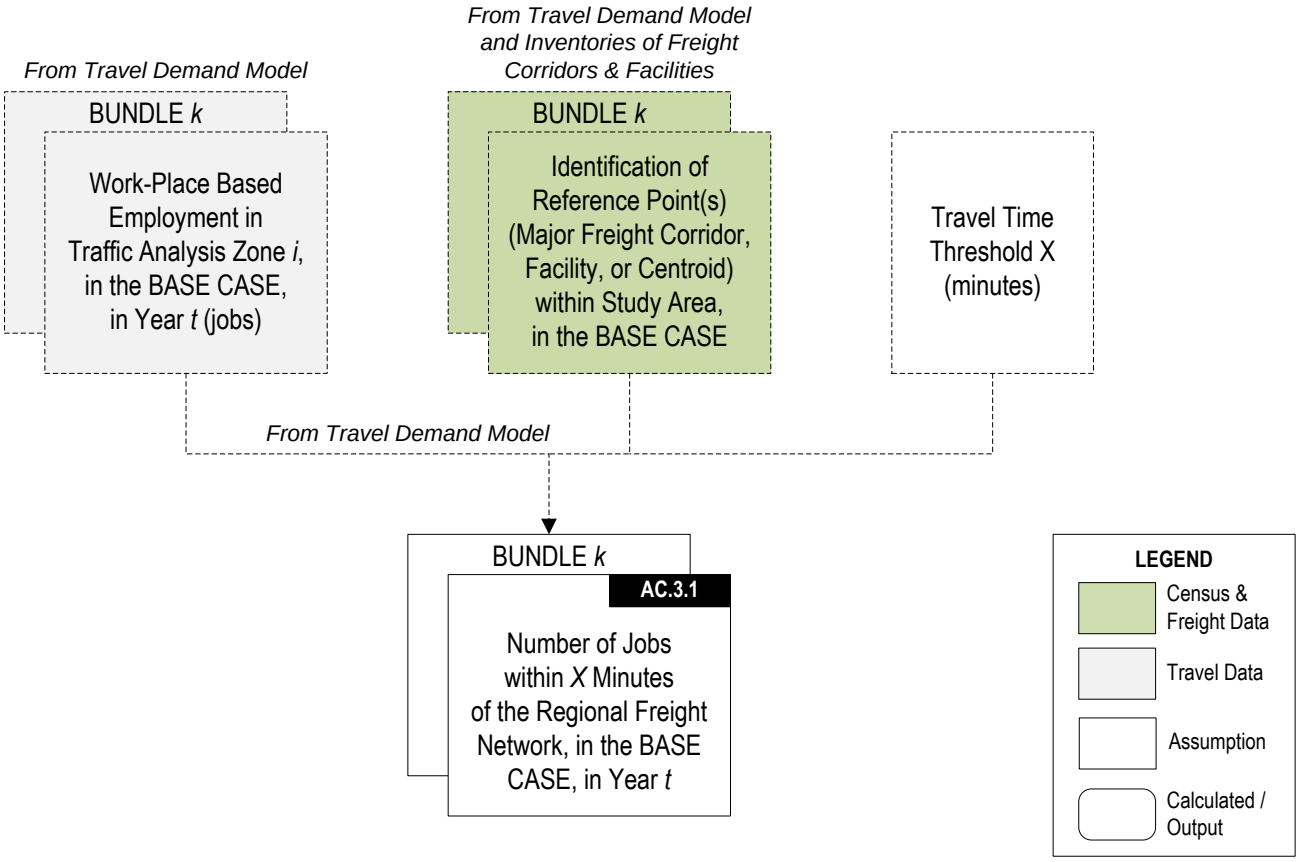
This specific indicator measures the number of industrial jobs located within a certain distance or travel time (to be determined by the user) from the regional freight network. As explained in the Mosaic Users' Guide, this indicator can be defined and measured in two slightly different ways:

- **Option 1:** Number of Jobs within X minutes of the Regional Freight Network. Under this first option, users of Mosaic would have to:
    - Identify specific parcels or TAZ centroids (e.g., a key intersection in an industrial district or a key employment center) that would serve as reference point(s); and
    - Use a travel demand model to determine the number of jobs available within a certain travel time (X minutes) from that parcel or centroid, under the Base Case and each bundle.
- **Option 2:** Number of Industrial Jobs within X miles of the Regional Freight Network. Under this alternative definition, users would have to:
    - Obtain GIS layers identifying the regional freight network and the location of jobs by labor classification (i.e., workplace-based employment in industrial sectors);
    - Define how distance between industrial jobs and the regional freight network are measured (e.g., straight-line distance between a given freight facility and the centroid of a major employment center, actual driving distance between TAZ centroids); and
    - Run a query in GIS to determine the number of industrial jobs within a given spatial distance (X miles) of the network, under the Base Case and each bundle.

Structure & Logic Diagram

The figure below provides a graphical representation of the main variables and calculations required to develop Specific Indicator AC.3 as defined under Option 1 (Number of Jobs within X minutes of the Regional Freight Network). The variables involved in operations performed *outside* the Mosaic workbook are represented in boxes with dotted lines.

Figure AC.3.11: S&L Diagram for Estimating the Location of Jobs in Relation to the Regional Freight Network



Data and Assumptions

This specific indicator (Location of Industrial Jobs in Relation to the Regional Freight Network) must be estimated *outside* the Mosaic workbook, for each bundle and for at least one forecast year. The value of the indicator must then be entered in the relevant table of the ACCESSIBILITY worksheet (Rows 70-80). Alternatively, a qualitative connectivity score can be assigned directly to each bundle, using a scale of -5 to +5.

Category: Accessibility
(ACCESSIBILITY Worksheet)

General Indicator: Modal Availability

Specific Indicator:

AC.4 – Population and Employment within ¼ Mile of a Transit Stop Served by at Least 30 Vehicles per Day

This specific indicator is the share of the total population and employment of a study area within a quarter mile of a transit stop served by at least 30 vehicles per day. As noted in the Mosaic Users' Guide, the maximum distance to a transit stop (1/4 mile) may be modified by the user.

Structure & Logic Diagram

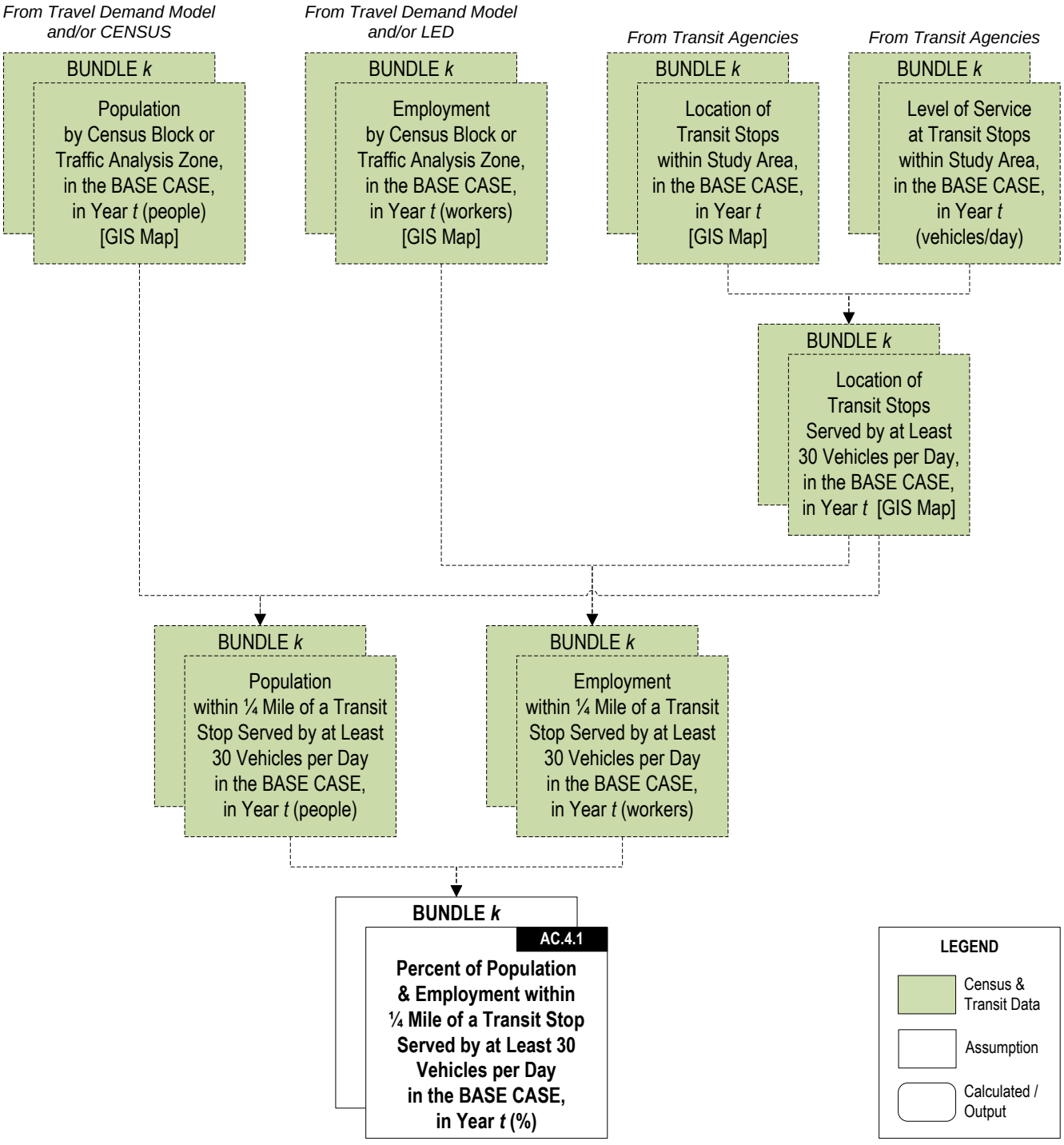
Figure AC.4.12 on the next page provides a graphical representation of the main variables and calculations required to develop Specific Indicator AC.4. The variables involved in operations performed *outside* the Mosaic tool are represented in boxes with dotted lines.

Additional guidance to develop this indicator can be found in the Mosaic Users' Guide (Step 4: Population the Mosaic Tool).

Data and Assumptions

This specific indicator must be estimated outside the Mosaic workbook, for each bundle and for at least one forecast year. The value of the indicator must then be entered in the relevant table of the ACCESSIBILITY worksheet (Rows 97-107). Alternatively, a qualitative modal availability score can be assigned directly to each bundle.

Figure AC.4.12: S&L Diagram for Estimating Population and Employment within ¼ Mile of a Transit Stop Served by at Least 30 Vehicles per Day



Category: Accessibility
(ACCESSIBILITY Worksheet)

General Indicator: Modal Availability

Specific Indicator:

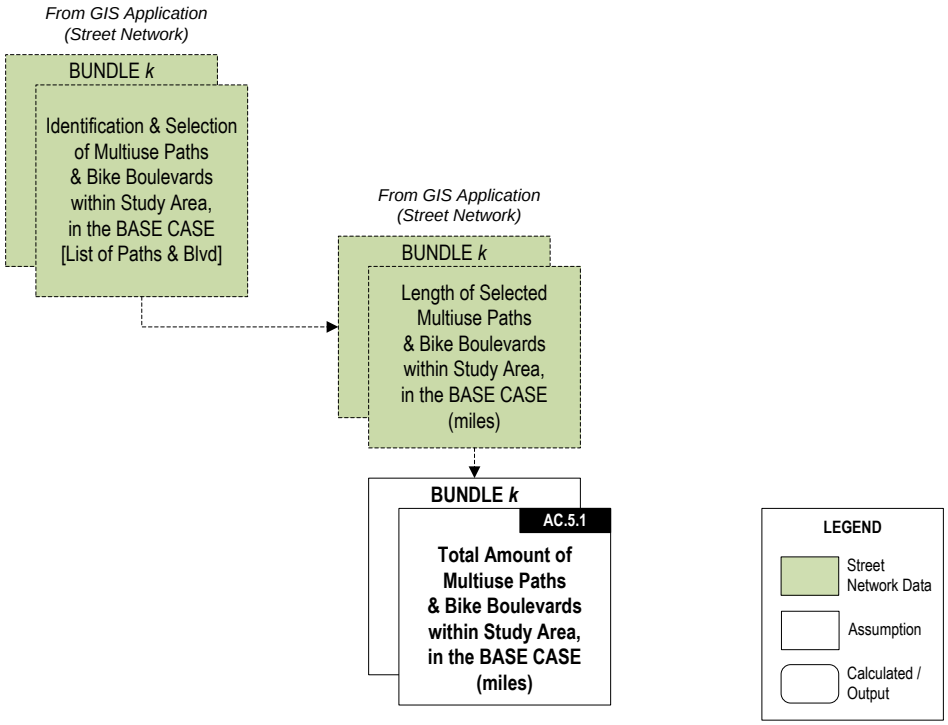
AC.5 – Amount of Multiuse Paths and Bike Boulevards

This specific indicator determines the accessibility of the network of bicycle facilities (such as multiuse paths, bike lanes, and boulevards) as a measurement for the availability of bicycling as a modal option. It is estimated as the total lane mileage of multiuse paths and bike boulevards under various planning options. Alternatively, the indicator can be expressed as an index, with a value of 100 in the Base Case.

Structure & Logic Diagram

The figure below provides a graphical representation of the main variables and calculations required to develop Specific Indicator AC.5. The variables involved in operations performed *outside* the Mosaic tool are represented in boxes with dotted lines.

Figure AC.5.13: S&L Diagram for Estimating the Amount of Multi-Use Paths and Bike Boulevards



Data and Assumptions

This specific indicator (Amount of Multiuse Paths and Bike Boulevards) must be estimated *outside* the Mosaic workbook, for each bundle and for at least one forecast year. The value of the indicator must then be entered in the relevant table of the ACCESSIBILITY worksheet (Rows 124 to 134). Alternatively, a qualitative score can be assigned directly to each bundle.

Category: Accessibility
(ACCESSIBILITY Worksheet)

General Indicator: Modal Availability

Specific Indicator:

AC.6 – Sidewalk Coverage

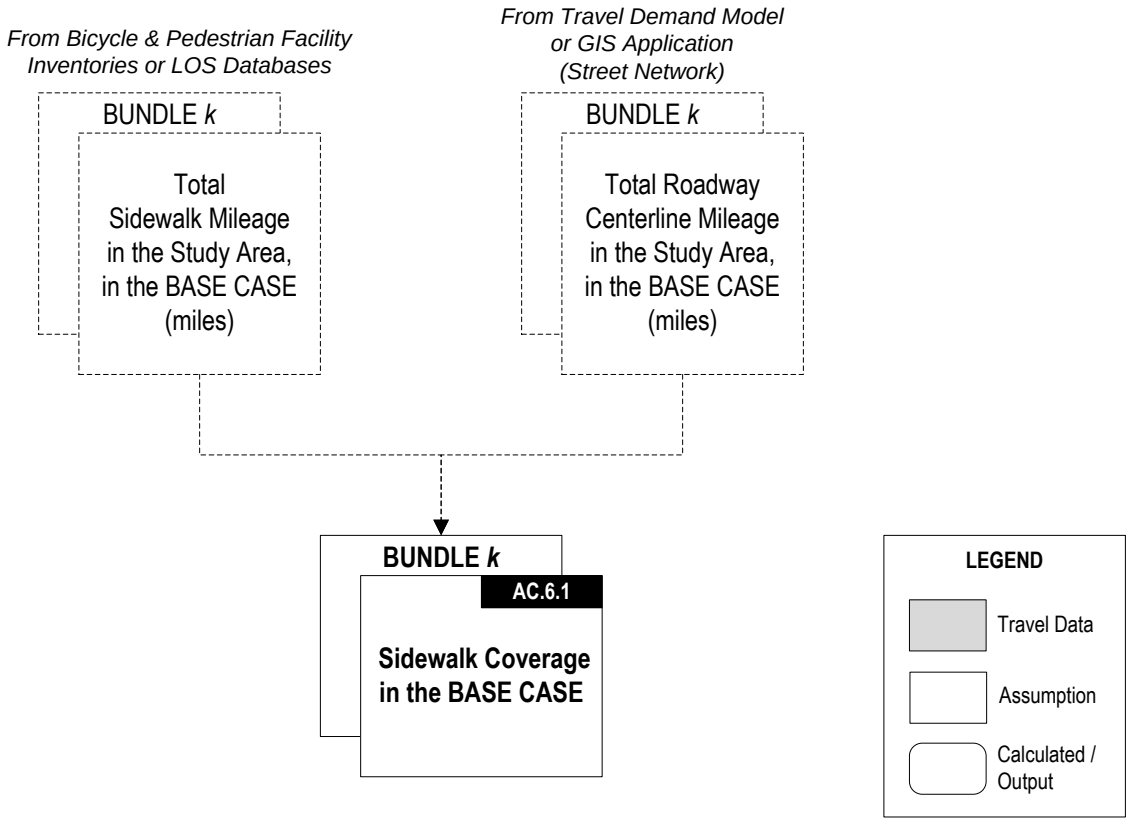
This specific indicator measures the network of pedestrian facilities (such as sidewalks and paths) as an indicator of pedestrian modal availability.

In Version 2.0 of the Mosaic tool, a single measure (Sidewalk Coverage, as defined below) is considered for this indicator. Synthetic measures combining sidewalks, paths and the number of marked street-crossings (either at intersections or in the middle of a block) may be developed in future versions.

Structure & Logic Diagram

The figure below provides a graphical representation of the main variables and calculations required to develop Specific Indicator AC.6. The variables involved in operations performed *outside* the Mosaic tool are represented in boxes with dotted lines.

Figure AC.6.14: S&L Diagram for Estimating Sidewalk Coverage



Specific indicator AC.6 can be estimated with the following equation:

Sidewalk Coverage = Total Sidewalk Mileage / Total Roadway Centerline Mileage

Where:

- *Total Sidewalk Mileage* is the total, cumulative length of sidewalks (in miles) within the boundaries of the study area; and
- *Total Roadway Centerline Mileage* is the total, cumulative length of roadway centerline within the study area.

Data and Assumptions

This specific indicator must be estimated *outside* the Mosaic workbook, for each bundle and for at least one forecast year. The value of the indicator must then be entered in the relevant table of the ACCESSIBILITY worksheet (Rows 151 to 161). Alternatively, a qualitative modal availability score can be assigned directly to each bundle.

Economic Vitality

Documentation sheets for the following specific indicators can be found in this section:

- EV.1 – Number of Jobs Created or Retained by Bundle
- EV.2 – Changes in Business Travel and Freight Transportation Costs by Industry
- EV.3 – Changes in Employment by Industry
- EV.4 – Changes in Productivity from Increased Connectivity (Agglomeration Effects)
- EV.5 – Changes in the Total Value of Exports and Imports

Category: Economic Vitality
(ECONOMIC VITALITY Worksheet)

General Indicator: Economic Impacts of Spending for Construction

Specific Indicator:

EV.1 – Number of Jobs Created or Retained by Bundle

This section describes how the specific indicator Number of Jobs Created or Retained by Bundle (EV.1) can be developed within the Mosaic tool. Users can rely on two sketch-planning models available in the Mosaic workbook. Alternatively, they may choose to run an economic simulation model (such as the IMPLAN input-output modeling system commercialized by the IMPLAN Group LLC (www.implan.com); or Oregon's State Wide Integrated Model (SWIM)), and simply enter the results of their analysis into the workbook.

Structure & Logic Diagram

Figure EV.1.15 provides a graphical representation of the equations coded in the Mosaic tool to estimate Specific Indicator EV.1, on the basis of a May 2009 memorandum prepared by the Council of Economic Advisers (CEA) and updated in September 2011[7].

The memorandum provides a simple rule for estimating the number of job-years "created" by government spending. It argues that $76,923 of government spending creates one job-year (one person employed for one year); with 64 percent of the job-year estimate representing direct and indirect effects, and 36 percent representing induced effects.

Thus, in Mosaic, the number of job-years associated with a bundle of actions can be estimated as:

Number of Job-Years = Total Capital Costs (dollars) / Government Spending per Job-Year (dollars per job-year)

Where:

- *Number of Job-Years* is the number of persons employed for one year (job-years created or retained) as result of short-term capital expenditures;
- *Total Capital Costs* is an estimate of total capital expenditures associated with a bundle of actions (see Specific Indicator FT.1, under the Funding the Transportation System & Finance category); and
- *Government Spending per Job-Year* is the estimate of $76,923 developed by the CEA and updated in September 2011.

[7] Executive Office of the President, Council of Economic Advisers, "Estimates of Job Creation from the American Recovery and Reinvestment Act of 2009," Washington, D.C., May 11, 2009; http://www.whitehouse.gov/administration/eop/cea/Estimate-of-Job-Creation (last access November 11, 2014); September 2011 update mentioned in Notice of Funding Availability for the Department of Transportation's National Infrastructure Investments under the Consolidated Appropriations Act, 2014, Footnote 3, page 13

The number of job-years resulting from direct and indirect spending is estimated simply as:

Number of Job-Years from Direct & Indirect Spending = Number of Job-Years x Percent of Job-Years from Direct & Indirect Spending

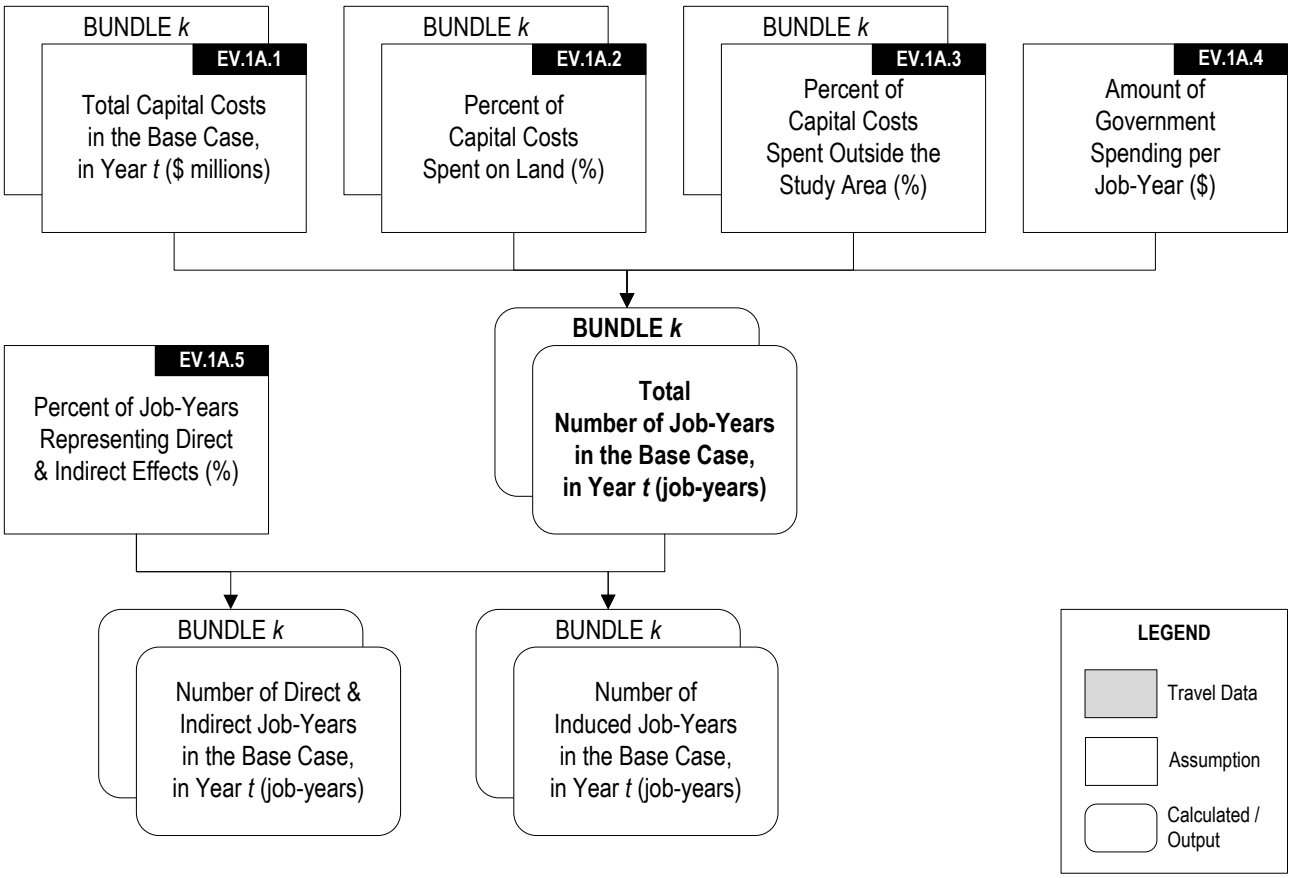
Where:

- *Number of Job-Years from Direct & Indirect Spending* is the number of job-years directly associated with a bundle (e.g., employment of construction workers on site) plus all job-years resulting from spending by businesses supplying intermediate goods and services to directly impacted firms (e.g., manufacturers of hard hats located in the study area).
- *Percent of Job-Years from Direct & Indirect Spending* is the estimate of 64 percent developed by the CEA.

The number of induced job-years (resulting from increased spending by all new or retained workers, employed by directly and indirectly impacted businesses) is estimated as:

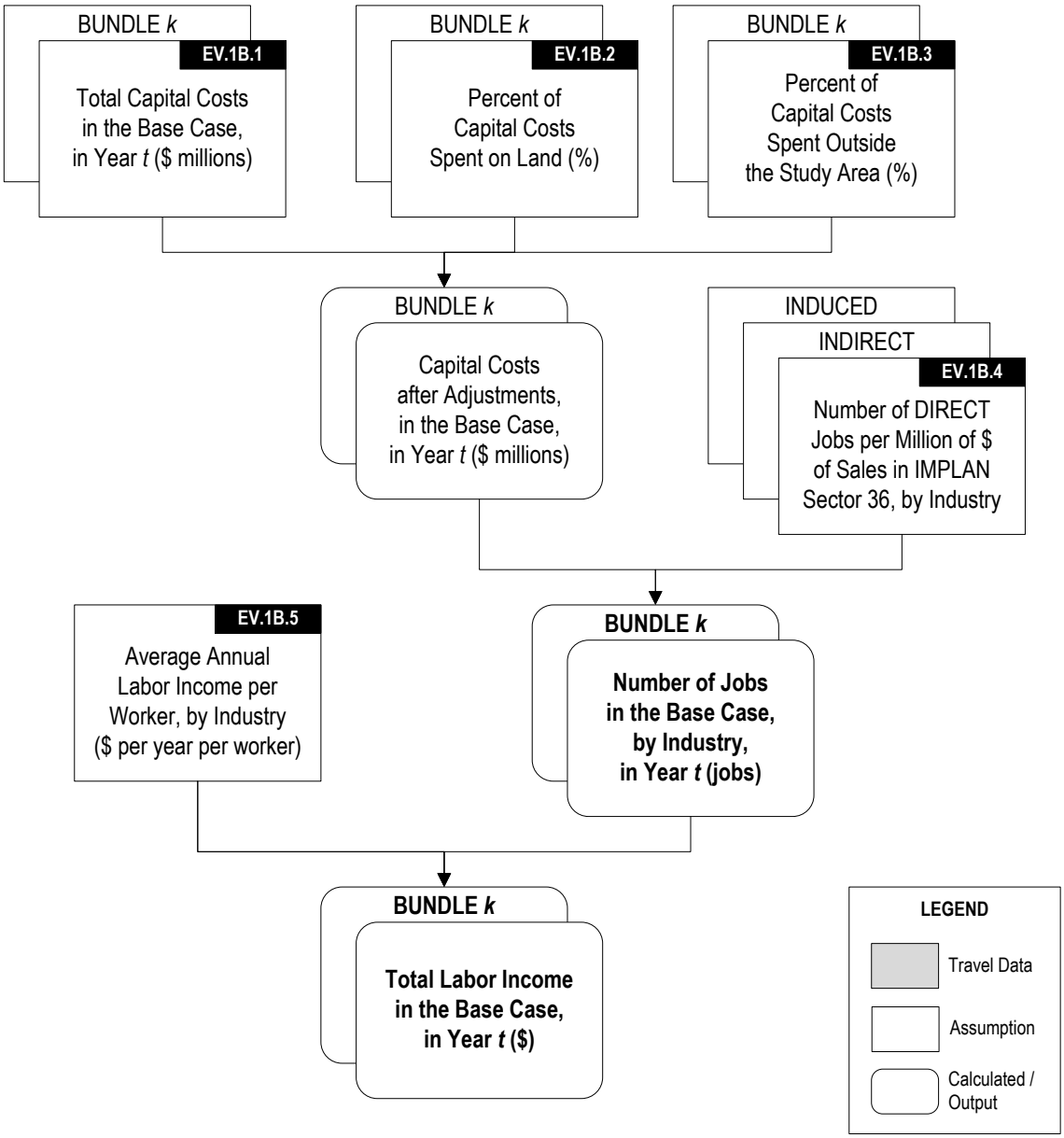
Number of Job-Years from Induced Spending = Number of Job-Years x (1 – Percent of Job-Years from Direct & Indirect Spending)

Figure EV.1.15: S&L Diagram for Estimating the Number of Jobs Created or Retained by Bundle, CEA Method



The second sketch-planning model available in the Mosaic tool uses a look-up table developed with IMPLAN. Detailed documentation on the IMPLAN modeling system is available from the Minnesota IMPLAN Group's website, at https://www.implan.com (last accessed November 11, 2014). Additional guidance on the use of IMPLAN can be found in the Specific Indicator sheet EV1.pdf available from the Mosaic website.

Figure EV.1.16: S&L Diagram for Estimating the Number of Jobs Created or Retained by Bundle, Use of IMPLAN Data



Data and Assumptions

Table EV.1A.7 and Table EV.1B.8 below provide information on the input variables used in the estimation of Specific Indicator EV.1. The input values that must be specified by Mosaic users are identified with an asterisk * in the Variable ID column. The last column of the tables indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table EV.1A.7: Data and Assumptions for Estimating the Number of Jobs Created or Retained by Bundle, CEA Method

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***EV.1A.1**	Total Capital Costs, in Year t	$ millions	Yes	Yes	Calculated from input data entered by user in COST & SCHEDULE worksheet	ECONOMIC VITALITY, Rows 19-29, Column D (calculated)
***EV.1A.2**	Percent of Capital Costs Spent on Land	%	Yes	Yes	Must be specified by user	ECONOMIC VITALITY, Rows 19-29, Column F
***EV.1A.3**	Percent of Capital Costs Spent Outside the Study Area	%	Yes	Yes	Must be specified by user	ECONOMIC VITALITY, Rows 19-29, Column G
EV.1A.4	Amount of Government Spending per Job-Year	$	No	No	Default value provided in tool, based in CEA research	SKETCH MODELS, Cell C506
EV.1A.5	Percent of Job-Years Representing Direct and Indirect Effects	%	No	No	Default value provided in tool, based in CEA research	SKETCH MODELS, Cell C507

Table EV.1B.8: Data and Assumptions for Estimating the Number of Jobs Created or Retained by Bundle, Use of IMPLAN Data

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***EV.1B.1**	Total Capital Costs, in Year t	$ millions	Yes	Yes	Calculated from input data entered by user in COST & SCHEDULE worksheet	ECONOMIC VITALITY, Rows 19-29, Column D (calculated)
***** **EV.1B.2**	Percent of Capital Costs Spent on Land	%	Yes	Yes	Must be specified by user	ECONOMIC VITALITY, Rows 19-29, Column F
***** **EV.1B.3**	Percent of Capital Costs Spent Outside the Study Area	%	Yes	Yes	Must be specified by user	ECONOMIC VITALITY, Rows 19-29, Column G
EV.1B.4	Number of Direct, Indirect and Induced Jobs per Million of Dollars of Sales in IMPLAN Sector 36, by Industry	Jobs	Yes	No	Default values provided in tool, derived with 2011 IMPLAN data for the United States	ECONOMIC DATA, Rows 12-452, Columns C-E; and Rows 462-482 (summary at 2-digit NAICS code)
EV.1B.5	Average Annual Labor Income per Worker, by Industry	$ per year per worker	Yes	No	Default values provided in tool, derived with 2011 IMPLAN data for the United States	ECONOMIC DATA, Rows 12-452, Column O; and Rows 462-482 (summary at 2-digit NAICS code)

The Number of Direct, Indirect and Induced Jobs per Million of Dollars of Sales in IMPLAN Sector 36 (Construction of other new nonresidential structures) by Industry and the Average Annual Labor Income per Worker by Industry were derived with 2011 IMPLAN data for the United States. Mosaic users should update these input values with data specific to their study area. IMPLAN data sets are available for individual States, Counties and Zip codes through the IMPLAN Group LLC.

Category: Economic Vitality
(ECONOMIC VITALITY Worksheet)

General Indicator: Economic Impacts of more Efficient Transportation Services

Specific Indicator:

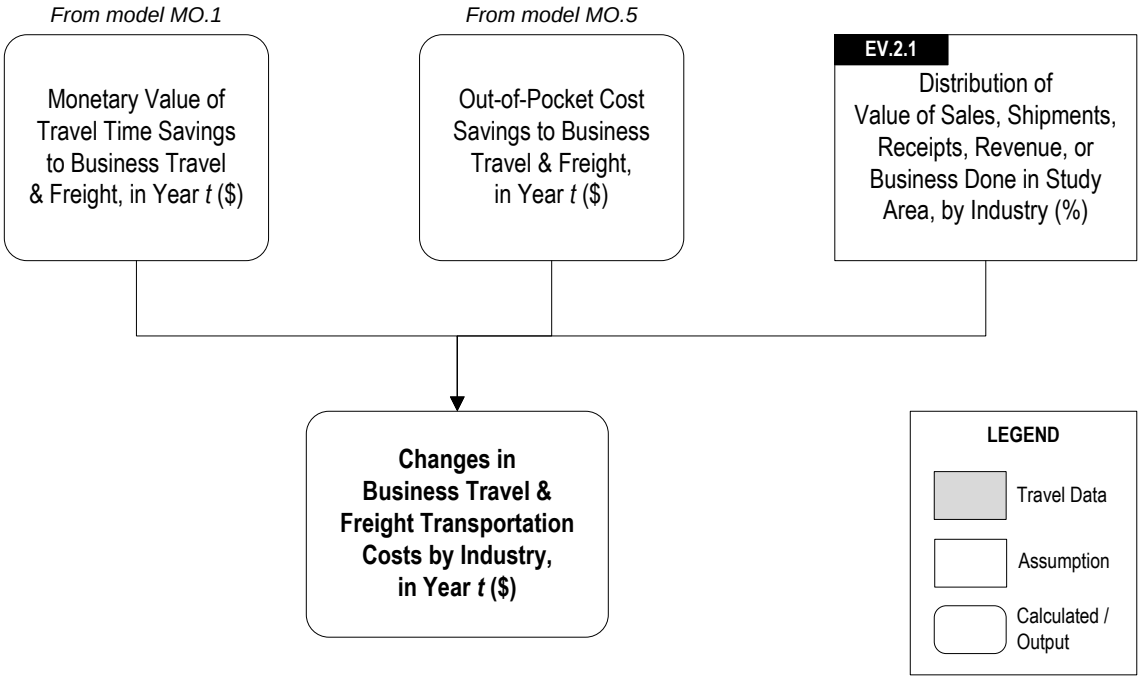
EV.2 – Changes in Business and Freight Transportation Costs by Industry

This specific indicator is the change in total transportation costs to business (on-the-job) travel and freight. It is estimated directly from travel demand model output, using the methods and parameter values introduced in the MOBILITY section of this document. It is, essentially, the portion of mobility benefits (travel time savings and reductions in out-of-pocket costs) accruing directly to businesses.

Structure & Logic Diagram

The figure below provides a graphical representation of the main variables and calculations used in the development of Specific Indicator EV.2.

Figure EV.2.17: S&L Diagram for Estimating Changes in Business Travel and Freight Transportation Costs by Industry



In Version 2.0 of the Mosaic tool, the distribution of transportation cost savings by industry is based simply on the distribution of economic activity within the study area. Default values for the State of Oregon, based on estimates of the Total Value of Sales, Shipments, Receipts, Revenue, or Business done from the 2007 Economic Census are provided within the workbook. In future applications, estimates of freight flows and employment by industry for a specific study area, freight corridor or set of corridors should be used instead.

Changes in Generalized Transportation Costs are distributed by 2-digit NAICS (North American Industry Classification System) code, with the equation:

$$Change\ in\ GC_k\ in\ Industry_i = Change\ in\ GC_k\ x\ Share\ of\ Industry_i\ in\ Total\ Value\ of\ Sales$$

Where:

- *Change in GC_k* is the change in the Generalized Cost of Travel estimated for all business travel and freight flows, under Bundle *k*; and
- *Share of Industry $_i$ in Total Value of Sales* is the Total Value of Sales, Shipments, Receipts, Revenue, or Business Done in Industry *i*, divided by the grand total for the study area (State).

Information on the North American Industry Classification System can be found at https://www.census.gov/eos/www/naics/ (last accessed November 11, 2014).

Data and Assumptions

The table below provides information on the input variables used in the estimation of Specific Indicator EV.2. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table EV.2.9: Data and Assumptions for Estimating Changes in Business Travel and Freight Transportation Costs by Industry

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
EV.2.1	Distribution of Value of Sales, Shipments, Receipts, Revenue, or Business Done in Study Area, by Industry	%	Yes	No	Default values provided in tool (for the State of Oregon, from the 2007 Economic Census)	ECONOMIC DATA, Rows 545-575

Category: Economic Vitality
(ECONOMIC VITALITY Worksheet)

General Indicator: Economic Impacts of more Efficient Transportation Services

Specific Indicator:

EV.3 – Changes in Employment by Industry

This specific indicator examines the long-term impacts of transportation improvements on economic activity and total employment within a study area.

The recommended approaches to estimating EV.3 consists of either running SWIM and using the outcomes of the analysis as inputs to the Mosaic tool; or using look-up tables based on SWIM scenario runs to estimate the indicator within the Mosaic tool. The sketch-planning models included in Version 2.0 of the Mosaic workbook should be viewed as an *interim solution*.

Structure & Logic Diagrams

The figures below illustrate how indicator EV.3 can be estimated within Version 2.0 of the Mosaic tool. Two broad categories of impacts are estimated:

- Impacts of traffic flow improvements (increase in average vehicle speed) on logistic costs, in the Construction, Manufacturing, Wholesale Trade, and Retail Trade industries, and resulting effects on regional employment (Figure EV.3.18); and
- Impacts of changes in commuting delay costs on the demand for labor and regional employment (Figure EV.3.19).

Detailed technical documentation on both categories of impacts is available from *The Economic Costs of Congestion in the New York City Region*, Final Report prepared by HDR for the Partnership for New York City, November 27, 2006.

Figure EV.3.18: S&L Diagram for Estimating Changes in Employment by Industry, and Associated Income Metrics, Impacts on Industry Costs

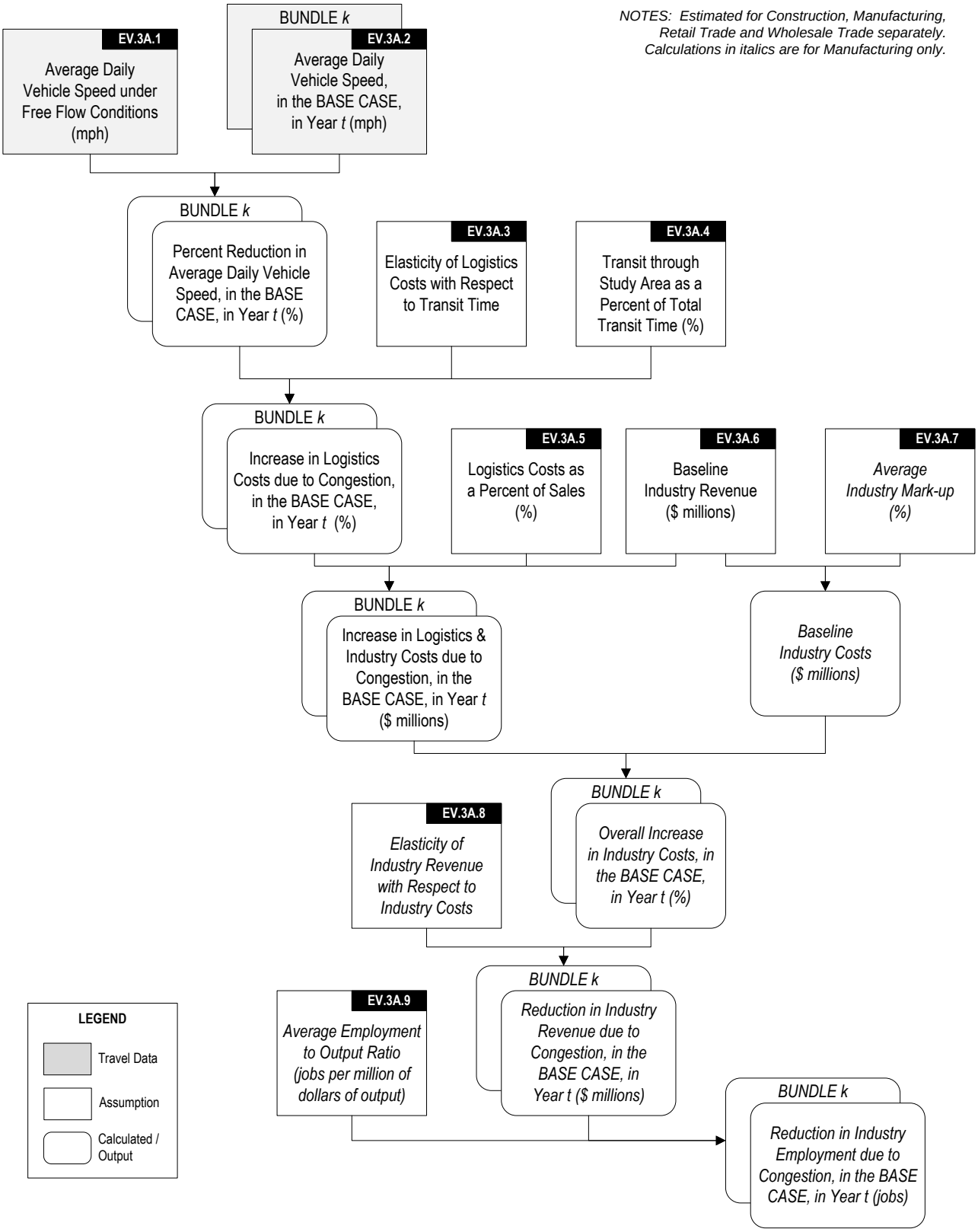
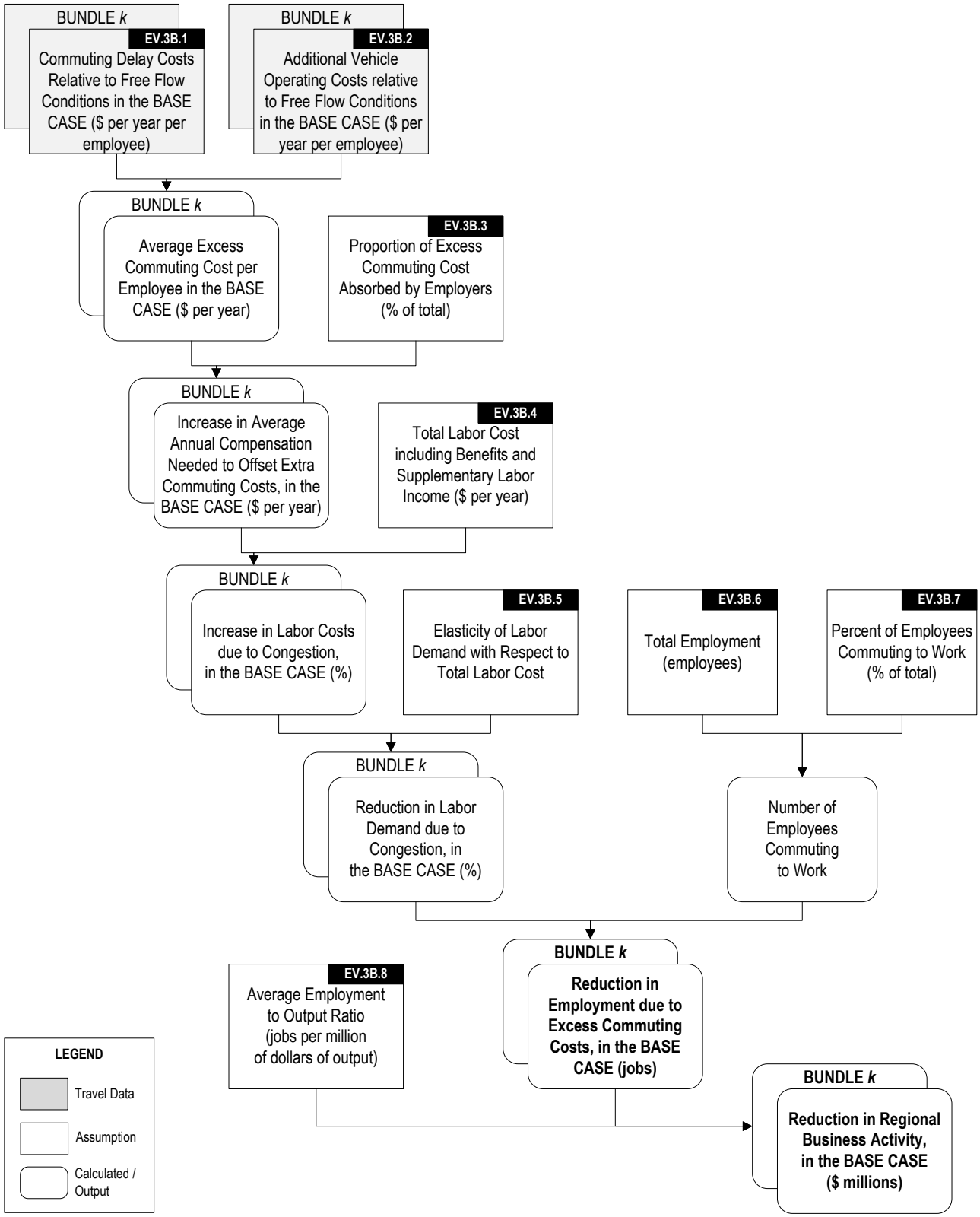


Figure EV.3.19: S&L Diagram for Estimating Changes in Employment by Industry, and Associated Income Metrics, Impacts on Labor Demand



Data and Assumptions

Table EV.3A.10 and Table EV.3B.11 below provide information on the input variables used in the estimation of Specific Indicator EV.3. The input values that must be specified by Mosaic users are identified with an asterisk * in the Variable ID column. The last column of the tables indicates where in the tool the input values must be entered.

Table EV.3A.10: Data and Assumptions for Estimating Changes in Employment by Industry, Impacts on Industry Costs

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***EV.3A.1**	Average Daily Vehicle Speed under Free Flow Conditions	Miles per hour	Yes	Yes	Must be specified by user (calculated from travel demand model output data)	SKETCH MODELS, Row 531
***EV.3A.2**	Average Daily Vehicle Speed, in Year t	Miles per hour	Yes	Yes	Must be specified by user (calculated from travel demand model output data)	SKETCH MODELS, Row 532
EV.3A.3	Elasticity of Logistics Costs with Respect to Transit Time, by Industry	n/a	No	No	Default parameter value provided in tool	SKETCH MODELS, Rows 556, 568, 580, and 600
EV.3A.4	Transit through Study Area as a Percent of Total Transit Time, by Industry	%	Yes	No	Default value provided in tool; should be updated by user	SKETCH MODELS, Rows 557, 569, 581, and 601
EV.3A.5	Logistics Costs as a Percent of Sales, by Industry	%	No	No	Default value provided in tool	SKETCH MODELS, Rows 555, 567, 579, and 599
EV.3A.6	Baseline Industry Revenue (in study area)	$ millions	Yes	No	Default value provided in tool, based on County data from BLS and 2007 Economic Census	SKETCH MODELS, Rows 558, 570, 582, and 602
EV.3A.7	Average Industry Mark-up (Manufacturing)	%	No	No	Default value provided in tool	SKETCH MODELS, Row 584
EV.3A.8	Elasticity of Industry Revenue with Respect to Industry Costs (Manufacturing)	n/a	No	No	Default parameter value provided in tool	SKETCH MODELS, Row 583
EV.3A.9	Average Employment to Output Ratio (Manufacturing)	Jobs per million of dollars of output	Yes	No	Default value provided in tool, calculated with State data from BLS and 2007 Economic Census	SKETCH MODELS, Row 586

Table EV.3B.11: Data and Assumptions for Estimating Changes in Employment by Industry, Impacts on Labor Demand

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***EV.3B.1**	Commuting Delay Costs Relative to Free Flow Conditions	$ per year per employee	Yes	Yes	Must be specified by user (calculated from travel demand model output data)	SKETCH MODELS, Row 641
***EV.3B.2**	Additional Vehicle Operating Costs relative to Free Flow Conditions	$ per year per employee	Yes	Yes	Must be specified by user (calculated from travel demand model output data)	SKETCH MODELS, Row 642
EV.3B.3	Proportion of Excess Commuting Cost Absorbed by Employers	% of total	Yes	No	Default value provided in tool	SKETCH MODELS, Row 644
EV.3B.4	Total Labor Cost including Benefits & Supplementary Labor Income	$ per year	Yes	No	Default value provided in tool, based on BLS data for the State	SKETCH MODELS, Row 646
EV.3B.5	Elasticity of Labor Demand with Respect to Total Labor Cost	n/a	No	No	Default parameter value provided in tool	SKETCH MODELS, Row 647
EV.3B.6	Total Employment (in study area)	Employees	Yes	No	Default value provided in tool, based on BLS data for the State of Oregon by County	SKETCH MODELS, Row 648
EV.3B.7	Percent of Employees Commuting to Work	% of total	Yes	No	Default value provided in tool, based on Census data for the State	SKETCH MODELS, Row 649
EV.3B.8	Average Employment to Output Ratio	jobs per million of dollars of output	Yes	No	Default value provided in tool	SKETCH MODELS, Row 651

Category: Economic Vitality
(ECONOMIC VITALITY Worksheet)

General Indicator: Structural Economic Effects of Transportation System Improvements

Specific Indicator:

EV.4 – Changes in Productivity from Increased Connectivity

This specific indicator examines the impacts of transportation improvements on the productivity of workers and firms within a study area, through so-called "agglomeration economies"[8].

Agglomeration economies are benefits resulting from the spatial concentration of economic activity. They arise through increases in the "effective density" of a location, defined as total employment in and around the area, weighted by their proximity to the location—where proximity is measured in terms of generalized travel cost, not distance. As effective density increases, firms and workers become more productive, as a result of knowledge spillover, labor market pooling, specialization, or more efficient input-output sharing (e.g., reduced delivery times).

The agglomeration consequences of a transportation plan have two components. One is always positive: a transportation improvement brings people and firms closer together (in terms of travel time between the firms' locations). The other can be positive or negative: positive if it encourages increased employment in cities or "clusters" of economic activity; negative if it encourages the dispersion of activity (firms or workers relocating as a result of declining travel costs).

Agglomeration economies can be measured using estimates of the elasticity of total productivity with respect to the density of employment in an area (by industry); the change in the effective density of employment in the area due to the transportation improvements; and a measure of economic output (e.g., Gross Domestic Product). The elasticity of productivity with respect to agglomeration (i.e., the change in density) is typically estimated using econometric techniques applied to cross-sectional data at the industry level. The resulting estimates of agglomeration economies are expressed in monetary terms. They represent additional output or value added (i.e., increases in Gross Domestic Product) and can be added to other benefit categories, in a Benefit-Cost Analysis.

Three options are available to Mosaic users for estimating this indicator:

- **Option 1:** Use of methods and parameter values described in the UK Department for Transport, Transport Analysis Guidance (TAG) Unit A2.1, Wider Impacts, dated January 2014. This document is available at https://www.gov.uk/government/publications/webtag-tag-unit-a2-1-wider-impacts (last accessed November 11, 2014).
- **Option 2:** For investments in rail transit in a Metropolitan Area only, use of a sketch-planning tool available in the Mosaic workbook and derived from a 2012 web-only document prepared

[8] The description of agglomeration economies and associated estimation methods, in this section, borrow from UK Department for Transport, Transport, Wider Economic Benefits, and Impacts on GDP, Discussion Paper, July 2005

for the Transportation Research Board, Transit Cooperative Research Program (TCRP) and available at https://www.trb.org/Main/Blurbs/167284.aspx (last accessed November 11, 2014).

- **Option 3:** Use of a spreadsheet-based calculator developed under Project C11 (Development of Tools for Assessing Wider Economic Benefits of Transportation) of the Transportation Research Board's Second Strategic Highway Research Program (SHRP 2). This calculator, external to Mosaic, can be downloaded at http://www.tpics.us/tools (last accessed November 11, 2014). Users should select the Effective Density Buyer-Supplier Market Access Tool, and follow instructions provided in the 2013 Accessibility Analysis Tools Technical Documentation and User's Guide.

Structure & Logic Diagrams

Options 1 and 2 are described further in the rest of this documentation sheet. For additional information on Option 3, users should refer to the SHRP 2 C11 Report and Technical Documentation.

Option 1 – Use of UK DfT Methods and Parameter Values

With travel data and modeling capabilities *external* to the Mosaic workbook, users would first estimate the impact of each bundle on the "effective density" of employment in a given area. The effective density of employment reflects the degree of accessibility to a firm or industry from a neighboring area, by weighting the number of workers living in the neighboring area by a measure of the transportation costs between the two locations.

Effective density can be estimated using the following formula:

$$D_{t,i} = \sum_{k=1}^{N} E_{t,k} \times T_{t,i,k}^{\propto}$$

Where:

- $D_{t,i}$ is the effective density of employment of area *i* in Year *t*;
- $E_{t,k}$ is work-place based employment in area *k* in Year *t*;
- $T_{t,i,k}$ is the generalized cost of travel between areas *i* and *k* in Year *t*; and
- α is a scaling parameter.

The following simplified equation can then be applied to estimate agglomeration economies:

Agglomeration Economies ($millions) =
Change in Density (percent) x Elasticity of Productivity x GDP ($millions)

Where:

- *Change in Density* is the percent change in the effective density of employment in the study area due to the bundle of actions;

- *Elasticity of Productivity* is the elasticity of total productivity with respect to the effective density of employment in the area; and
- *GDP* is total value added, or Gross Domestic Product, within the area.

Detailed specifications for the above two equations can be found in the UK DfT guidance document referenced above (TAG Unit A2.1, January 2014).

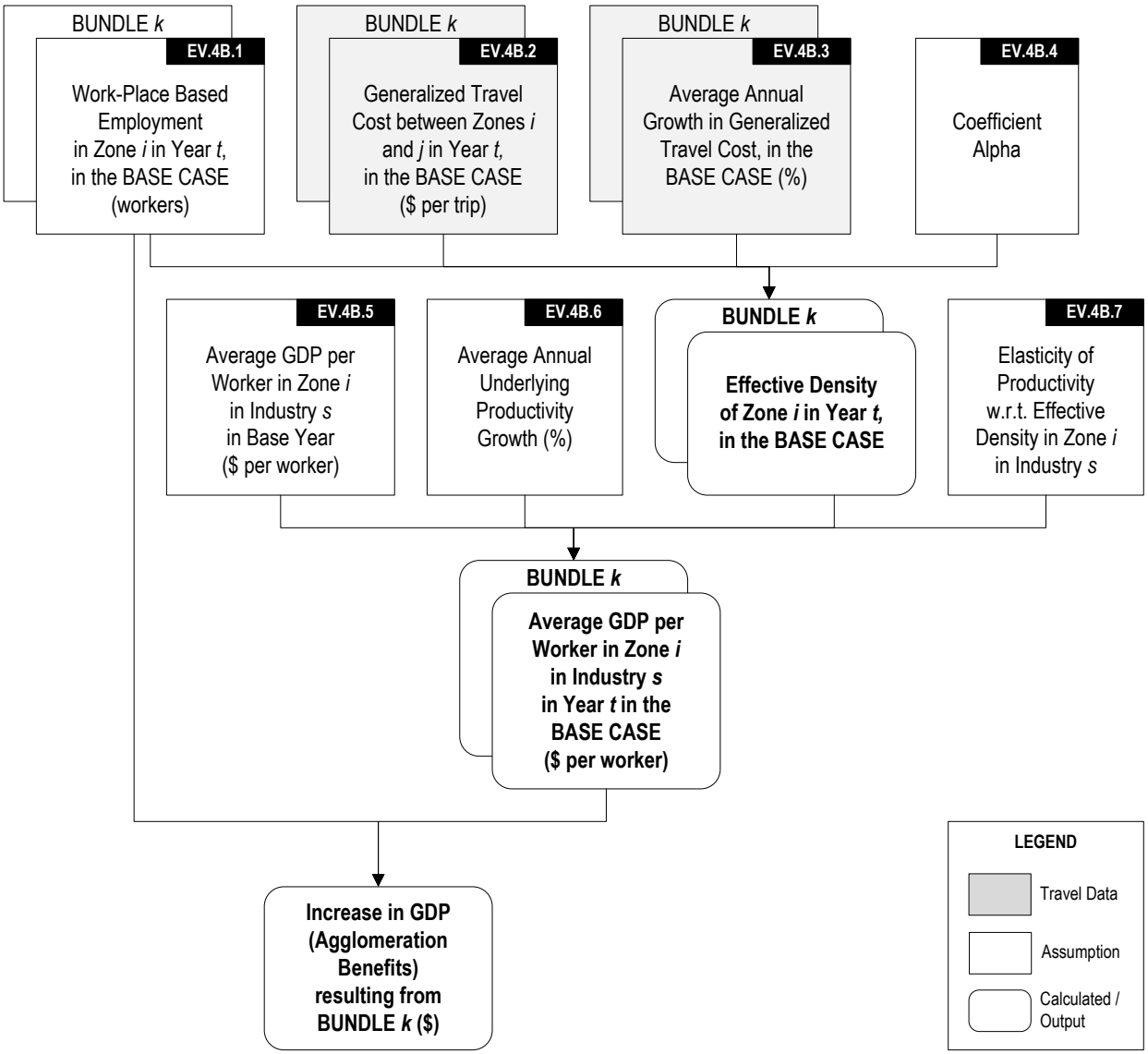
Estimates of the elasticity of total productivity with respect to effective density may be sourced from the economic literature, although estimation for a specific study area is typically recommended. As an illustration, the table below summarizes the findings of a meta-analysis of 729 elasticities taken from 34 different studies.

	Number of Observations	Mean	Median	Standard Deviation	Min	Max
By measure of urban agglomeration						
Market Potential	279	0.101	0.076	0.143	-0.277	0.658
Density	158	0.030	0.039	0.099	-0.800	0.300
Size	292	0.032	0.030	0.076	-0.410	0.319
By type of response variable						
Labor productivity	342	0.053	0.038	0.095	-0.366	0.503
Output	264	0.076	0.057	0.156	-0.800	0.658
Wages	123	0.034	0.032	0.030	-0.096	0.143
By industry group						
Economy-wide	168	0.031	0.034	0.099	-0.800	0.250
Manufacturing	427	0.040	0.036	0.095	-0.366	0.658
Services	134	0.148	0.142	0.148	-0.219	0.503

Source: Melo et al., A Meta-Analysis of Estimates of Urban Agglomeration Economies, Regional Science and Urban Economics, 39, 2009, 332-342, Table 2 page 335

The UK DfT method is illustrated in Figure EV.4.20, on the next page.

Figure EV.4.20: S&L Diagram for Estimating Changes in Productivity from Increased Connectivity, UK DfT Method

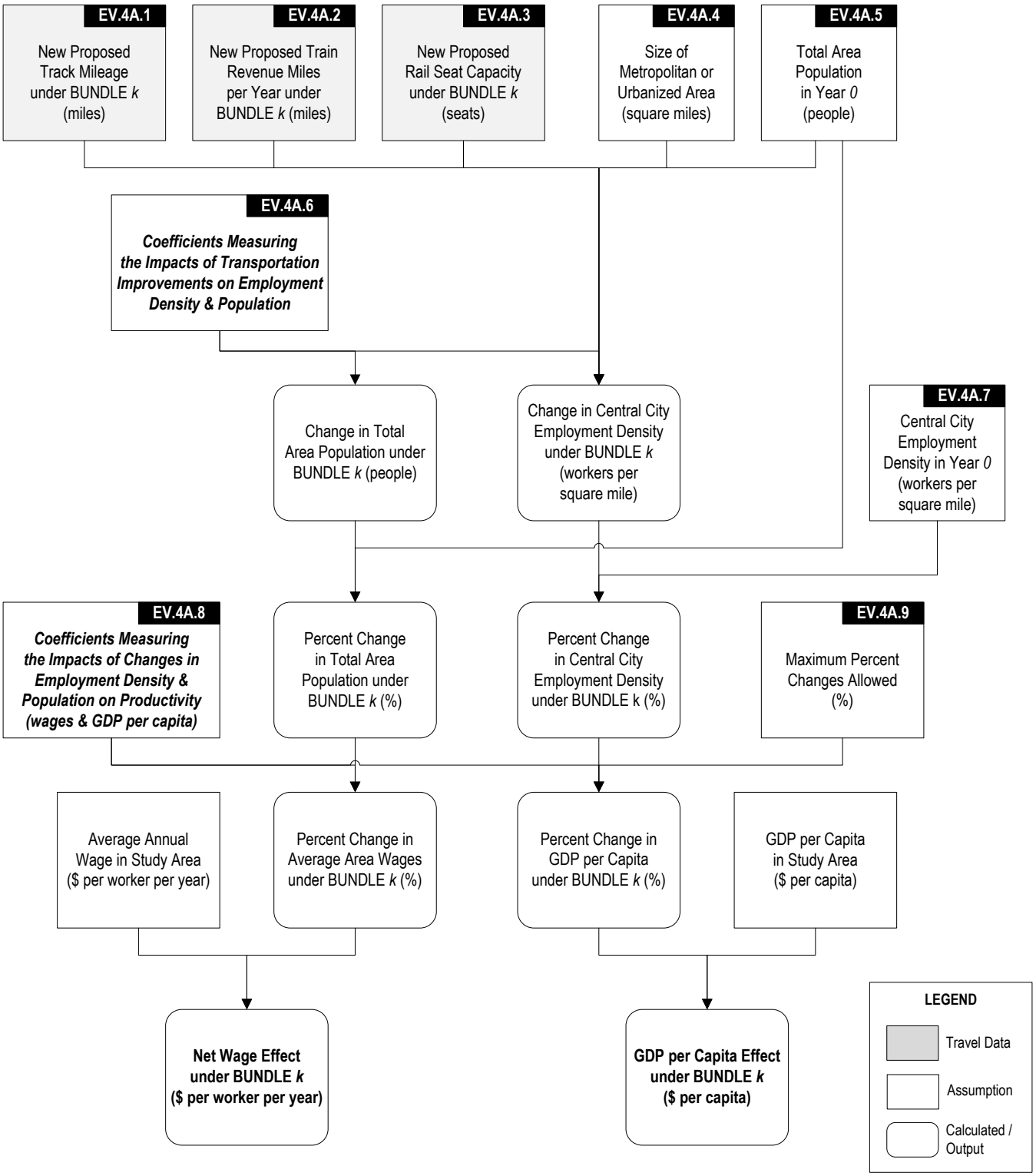


Option 2 – Use of TCRP Method for Investments in Rail Transit

The S&L diagram on the next page illustrates the method developed for TCRP under the 2012 research project *Methodology for Determining the Economic Development Impacts of Transit Projects.* As noted above, users should refer to the TRB website https://www.trb.org/TCRP/Blurbs/167284.aspx (last accessed November 11, 2014), for more information on the TCRP model and supporting research findings.

The TCRP method has been coded in the Mosaic workbook and is available for use in the SKETCH MODELS worksheet, starting in Row 669.

Figure EV.4.21: S&L Diagram for Estimating Changes in Productivity from Increased Connectivity, TCRP Method



Data and Assumptions

The table below provides information on the input variables used in the estimation of Specific Indicator EV.4 using the TCRP method (Option 2, for investments in rail transit only). The input values that must be specified by Mosaic users are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table EV.4A.12: Data and Assumptions for Estimating Changes in Productivity from Increased Connectivity, TCRP Method

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***EV.4A.1**	New Proposed Track Mileage	Miles	Yes	Yes	Must be specified by user	SKETCH MODELS, Row 672
***EV.4A.2**	New Proposed Train Revenue Miles per Year	Miles	Yes	Yes	Must be specified by user	SKETCH MODELS, Row 673
***EV.4A.3**	New Proposed Rail Seat Capacity	Seats	Yes	Yes	Must be specified by user	SKETCH MODELS, Row 674
EV.4A.4	Size of Metropolitan or Urbanized Area	Square miles	Yes	No	Default value provided for Portland MSA	SKETCH MODELS, Row 686
EV.4A.5	Total Area Population, in Year 0	Persons	Yes	Yes/No	Default value provided for Portland MSA	SKETCH MODELS, Row 677
EV.4A.6	Coefficients Measuring the Impacts of Transportation Improvements on Employment Density and Population	n/a	No	No	Parameter values provided in tool, based on TCRP research findings	SKETCH MODELS, Rows 692-698
EV.4A.7	Central City Employment Density in Year 0	Workers per square mile	Yes	No	Default value provided for Portland MSA	SKETCH MODELS, Row 684
EV.4A.8	Coefficients Measuring the Impacts of Changes in Employment Density and Population on Productivity	n/a	No	No	Parameter values provided in tool, based on TCRP research findings	SKETCH MODELS, Rows 704-707
EV.4A.9	Maximum Percent Changes Allowed	%	No	No	Value provided in tool, as defined in TCRP model	SKETCH MODELS, Rows 714-720, Column I

Category: Economic Vitality
(ECONOMIC VITALITY Worksheet)

General Indicator: Structural Economic Effects of Transportation System Improvements

Specific Indicator:

EV.5 – Changes in the Total Value of Exports and Imports

This specific indicator examines how transportation improvements may affect the total value of international trade between Oregon and other countries. The preferred approach for developing this indicator is to use Oregon's State Wide Integrated Model (SWIM) and enter the resulting estimates of changes in exports and imports directly into the tool. An alternative, sketch-planning, approach is proposed in Version 2.0 of the Mosaic tool. This approach is outlined below.

Structure & Logic Diagram

The variables and operations used in the development of Specific Indicator EV.5 are illustrated in the S&L diagram on the next page. The method is based on recommendations developed by NERA Economic Consulting for the UK Department for Transport, summarized in the report "Representing International Business Impacts in Transport Appraisal", dated April 2010 and available online at http://webarchive.nationalarchives.gov.uk/20111005175811/http://www.dft.gov.uk/publications/repr esenting-international-business-impacts-in-transport-appraisal (last accessed November 11, 2014).

Changes in the values of exports and imports are estimated separately using the following equations:

$$\textit{Change in Imports}_{(\%)} = \textit{Change in Transportation Cost Factor}_{(\%)} \times \textit{Trade Elasticity for Imports}$$

$$\textit{Change in Imports}_{(\$millions)} = \textit{Change in Imports}_{(\%)} \times \textit{Value of Imports in Base Case}_{(\$millions)}$$

And:

$$\textit{Change in Exports}_{(\%)} = \textit{Change in Transportation Cost Factor}_{(\%)} \times \textit{Trade Elasticity for Exports}$$

$$\textit{Change in Exports}_{(\$millions)} = \textit{Change in Exports}_{(\%)} \times \textit{Value of Exports in Base Case}_{(\$millions)}$$

Where:

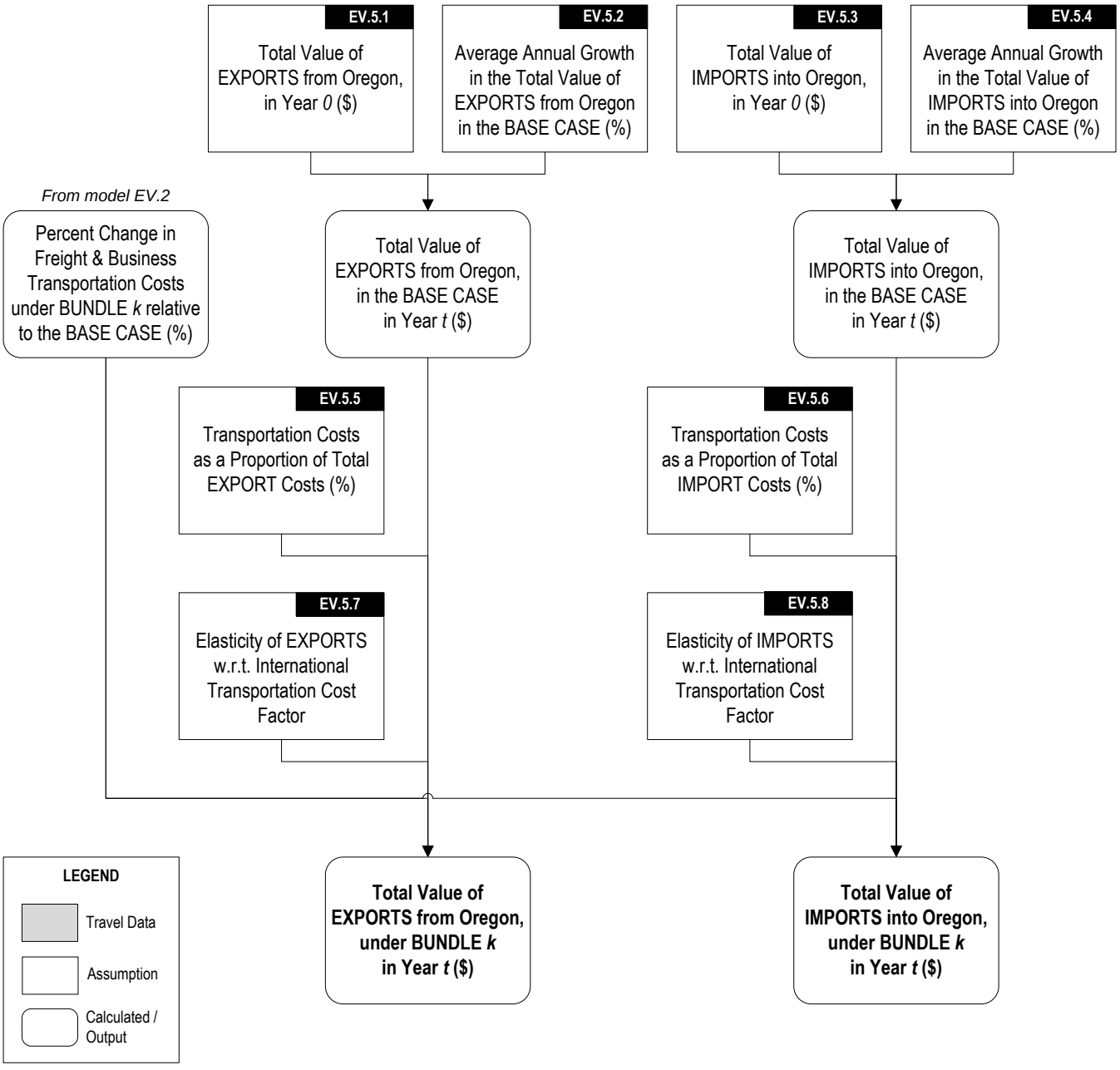
- *Transportation Cost Factor* is the ratio of international transportation costs to total costs (including all production, marketing and logistic costs);
- *Trade Elasticity* is the elasticity of international trade with respect to the international transportation cost factor; and
- *Value of Imports (or Exports)* is the annual, dollar value of imports (or exports) of merchandise from (to) other countries[9].

[9] Similar calculations can be set-up for international trade in services.

Estimates of the elasticity of international trade with respect to the international transportation cost factor are available from the economic literature. The range recommended in NERA (2010) is defined by a low elasticity of -0.3 and a high elasticity of -3.0 (page 21).

Changes in the generalized transportation cost of internationally traded goods (for all goods traded in the Base Case) should be estimated from Specific Indicator EV.2, Changes in Transportation Costs by Industry. Estimates for the international Transportation Cost Factor may be developed from reasoned assumptions, parameter values from SWIM, or international trade data (e.g., commodity values, major origins and destinations).

Figure EV.5.22: S&L Diagram for Estimating Changes in the Total Value of Exports and Imports



Data and Assumptions

The table below provides information on the input variables used in the estimation of Specific Indicator EV.5. The input values that must be specified by Mosaic users are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table EV.5.13: Data and Assumptions for Estimating Changes in Changes in the Total Value of Exports and Imports

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
EV.5.1	Total Value of EXPORTS from Oregon, in Year 0	$	No	No	Value provided in tool, based on US Department of Commerce (DOC) Trade Stats	ECONOMIC VITALITY, Cell D200
EV.5.2	Average Annual Growth in the Total Value of EXPORTS from Oregon in the BASE CASE	%	No	No	Default value provided in tool (conservatively set to zero)	ECONOMIC VITALITY, Cell F200
EV.5.3	Total Value of IMPORTS into Oregon, in Year 0	$	No	No	Value provided in tool, based on US DOC Trade Stats	ECONOMIC VITALITY, Cell D201
EV.5.4	Average Annual Growth in the Total Value of IMPORTS into Oregon in the BASE CASE	%	No	No	Default value provided in tool (conservatively set to zero)	ECONOMIC VITALITY, Cell F201
EV.5.5	Transportation Costs as a Proportion of Total EXPORT Costs	%	No	No	Value provided in tool, for illustration	ECONOMIC VITALITY, Cell D204
EV.5.6	Transportation Costs as a Proportion of Total IMPORT Costs	%	No	No	Value provided in tool, for illustration	ECONOMIC VITALITY, Cell D205
EV.5.7	Elasticity of EXPORTS with Respect to International Transportation Cost Factor	n/a	No	No	Default parameter value provided in tool, based on literature	ECONOMIC VITALITY, Cell D208
EV.5.8	Elasticity of IMPORTS with Respect to International Transportation Cost Factor	n/a	No	No	Default parameter value provided in tool, based on literature	ECONOMIC VITALITY, Cell D209

Environmental Stewardship

Documentation sheets for the following specific indicators can be found in this section:

- ES.1 – Criteria Air Contaminants
- ES.2 – Air Toxics (Benzene and Diesel Particulate Matter)
- ES.3 – Life-Cycle CO2e
- ES.4 – Natural, Built, and Cultural Resources at Risk

Category: Environmental Stewardship (ENVIRONMENT Worksheet)

General Indicator: Air

Specific Indicator:

ES.1 – Criteria Air Contaminants

This specific indicator is the change in vehicular emissions of Criterion Air Contaminants (CAC). CAC refer to six pollutant compounds: nitrogen oxides (NOx), sulfur dioxide (SO2), fine particulate matter (PM2.5), ozone, carbon monoxide (CO), and lead. An additional pollutant, volatile organic compounds (VOCs), although not defined by the U.S. Environmental Protection Agency (EPA) as a CAC, is also considered in this group because it is regulated and has similar effects on human health and welfare.

Structure & Logic Diagram

Figure ES.1.23 on the next page provides a graphical representation of the main variables and calculations required to develop Specific Indicator ES.1. The variables involved in operations performed *outside* the Mosaic workbook are represented in boxes with dotted lines. They include:

- Estimates of annual Vehicle Miles Traveled by County, Facility Type, Speed Bin and Year, in the base case and for each bundle;
- Vehicle fleet mix forecasts from the Regional Strategic Planning Model (RSPM, formerly "GreenSTEP") or other sources; and
- Emission rates, in grams per VMT, by County, Facility Type, Vehicle Type, Speed Bin and Year, from the EPA's MOtor Vehicle Emission Simulator (MOVES).

Fleet mix forecasts from RSPM or other sources should be combined with annual VMT projections developed from a travel demand model to obtain future VMT by vehicle type. Total annual emissions can then be estimated using emission rates from MOVES, as follows:

$$\textit{Annual Emissions} = \textit{Annual VMT}_{\textit{(miles)}} \times \textit{Emission rate}_{\textit{(grams/mile)}}$$

Information on the RSPM model can be found on the Oregon DOT website, at https://www.oregon.gov/ODOT/Planning/Pages/Technical-Tools.aspx#GreenSTEP (last accessed November 10, 2014).

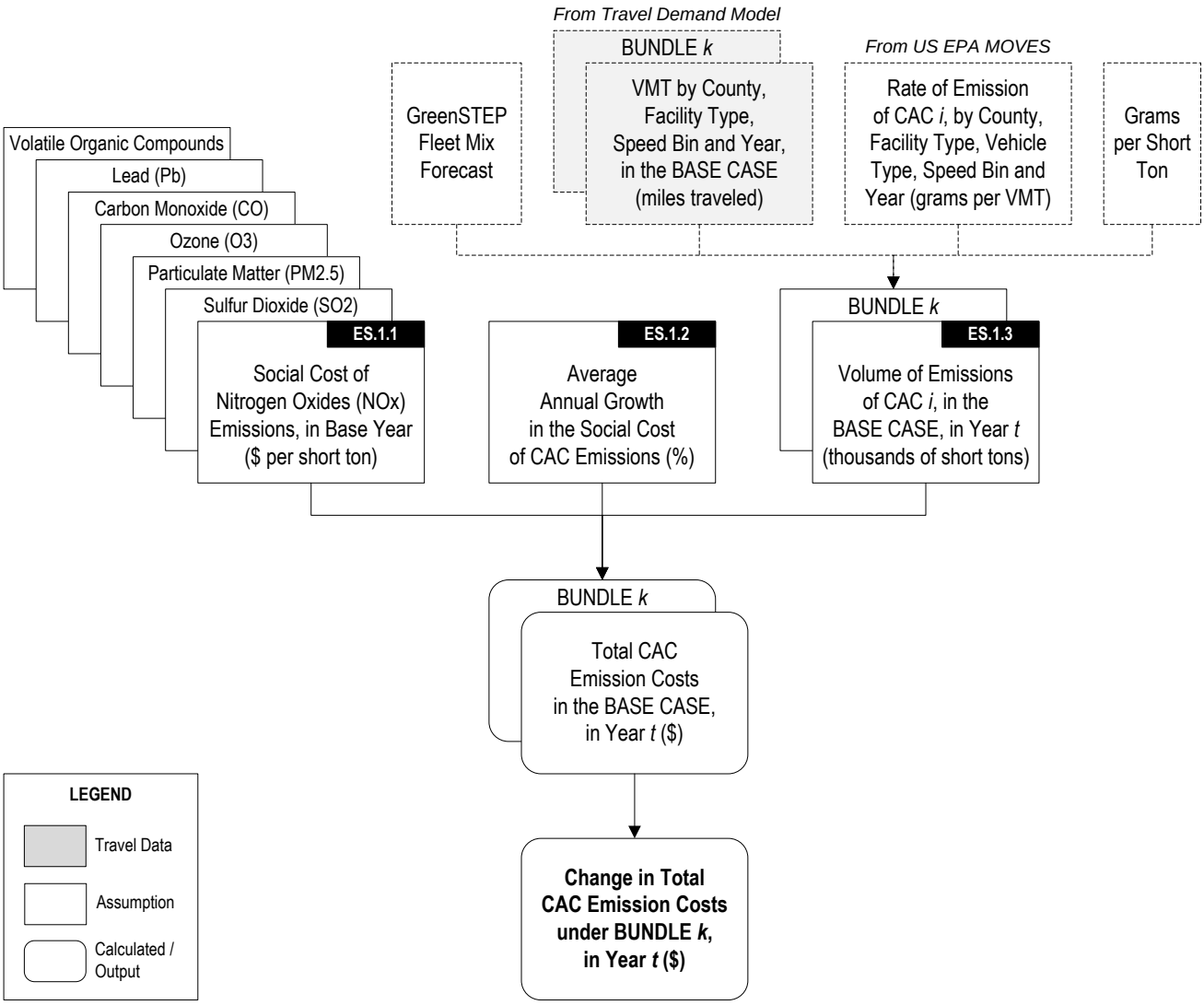
User documents and tools related to MOVES are available on the EPA website at https://www.epa.gov/moves (last accessed November 10, 2014).

In the Mosaic tool, the volumes of CAC emissions are monetized with estimates of average damage costs (i.e., human health effects, in dollars per ton of pollutants) developed by the EPA. Annual monetized emission costs are estimated within the tool as:

$$\textit{Annual Emission Costs} = \textit{Annual Emissions}_{\textit{(tons)}} \times \textit{Emission Costs}_{\textit{(\$/ton)}}$$

Detailed information on the damage costs developed by EPA is available in the Joint Technical Support Document of the Final Rulemaking for 2017-2025 Light-Duty Vehicle Greenhouse Gas Emission Standards and Corporate Average Fuel Economy Standards, dated August 2012 (pp. 4-42 to 4-45); available at https://www.regulations.gov/document?D=EPA-HQ-OAR-2010-0799-12013 (last accessed November 10, 2014).

Figure ES.1.23: S&L Diagram for Estimating Changes in Criteria Air Contaminant Emissions



Data and Assumptions

The table below provides information on the input variables used in the estimation of Specific Indicator ES.1, within the Mosaic tool. The input values that must be specified by the user are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table ES.1.14: Data and Assumptions for Estimating Changes in Criteria Air Contaminant Emissions

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
ES.1.1	Social Cost of CAC Emissions, in Base Year	$ per short ton	No	No	Default values provided in tool, based on estimates developed by EPA; estimates not available for Lead and Ozone	MODEL PARAMETERS, Rows 69-95
ES.1.2	Average Annual Growth in the Social Cost of CAC Emissions	% per year	No	No	Assumed to be zero in this version of the tool	Can be adjusted in ENVIRONMENT, Cells starting CA12 , CA27, CA42, CA57, CA72, CA87, CA102
***ES.1.3**	Volume of Emissions of CAC, in Year t	Thousands of short tons	Yes	Yes	Must be specified by user (use of emission rates from EPA's MOVES recommended)	ENVIRONMENT, Rows 14-24, 29-39, 44-54, 59-69, 74-84, 89-99, and 104-114

Category: Environmental Stewardship (ENVIRONMENT Worksheet)

General Indicator: Air

Specific Indicator: ES.2 – Air Toxics (Benzene & Diesel PM)

This indicator examines changes in the emissions of two air toxics: Benzene (a Mobile Source Air Toxic, or MSAT) and Diesel Particulate Matter (a Non-Mobile Source Air Toxic, or NMSAT). MSAT and NMSAT are compounds emitted from highway vehicles and non-road equipment which are known or suspected to cause cancer and other serious health effects. The effects of NMSAT are generally more localized than those of MSAT.

In Version 2.0 of the Mosaic tool, Specific Indicator ES.2 is defined as the sum of Benzene and Diesel PM emissions (expressed in short tons). This indicator is not monetized but can be scored, quantitatively or qualitatively.

Structure & Logic Diagram

The figure below provides a graphical representation of the main variables and calculations required to develop Specific Indicator ES.2. The variables involved in operations performed *outside* the Mosaic workbook are represented in boxes with dotted lines. They include:

- Estimates of annual Vehicle Miles Traveled by County, Facility Type, Speed Bin and Year, in the base case and for each bundle;
- Vehicle fleet mix forecasts from the RSPM model or other sources; and
- Benzene and Diesel PM emission rates, in grams per VMT, by County, Facility Type, Vehicle Type, Speed Bin and Year, from the U.S. EPA's MOtor Vehicle Emission Simulator (MOVES).

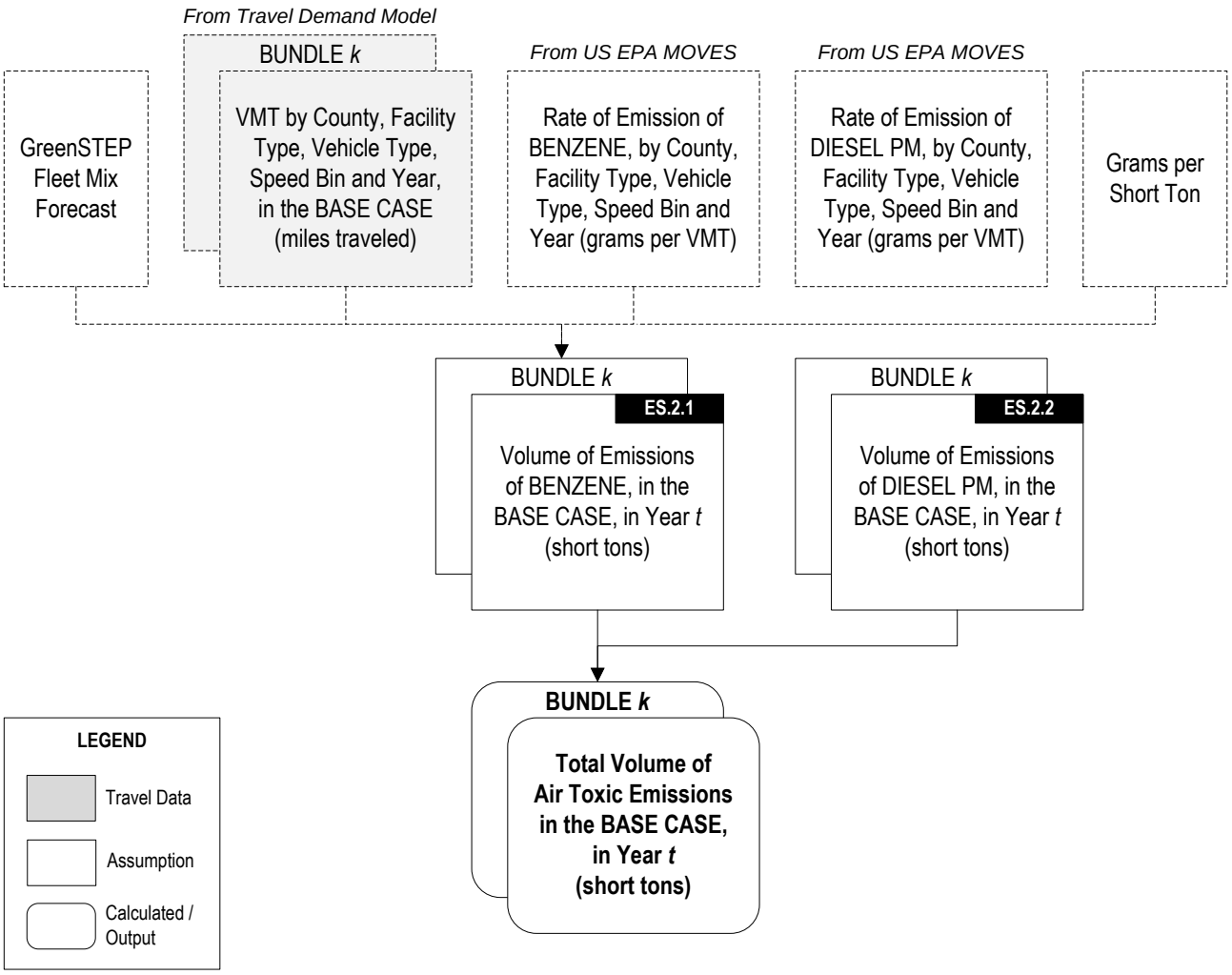
Fleet mix forecasts from RSPM (or other sources) should be combined with annual VMT projections developed from a travel demand model to obtain future VMT by vehicle type. Total emission volumes can then be estimated using emission rates from MOVES, as follows:

$$\textit{Annual Emissions} = \textit{Annual VMT}_{\textit{(miles)}} \times \textit{Emission rate}_{\textit{(grams/mile)}}$$

Information on the RSPM model can be found on the Oregon DOT website, at https://www.oregon.gov/ODOT/Planning/Pages/Technical-Tools.aspx#GreenSTEP (last accessed November 10, 2014).

User documents and tools related to MOVES are available on the EPA website at https://www.epa.gov/moves (last accessed November 10, 2014).

Figure ES.2.24: S&L Diagram for Estimating Emissions of Air Toxics



Data and Assumptions

This specific indicator (Air Toxics) must be estimated *outside* the Mosaic workbook for each bundle and for at least one forecast year. The value of the indicator must then be entered in the relevant table of the ENVIRONMENT worksheet: Rows 149-159 for Benzene (variable ES.2.1 in the S&L diagram) and Rows 164-174 for Diesel PM (variable ES.2.2).

The tool then calculates total emissions of air toxics (by simple summation) and determines a quantitative score based on differences in emissions across bundles. Alternatively, a qualitative score can be assigned directly to each bundle, using a scale of -5 to +5.

Category: Environmental Stewardship (ENVIRONMENT Worksheet)

General Indicator: Greenhouse Gazes

Specific Indicator:

ES.3 – Life-Cycle CO2e

This specific indicator is the total volume of CO2 emitted by vehicles under alternative planning options. It is a "well-to-wheel" measure that includes emissions from refining and transporting fuels.

Structure & Logic Diagram

The figure on the next page provides a graphical representation of the main variables and calculations required to develop Specific Indicator ES.3. The variables involved in operations performed *outside* the Mosaic workbook are represented in boxes with dotted lines. They include:

- Estimates of annual VMT by County and Year, in the base case and for each bundle; and
- Emission rates (in grams per VMT) from MOVES or RSPM; and
- Vehicle fleet mix forecasts from RSPM or other sources.

Total tons of emissions for lifecycle CO2 can be computed as the product of VMT by vehicle type (obtained from the travel demand model) and emissions rates in tons per VMT, from MOVES or RSPM.

Information on the RSPM model can be found on the Oregon DOT website, at https://www.oregon.gov/ODOT/Planning/Pages/Technical-Tools.aspx#GreenSTEP (last accessed November 10, 2014).

User documents and tools related to MOVES are available on the EPA website at https://www.epa.gov/moves (last accessed November 10, 2014).

In the Mosaic tool, the volumes of CO2 emissions are monetized with estimates of the Social Cost of Carbon (SCC) developed by the Interagency Working Group on the Social Cost of Carbon (IWGSCC)[10].

Annual monetized emission costs are estimated as:

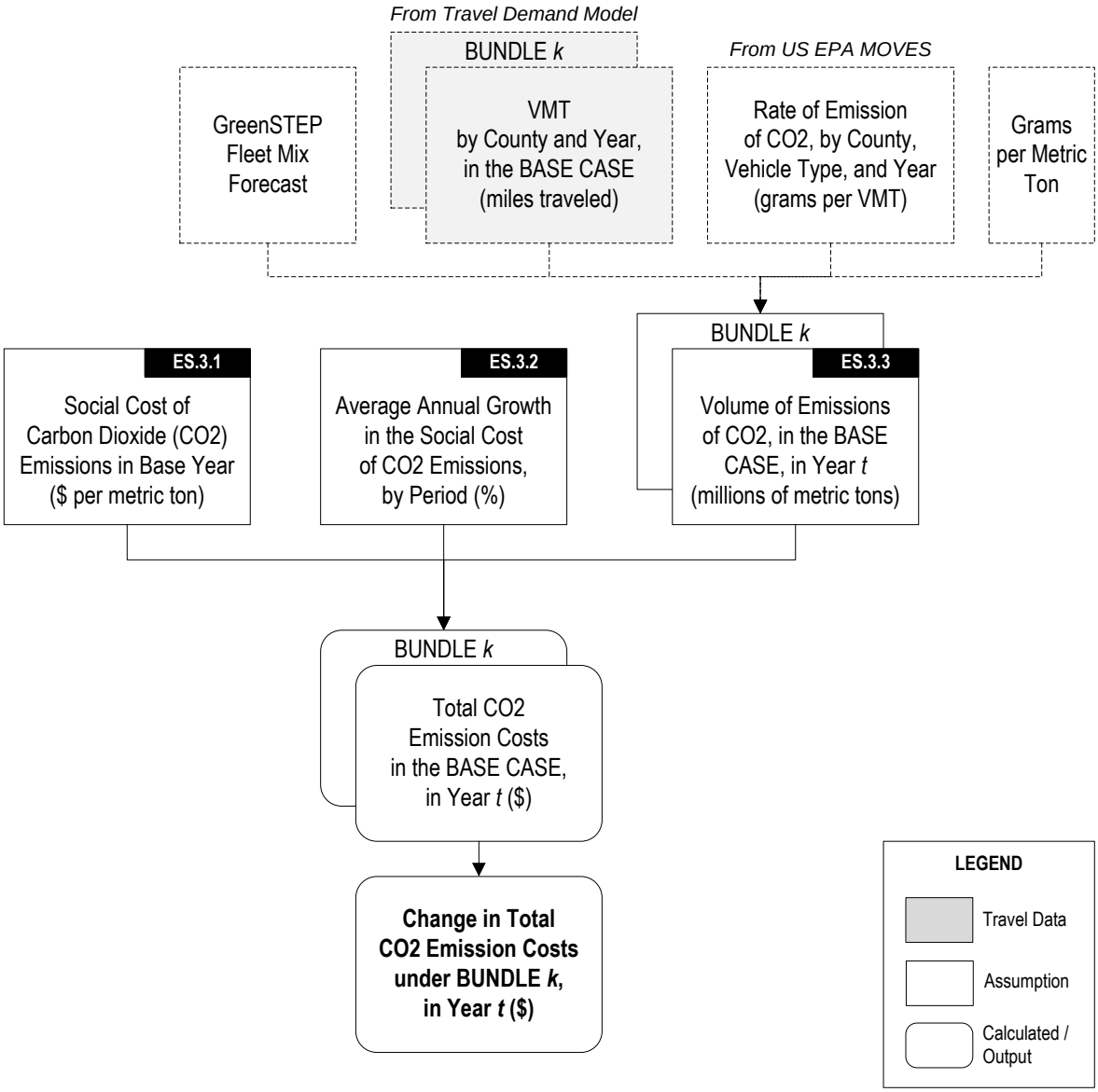
$$\textit{Annual CO2 Emission Costs} = \textit{Annual CO2 Emissions}_{(tons)} \times \textit{Social Cost of Carbon}_{(\$/ton)}$$

The SCC is an estimate of the monetized damages associated with an increase in carbon emissions in a given year. It is intended to include changes in net agricultural productivity, human health effects, property damages from increased flood risk, and the value of ecosystem services due to climate change.

[10] Interagency Working Group on Social Cost of Carbon, U.S. Government, For regulatory impact analysis under Executive Order 12866, Revised November 2013; http://www.whitehouse.gov/sites/default/files/omb/assets/inforeg/technical-update-social-cost-of-carbon-for-regulator-impact-analysis.pdf (last accessed November 10, 2014)

The SCC increases over time because future emissions are expected to produce larger incremental damages as physical and economic systems become more stressed in response to greater climatic change[11].

Figure ES.3.25: S&L Diagram for Estimating Changes in Life-Cycle CO2e Emissions



11 U.S. EPA, Regulatory Impact Analysis: Final Rulemaking for Model Year 2017-2025 Light-Duty Vehicle Greenhouse Gas Emission Standards and Corporate Average Fuel Economy Standards, August 2012, page 7-3.

Data and Assumptions

The table below provides information on the input variables used in the estimation of Specific Indicator ES.3. The input values that must be specified by Mosaic users are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table ES.3.15: Data and Assumptions for Estimating Changes in Life-Cycle CO2e

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
ES.3.1	Social Cost of Carbon Dioxide Emissions in Base Year	$ per metric ton	No	No	Default values provided in tool, based on estimates from IWGSCC	MODEL PARAMETERS, Rows 97-99
ES.3.2	Average Annual Growth in the Social Cost of CO2 Emissions, by Period	%	No	No	Default values provided in tool, based on estimates from IWGSCC	MODEL PARAMETERS, Rows 101-115
***ES.3.3**	Volume of Life-cycle Emissions of CO2, in Year t	Millions of metric tons	Yes	Yes	Must be specified by user (use of EPA's MOVES or RSPM recommended)	ENVIRONMENT, Rows 207-217

Category: Environmental Stewardship (ENVIRONMENT Worksheet)

General Indicator: Resources at Risk

Specific Indicator:

ES.4 – Natural, Built, and Cultural Resources at Risk

This specific indicator examines several factors to understand the "natural, built, and cultural resources at risk" including: (i) potential impacts to threatened and endangered (T&E) species; (ii) potential impacts on surface water; (iii) potential impacts on wetlands; (iv) potential risk of hazardous material being located within the plan footprint; and (v) the potential risk of crossing a local, state, or national park with special significance.

Structure & Logic Diagrams

The series of S&L diagrams provided below illustrates how Specific Indicator ES.4 may be developed. The variables involved in operations performed *outside* the Mosaic workbook are represented in boxes with dotted lines. In the first diagram, severity indices for all five sources of risk are combined together using weighting factors, to arrive at an aggregate (weighted average) severity index.

Figure ES.4.26: S&L Diagram for Estimating Natural, Built, and Cultural Resources at Risk

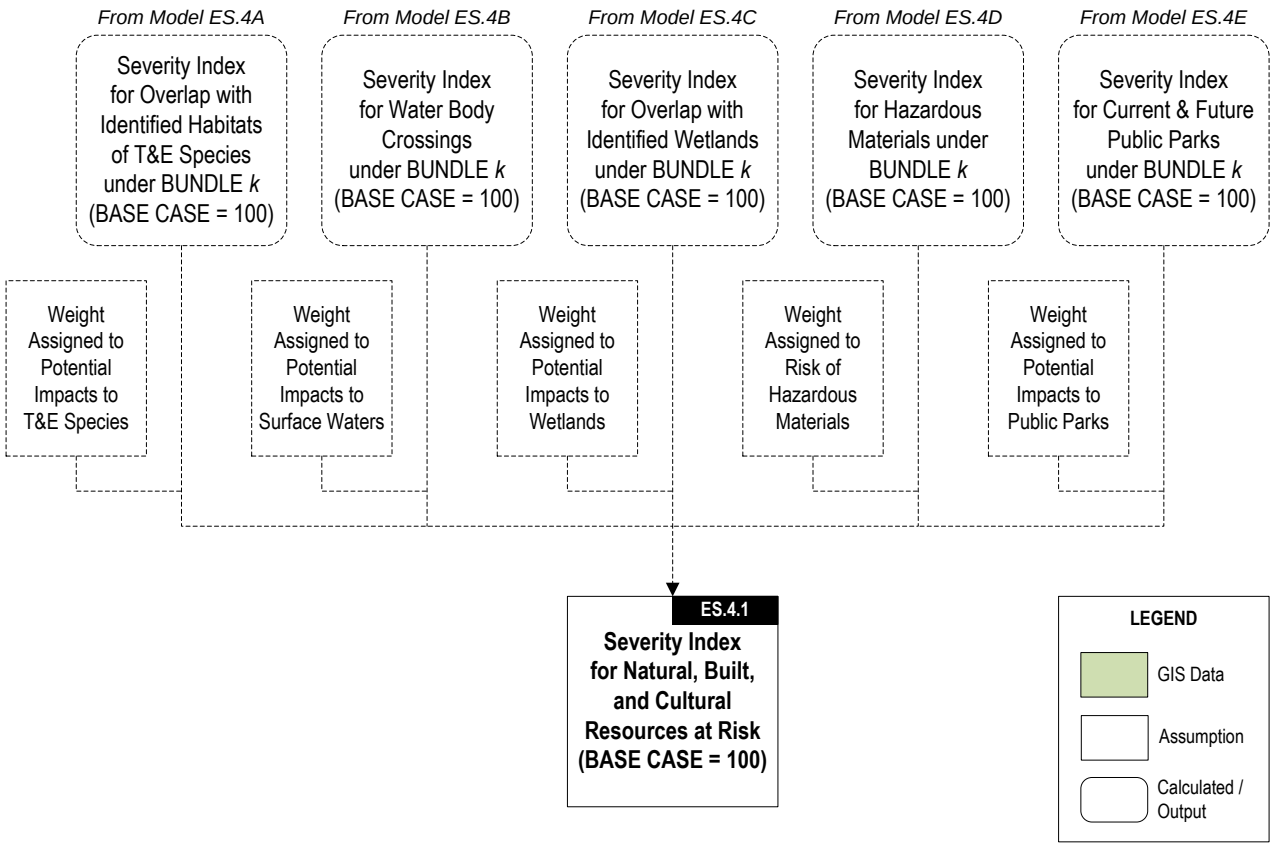


Figure ES.4.27: S&L Diagram for Estimating Potential Impacts to Threatened and Endangered Species

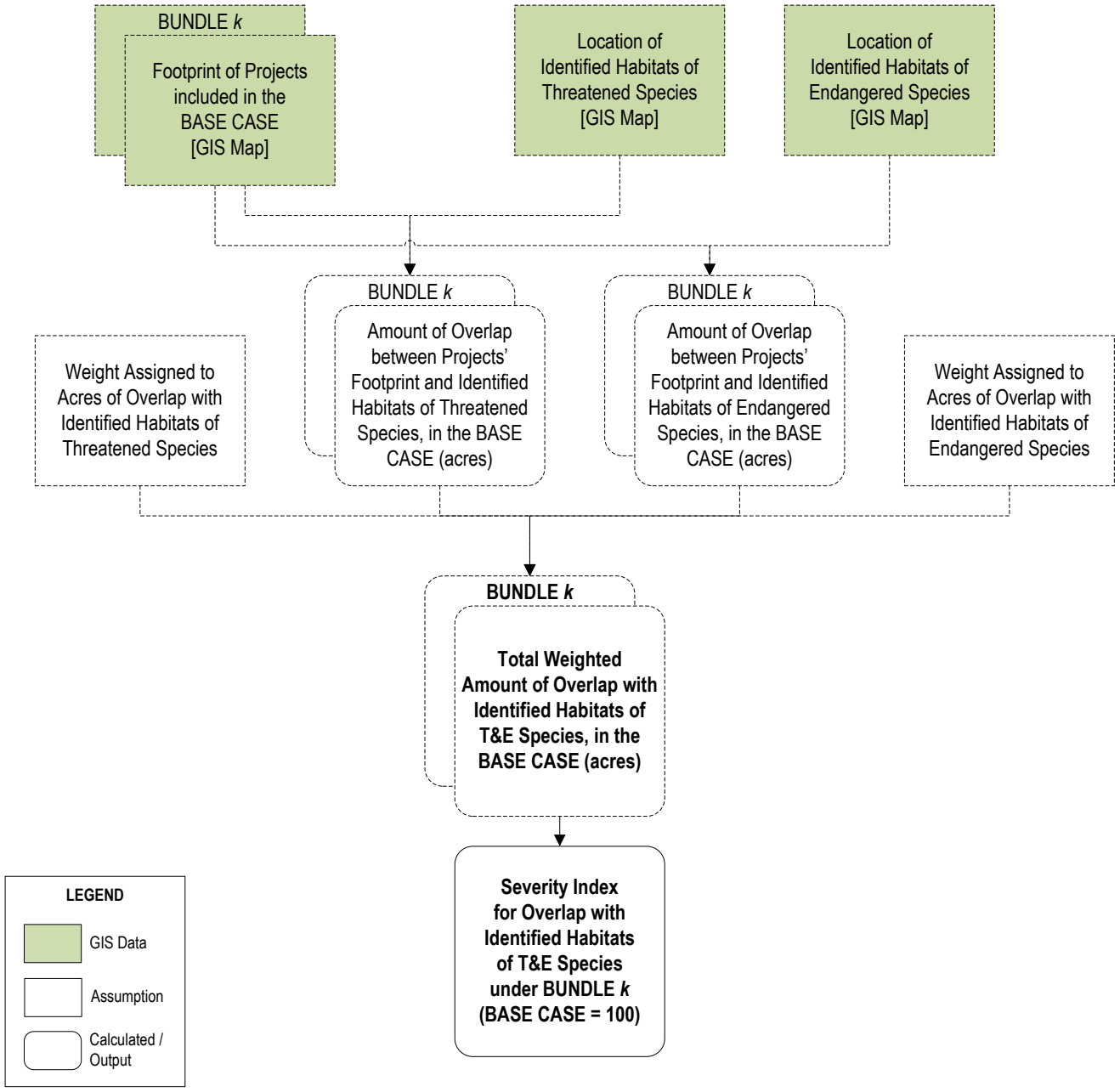


Figure ES.4.28: S&L Diagram for Estimating Potential Impacts to Surface Waters

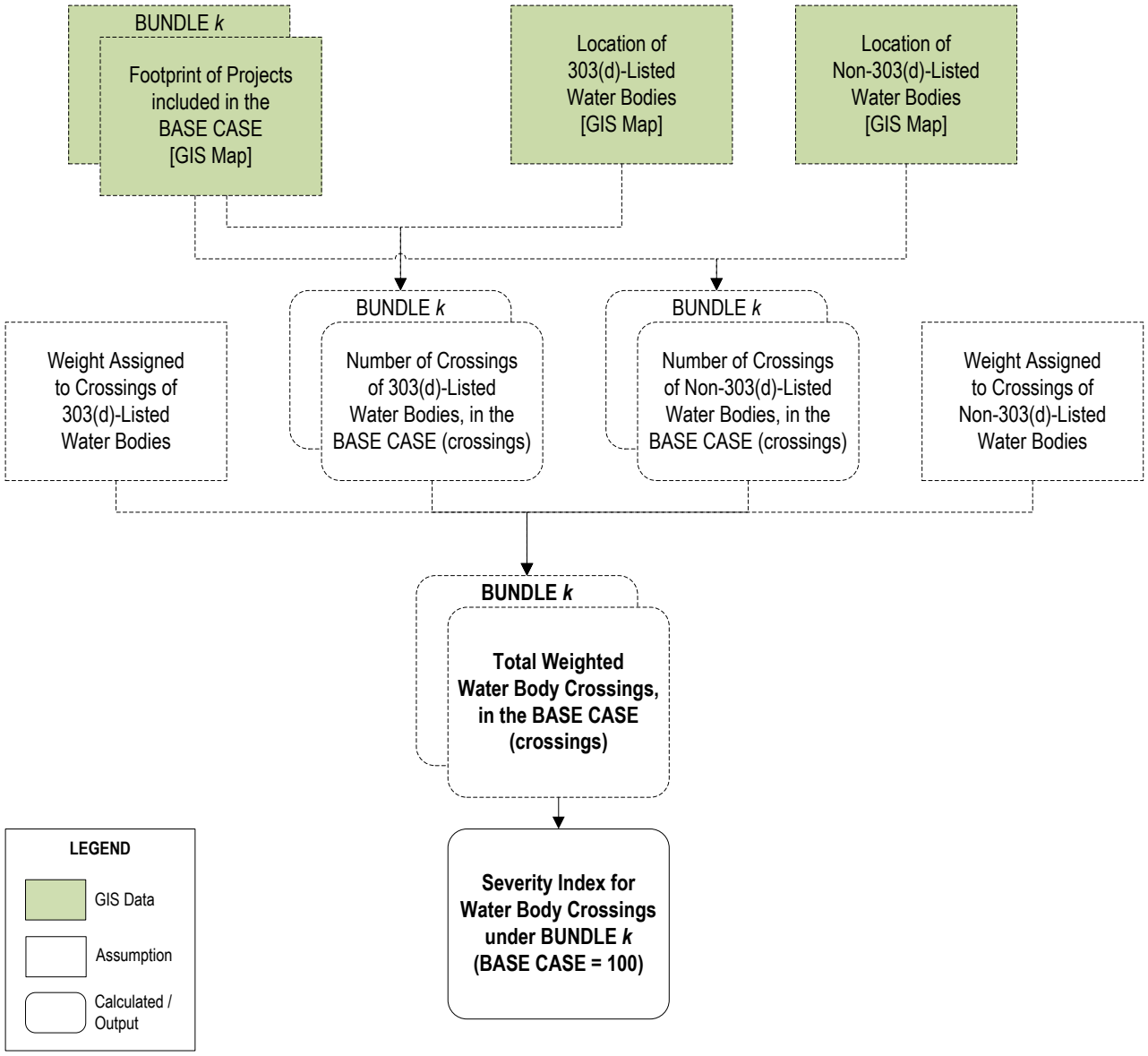


Figure ES.4.29: S&L Diagram for Estimating Potential Impacts to Wetlands

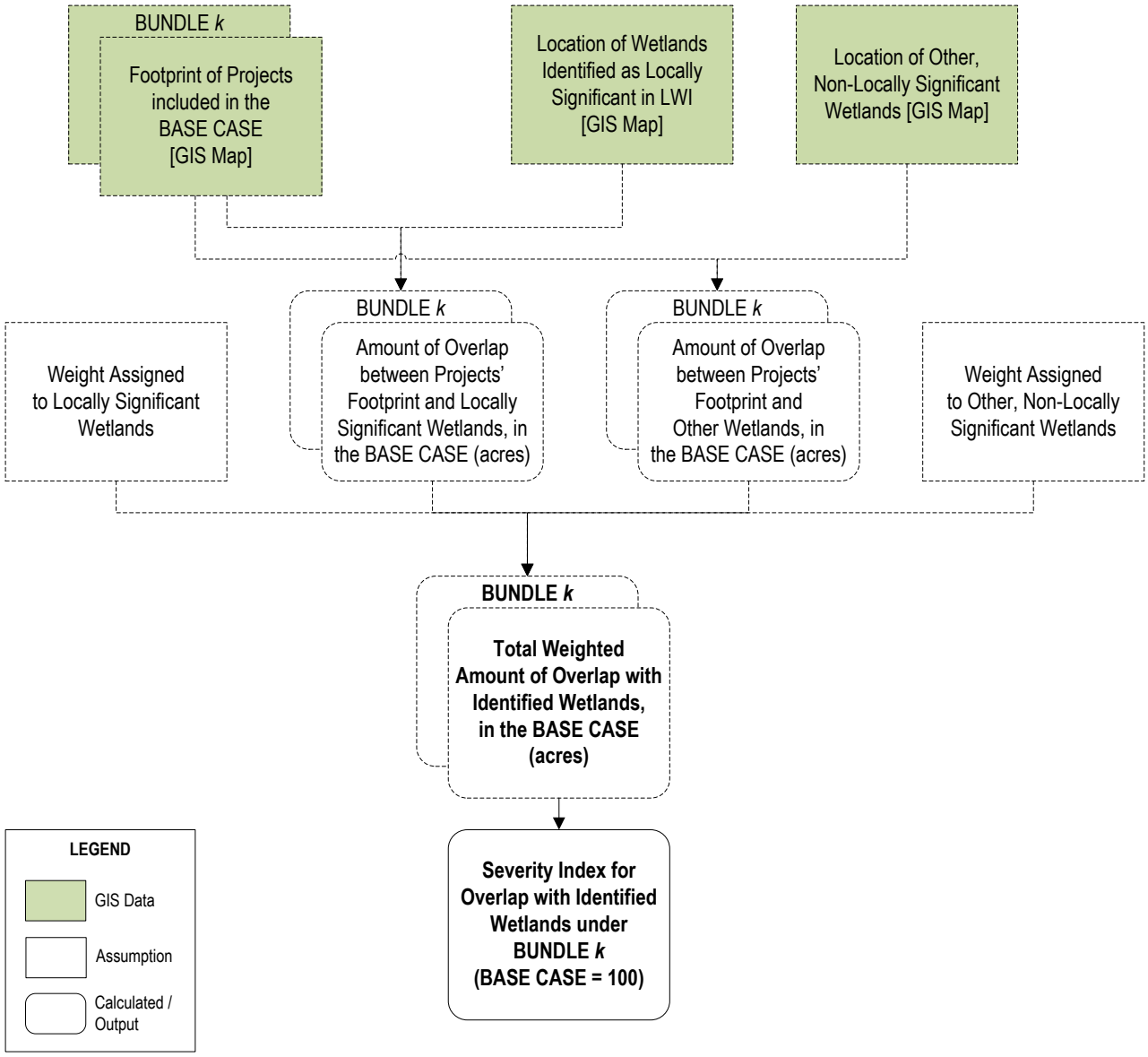


Figure ES.4.30: S&L Diagram for Estimating the Risk of Hazardous Materials in Project Footprint

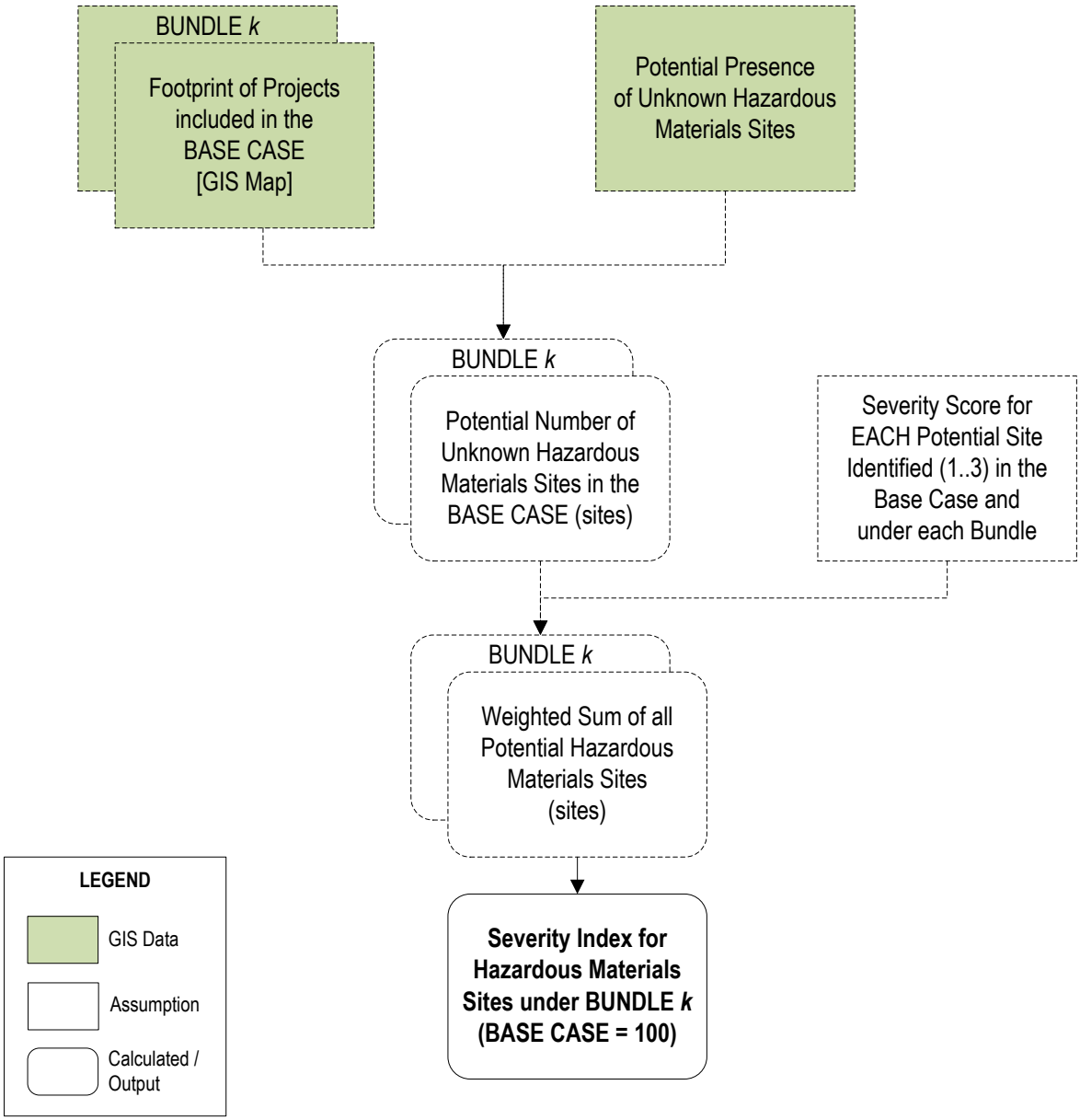
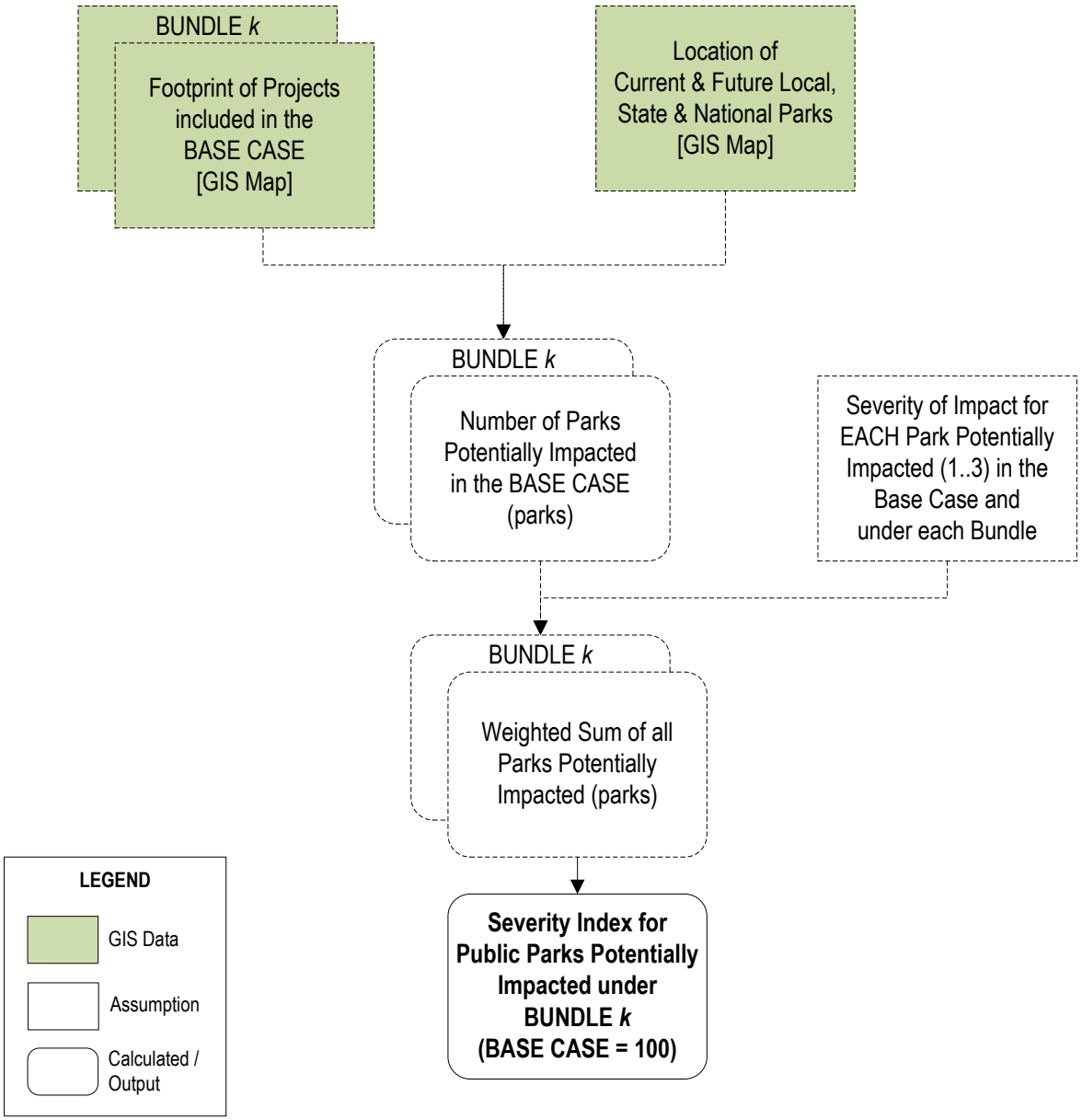


Figure ES.4.31: S&L Diagram for Estimating the Potential for Public Parks in Project Footprint



Data and Assumptions

This specific indicator (Natural, Built, and Cultural Resources at Risk) – and associated severity indices – must be estimated *outside* the Mosaic workbook, for each bundle and for at least one forecast year.

The value of the indicator (resulting from the application of weighting factors described in Figure ES.4.26) must be entered in the relevant table of the ENVIRONMENT worksheet (Rows 234-244). Alternatively, a qualitative "resources at risk" score may be directly assigned to each bundle using a scale of -5 to +5.

Funding the Transportation System & Finance

Documentation sheets for the following specific indicators can be found in this section:

- FT.1 – Capital Costs
- FT.2 – Other Life-Cycle Costs
- FT.3 – Total Revenue
- FT.4 – Share of Life-Cycle Funds that are “New” or “Recycled”
- FT.5 – Net Impact of Program on State and Local Fiscal Balance

Category: Funding the Transportation System & Finance (FUNDING Worksheet)

General Indicator: Capital Costs

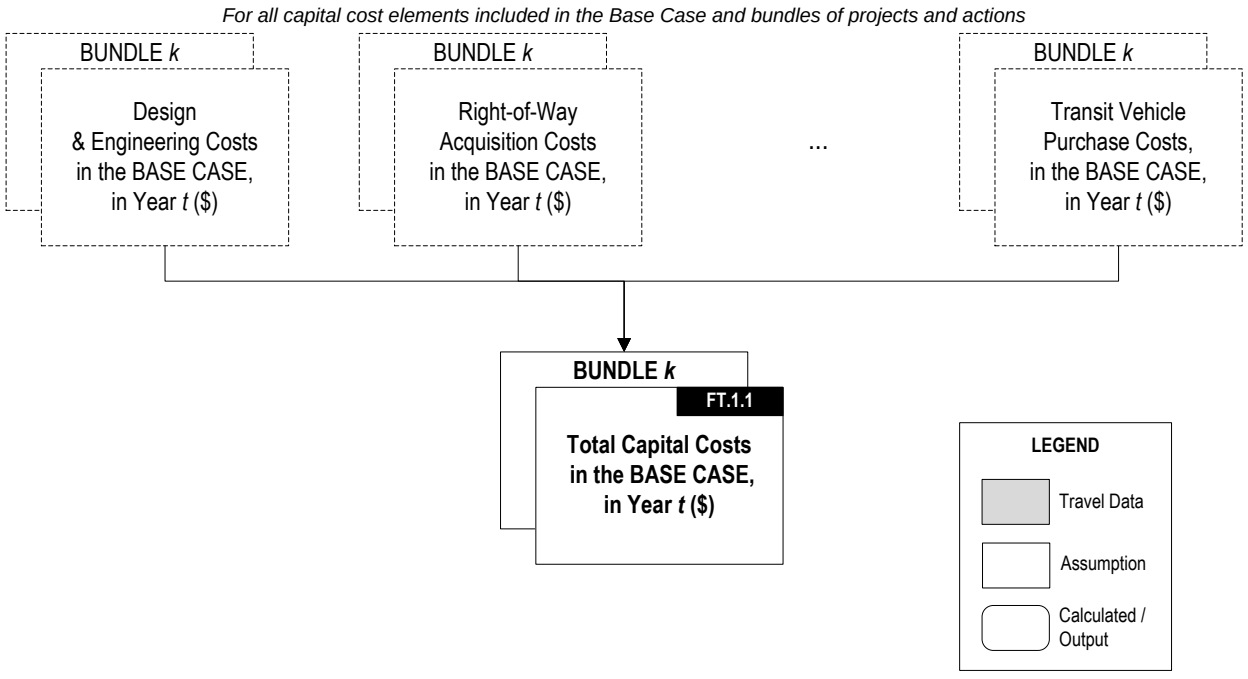
Specific Indicator: FT.1 – Capital Costs

This specific indicator is the sum of all fixed, capital expenses associated with a bundle, including design, engineering, permitting, right-of-way acquisition, construction, and equipment purchase costs. It must be estimated annually over the full period of analysis, and considering all projects and actions included within a bundle.

Structure & Logic Diagram

The figure below provides a graphical representation of the main variables and calculations required to develop estimates of capital costs. The variables involved in operations performed outside the Mosaic workbook are represented in boxes with dotted lines.

Figure FT.1.32: S&L Diagram for Estimating Total Capital



Individual capital cost estimates may be derived from engineering documents generated for specific bundle components, including preliminary design studies. Alternative data sources include: historical records of capital expenditures within ODOT and other agencies; comparable from the literature; and publicly available databases (such as the National Transit Database).

Data and Assumptions

Specific Indicator FT.1 is estimated directly from annual input values entered by the user in the COST & SCHEDULE worksheet of the Mosaic tool (Rows 12 to 22).

Category: Funding the Transportation System & Finance (FUNDING Worksheet)

General Indicator: Life-Cycle Costs

Specific Indicator:

FT.2 – Other Life-Cycle Costs

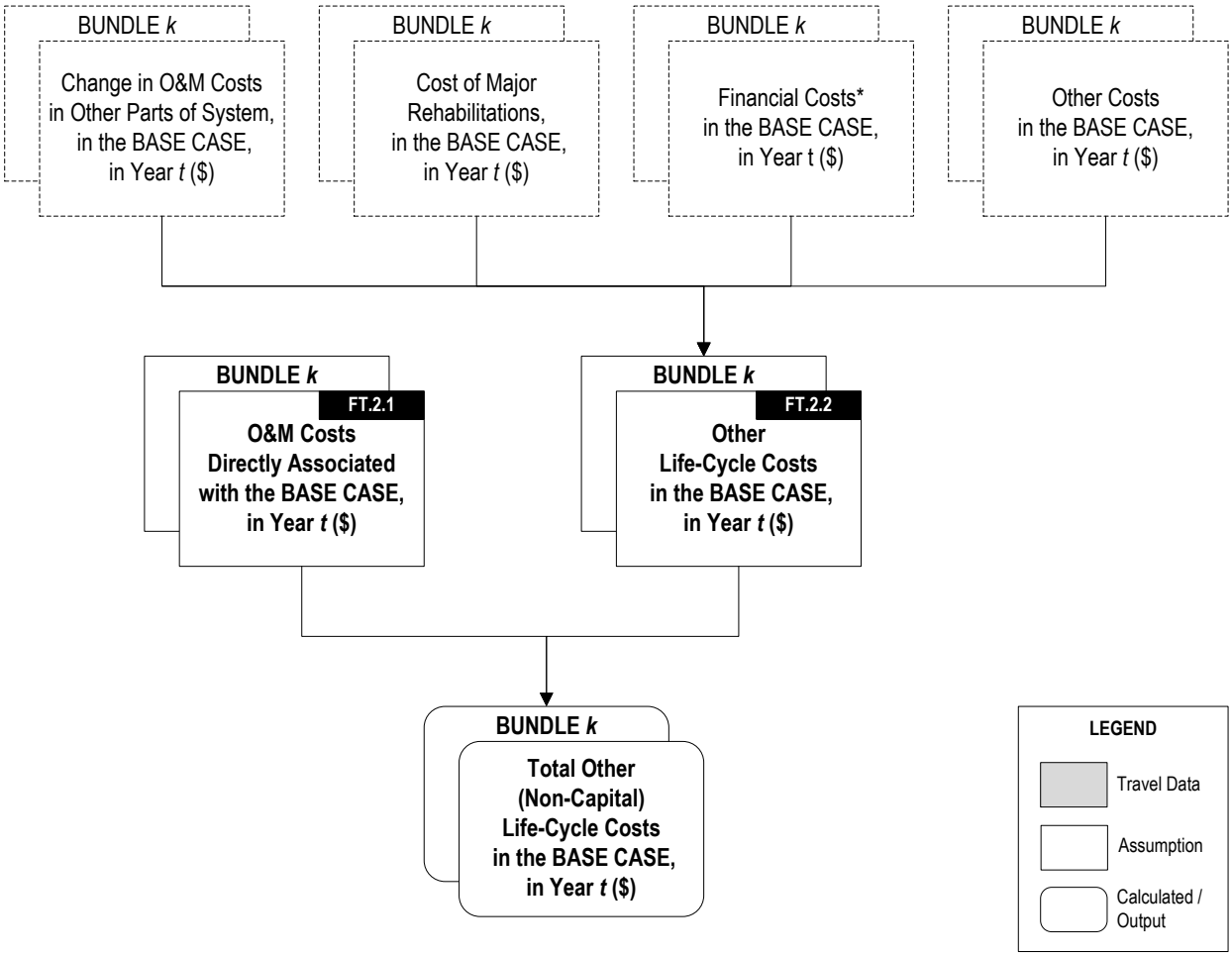
This specific indicator is the sum of all life-cycle costs, other than Capital Costs, associated with the projects and actions included within a bundle. These include annual operating and maintenance costs, major rehabilitation work, financial costs, and changes in operating and maintenance costs in other parts of the transportation system.

Structure & Logic Diagram

Figure FT.2.33 on the next page provides a graphical representation of the main variables and calculations required to develop estimates of Specific Indicator FT.2. The variables involved in operations performed outside the Mosaic workbook are represented in boxes with dotted lines.

Mosaic users are encouraged to follow existing guidance on Life-Cycle Cost Analysis (LCCA) to estimate this indicator. For example, users can consult *Life Cycle Cost Analysis Primer,* by the Office of Asset Management of the Federal Highway Administration. The primer is available at: https://www.fhwa.dot.gov/infrastructure/asstmgmt/lcca.cfm (last accessed November 10, 2014).

Figure FT.2.33: S&L Diagram for Estimating Other Life-Cycle Costs



** Must be reported separately, and excluded from estimation of NPV and other BCA indicators.*

Data and Assumptions

Total Other Life Cycle Costs are estimated as the sum of O&M Costs (variable FT.2.1 in the S&L diagram), plus other Life-Cycle Costs (variable FT.2.2). Both variables must be entered by users in the COST & SCHEDULE worksheet of the Mosaic tool (Rows 25-35 and 38-48).

Category: Funding the Transportation System & Finance (FUNDING Worksheet)

General Indicator: Operating Revenues

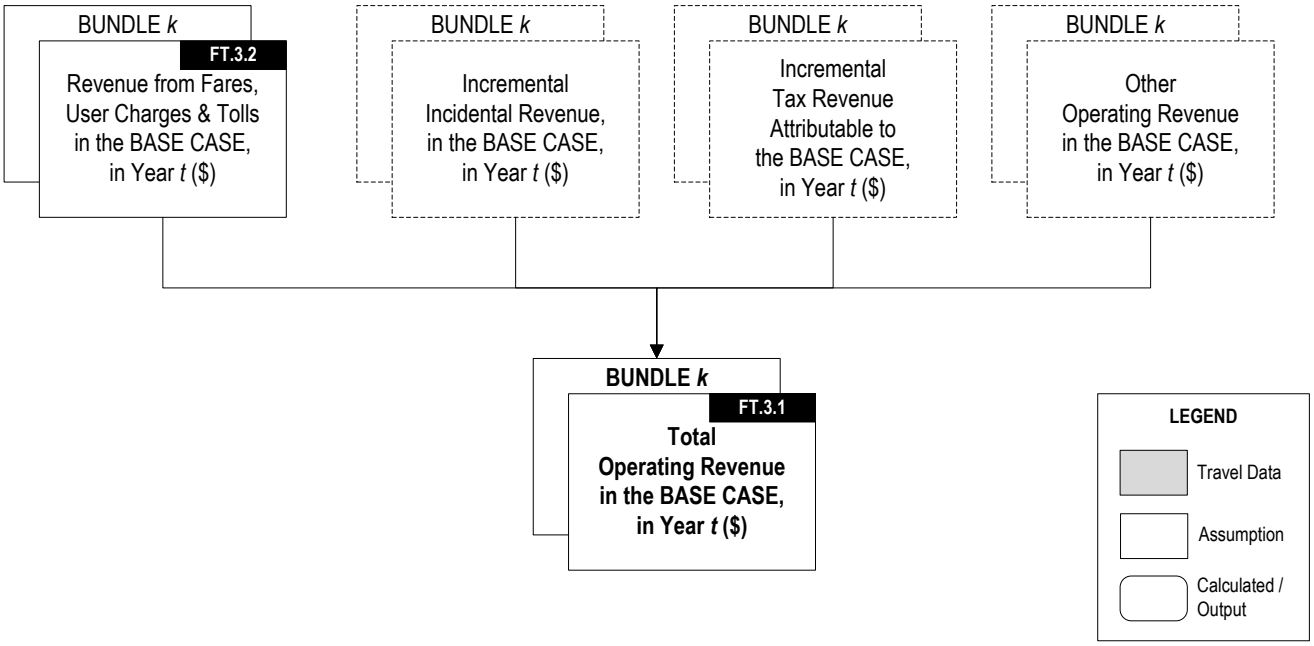
Specific Indicator: FT.3 – Total Revenue

This specific indicator, total operating revenue, is defined as the sum of incidental revenue (e.g., leasing of right-of-way); *plus* revenue from transit fares, user charges and tolls; *plus* changes in tax revenue resulting from the implementation of a bundle. As explained in the Mosaic Users' Guide, this indicator is only relevant for those bundles which contain dedicated user fees and charges (such as toll facilities, transit fares or other such fees), and/or have a significant impact of agency revenue.

Structure & Logic Diagram

The figure below provides a graphical representation of the main variables and calculations required to develop estimates of total operating revenue. The variables involved in operations performed outside the Mosaic workbook are represented in boxes with dotted lines.

Figure FT.3.34: S&L Diagram for Estimating Total Revenue



Data and Assumptions

This indicator is estimated directly from input values specified by the user in the COST & SCHEDULE worksheet of the Mosaic tool, Rows 68 through 78 (for variable FT.3.1 in the S&L diagram) and Rows 81 through 91 (for variable FT.3.2).

Category: Funding the Transportation System & Finance (FUNDING Worksheet)

General Indicator: Leveraging Funds from Private Sector and Other Agencies

Specific Indicator:

FT.4 – Share of Life-Cycle Funds that are "New" or "Recycled"

This specific indicator examines the relative contribution of "new" funds (committed by the private sector or newly generated by local public agencies) and/or "recycled" funds in total bundle costs. It is defined as the ratio of New and Recycled Funds, over total life-cycle funds.

Structure & Logic Diagram

The figure below provides a graphical representation of the main variables and calculations required to develop the indicator. The variables involved in operations performed *outside* the Mosaic workbook are represented in boxes with dotted lines.

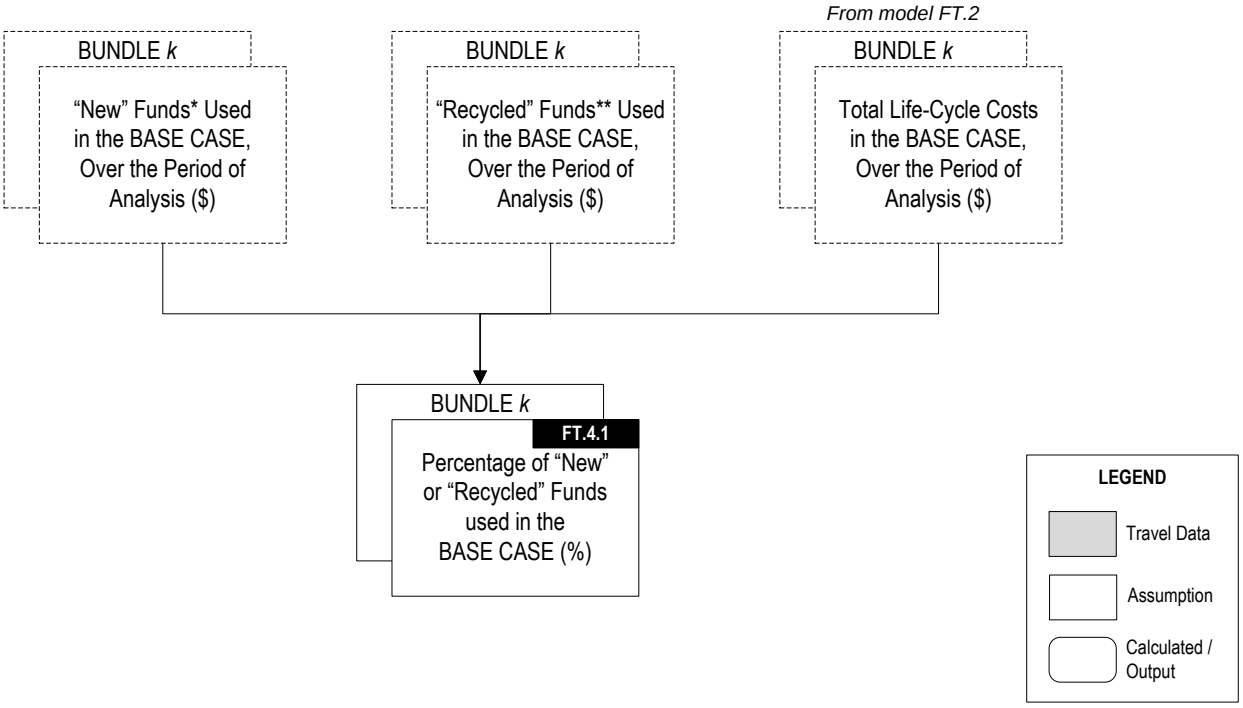
As illustrated in the S&L diagram, FT.4 is estimated as follows:

$$Percentage = [New\ Funds_{(\$millions)} + Recycled\ Funds_{(\$millions)}] / Total\ Funds_{(\$millions)}$$

Where:

- *New Funds* are funds committed by private investors (e.g., capital costs raised directly by a private owner/operator of a toll road or transit service) or fresh funds generated specifically to implement a bundle by a local public agency (i.e., taxes, fees, charge or levies which do not exist today);
- *Recycled Funds* are funds redirected from other uses, within an existing budget (e.g., contributions of local or regional governments from a revolving loan fund); and
- *Total Funds* is the sum of all the funds required to implement a bundle. It is equivalent to total life-cycle costs, and is estimated as the sum of Specific Indicator FT.1 and Specific Indicator FT.2.

Figure FT.4.35: S&L Diagram for Estimating the Share of Life-Cycle Funds that are "New" or "Recycled"



** New funds originating outside the State or Local budget (e.g., private funds, new federal grant)*

*** Funds recycled within the State or Local budget, with no additional fiscal burden*

Data and Assumptions

This specific indicator (Share of Life-Cycle Funds that are "New" or "Recycled") must be estimated *outside* the Mosaic workbook, using the approach outlined in the above chart.

The value of the indicator is then entered in the relevant table of the FUNDING worksheet (Rows 82-92). Alternatively, a qualitative score can be assigned to each bundle, using a scale of -5 to +5.

Category: Funding the Transportation System & Finance (FUNDING Worksheet)

General Indicator: Net Impact on State and Local Fiscal Balance and Debt

Specific Indicator:

FT.5 – Net Impact of Program on State and Local Fiscal Balance

This indicator measures the gap between total funds available (i.e., "new" funds plus "recycled" funds plus operating revenue) and the total life-cycle costs of a bundle.[12] In the current version of the Mosaic tool, users are only able to score this indicator qualitatively.

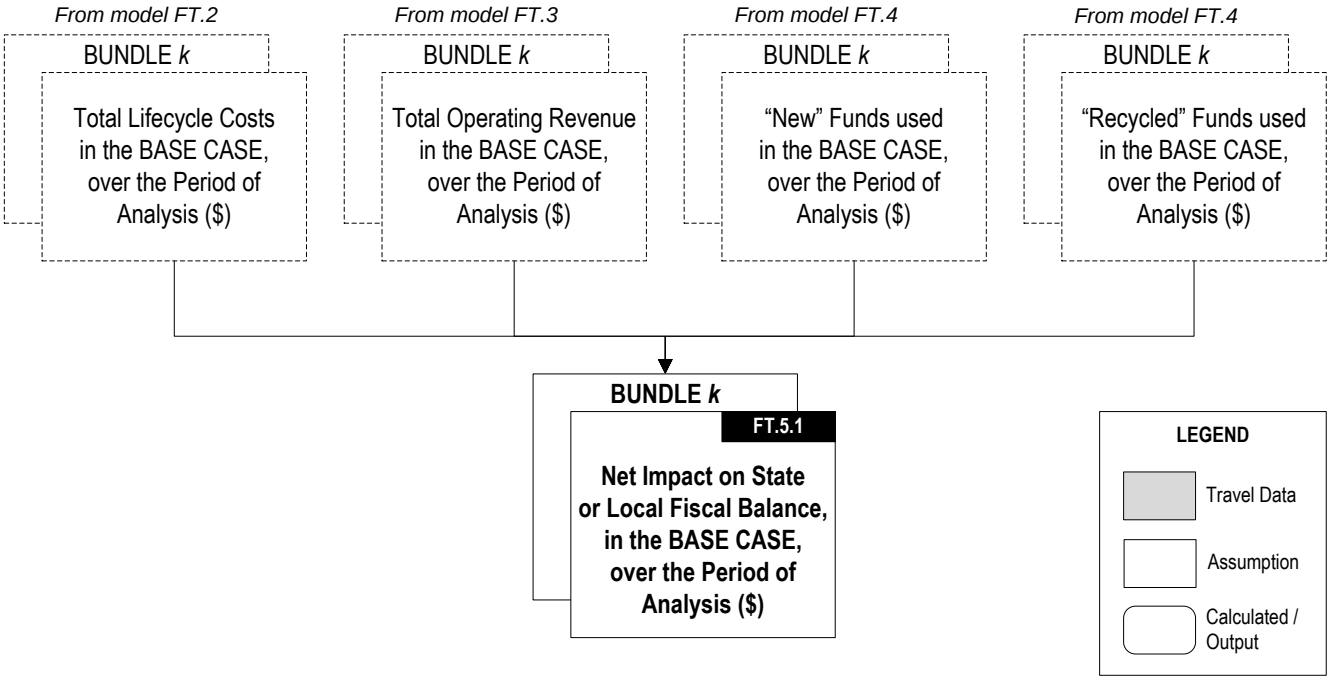
Structure & Logic Diagram

The figure below provides a graphical representation of the main variables and calculations required to develop the indicator. The variables involved in operations performed outside the Mosaic workbook are represented in boxes with dotted lines.

The indicator may be developed as follows:

Fiscal Impact = New Funds + Recycled Funds + Total Operating Revenue – Total Life-Cycle Cost

Figure FT.5.36: S&L Diagram for Estimating the Net Impact of Program on State and Local Fiscal Balance



[12] This indicator is only relevant under unusual financial circumstances (e.g., when the revenues for a bundle may affect the funding agency's credit rating, or when expenditures are affected by "compression" under Oregon Measure 5 (see https://www.oregon.gov/DOR/programs/property/Pages/property-taxes.aspx).

Data and Assumptions

The net impact of a bundle on State and local fiscal balances may be estimated *outside* the Mosaic workbook using the approach outlined in the S&L diagram.

In this version of the tool, however, the use of a qualitative score is recommended. The following table is provided in the tool (FUNDING & FINANCE worksheet, Rows 125-136) to guide users in the determination of a score.

EXPECTED IMPACT OF BUNDLE	ORDER OF MAGNITUDE	SCORE
Large Adverse	<-3%	-5
		-4
Moderate Adverse	-3 to -1%	-3
		-2
Slight Adverse	-1 to 0%	-1
Neutral	0%	0
Slight Beneficial	0 to 1%	1
Moderate Beneficial	1 to 3%	2
		3
Large Beneficial	> 3%	4
		5

Source: Mosaic Tool, Version 2.0, FUNDING & FINANCE worksheet

Safety & Security

Documentation sheets for the following specific indicators can be found in this section:

- SA.1 – Fatal, Injury A, and Injury B Crashes
- SA.2 – Property Damage Only (PDO) Accidents
- SA.3 – Emergency Management Systems Response Times
- SA.4 – Resiliency of the Network

Category: Safety & Security
(SAFETY Worksheet)

General Indicator: System Safety

Specific Indicator: SA.1 – Fatal, Injury A, and Injury B Crashes

This specific indicator is the total number of Fatal, Injury A (incapacitating) and Injury B (non-incapacitating) crashes, across all modes and for the entire study area.

Structure & Logic Diagram

Figure SA.1.37 provides a graphical representation of the main variables and calculations required to estimate this indicator and related metrics. The variables involved in operations performed outside the Mosaic workbook are represented in boxes with dotted lines.

It is expected that Mosaic users will develop estimates of the number of fatalities, incapacitating and non-incapacitating injuries (variables SA.1.1, SA.1.2, and SA.1.3) with the Mosaic Safety Tool available on the Mosaic website, or other resources *external* to the Mosaic workbook. Information on the Mosaic Safety Tool is available in the Mosaic User's Guide (Table 4 – Instructions for Using the Mosaic Safety Tool).

Estimates of the Value of Statistical Life (VSL) developed by the US Department of Transportation, Office of the Secretary (Treatment of the Value of Preventing Fatalities and Injuries in Preparing Economic Analyses, Revised Departmental Guidance 2013) are used in the Mosaic workbook to monetize fatalities and injuries.

The VSL and associated injury values used in Mosaic are willingness-to-pay measures that include productivity losses, pain, suffering, and lost quality of life (including "psychic disutility") as well as the portion of medical expenses paid for by individuals. They do not include medical expenses paid through societal mechanisms (such as insurance, tax-supported welfare programs, and charity), nor property damage and traffic delay.

The constant-dollar value of VSL is expected to grow over time, with real income levels. The following formula is used in the Mosaic workbook to estimate VSL in future years:

$$VSL_{Year\ t} = VSL_{Year\ t-1} \times (1 + Real\_Income\_Growth)^{Income\_Elasticity}$$

Where:

- $VSL_{Year\ t}$ is the value of statistical life in Year t;
- $VSL_{Year\ t-1}$ is the value of statistical life in Year t- 1;
- *Real_Income_Growth* is the annual growth rate of labor productivity or real income per capita; and
- *Income_Elasticity* is the elasticity of VSL with respect to changes in real income.

The costs of incapacitating (A) and non-incapacitating (B) injuries are derived from the estimates of VSL, using relative disutility factors by injury severity level, using the following approach:

$$Injury\ Cost_{Year\ t} = VSL_{Year\ t} \times Relative\ Disutility\ Factor$$

Where:

- *Injury Cost* $_{Year\ t}$ is the value of preventing an injury in Year t;
- *VSL* $_{Year\ t}$ is the value of statistical life in Year t; and
- *Relative Disutility Factor* is a coefficient representing a fraction of VSL.

The relative disutility factors used in Mosaic are those provided in the US DOT guidance document. They are reproduced in the table below.

Severity Level	Fraction of VSL
AIS 0 - No Injury	0.000
AIS 1 - Minor injury	0.003
AIS 2 - Moderate injury	0.047
AIS 3 - Serious injury	0.105
AIS 4 - Severe injury	0.266
AIS 5 - Critical injury	0.593
AIS 6 - Fatality	1.000

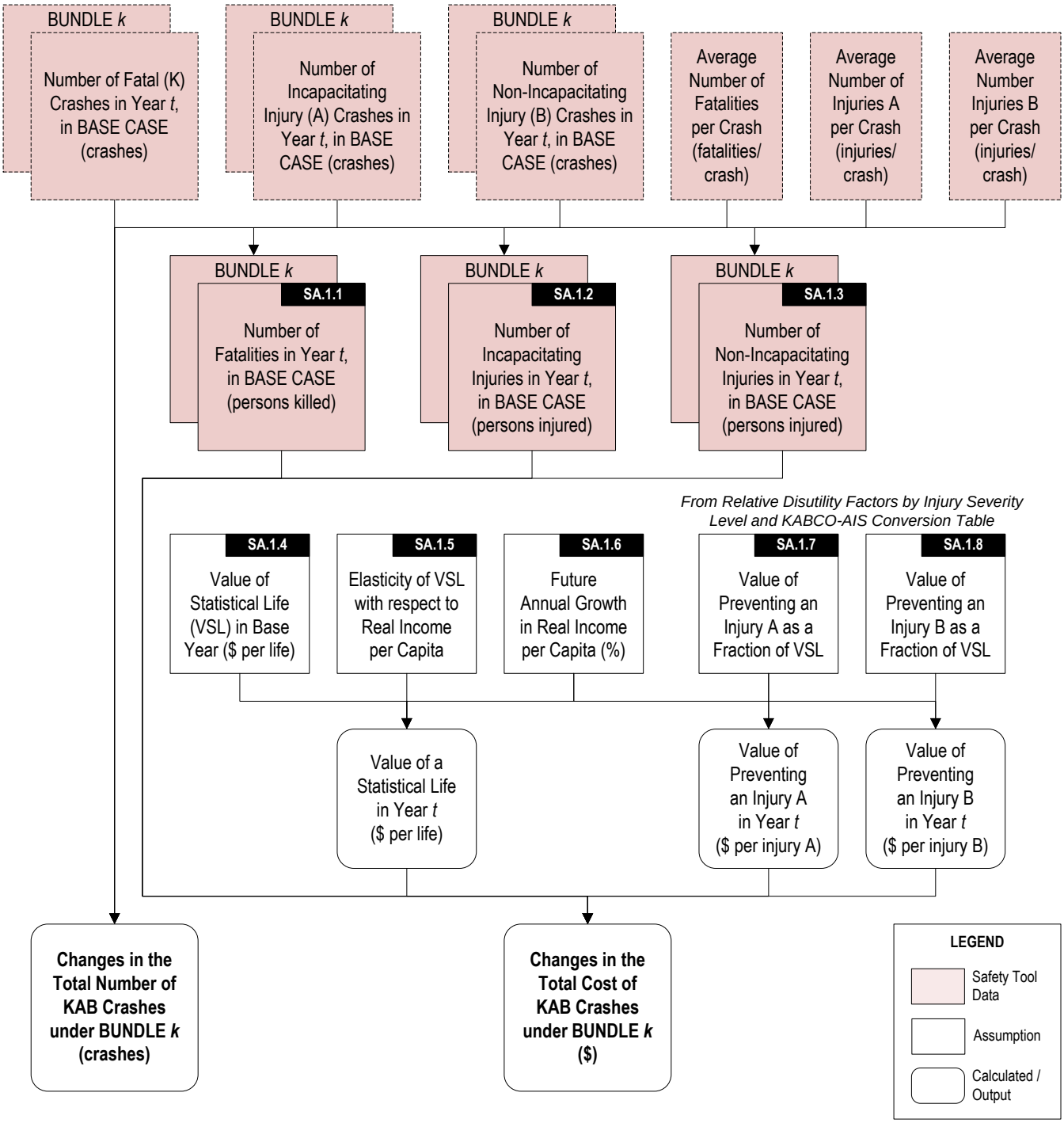
Source: US DOT (2013), Table 2

The severity levels set forth by US DOT are coded according to the Abbreviated Injury Scale (AIS). Injury A and Injury B, on the other hand, are severity levels defined along the KABCO scale. The following table can be used to convert AIS into KABCO.

	O	C	B	A	K	Injured Severity Unknown	Unknown if Injured
	No Injury	Possible Injury	Non Inca-pacitating	Inca-pacitating	Killed		
AIS 0 - No Injury	0.92534	0.23437	0.08347	0.03437	0.00000	0.21538	0.43676
AIS 1 - Minor injury	0.07257	0.68946	0.76843	0.55449	0.00000	0.62727	0.41739
AIS 2 - Moderate injury	0.00198	0.06391	0.10898	0.20908	0.00000	0.10400	0.08872
AIS 3 - Serious injury	0.00008	0.01071	0.03191	0.14437	0.00000	0.03858	0.04817
AIS 4 - Severe injury	0.00000	0.00142	0.00620	0.03986	0.00000	0.00442	0.00617
AIS 5 - Critical injury	0.00003	0.00013	0.00101	0.01783	0.00000	0.01034	0.00279
AIS 6 – Fatality	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000
Total	**1.00000**	**1.00000**	**1.00000**	**1.00000**	**1.00000**	**0.99999**	**1.00000**

Source: US DOT, TIGER Benefit-Cost Analysis (BCA) Resource Guide, https://www.transportation.gov/policy-initiatives/tiger/tiger-benefit-cost-analysis-bca-resource-guide (last accessed November 10, 2014)

Figure SA.1.37: S&L Diagram for Estimating Fatal, Injury A, and Injury B Crashes



Data and Assumptions

The table below provides information on the input variables used in the estimation of Specific Indicator SA.1. The input values that must be specified by users of Mosaic are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table SA.1.16: Data and Assumptions for Estimating Fatal, Injury A, and Injury B Crashes

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***SA.1.1**	Number of Fatalities, in Year t	Fatalities	Yes	Yes	Must be specified by user (with Mosaic Safety Tool or other external resources)	SAFETY, Rows 14-24
***SA.1.2**	Number of Incapacitating Injuries, in Year t	Injuries	Yes	Yes	Must be specified by user (with Mosaic Safety Tool or other external resources)	SAFETY, Rows 29-39
***SA.1.3**	Number of Non-Incapacitating Injuries, in Year t	Injuries	Yes	Yes	Must be specified by user (with Mosaic Safety Tool or other external resources)	SAFETY, Rows 44-54
SA.1.4	Value of Statistical Life (VSL) in Base Year	$ per life	No	No	Default parameter value provide in tool, based on US DOT guidance	MODEL PARAMETERS, Rows 118-120
SA.1.5	Elasticity of VSL with respect to Real Income per Capita	n/a	No	No	Default parameter value provided in tool, based on US DOT guidance	MODEL PARAMETERS, Rows 122-124
SA.1.6	Future Annual Growth in Real Income per Capita	% per year	No	No	Default value provided in tool, based on US DOT recommendations	MODEL PARAMETERS, Rows 56-58
SA.1.7	Value of Preventing an Injury A as a Fraction of VSL	%	No	No	Default value provided in tool, based on US DOT recommendations	SUPPORTING DATA, Cell E147
SA.1.8	Value of Preventing an Injury B as a Fraction of VSL	%	No	No	Default value provided in tool, based on US DOT recommendations	SUPPORTING DATA, Cell E148

Category: Safety & Security (SAFETY Worksheet)

General Indicator: System Safety

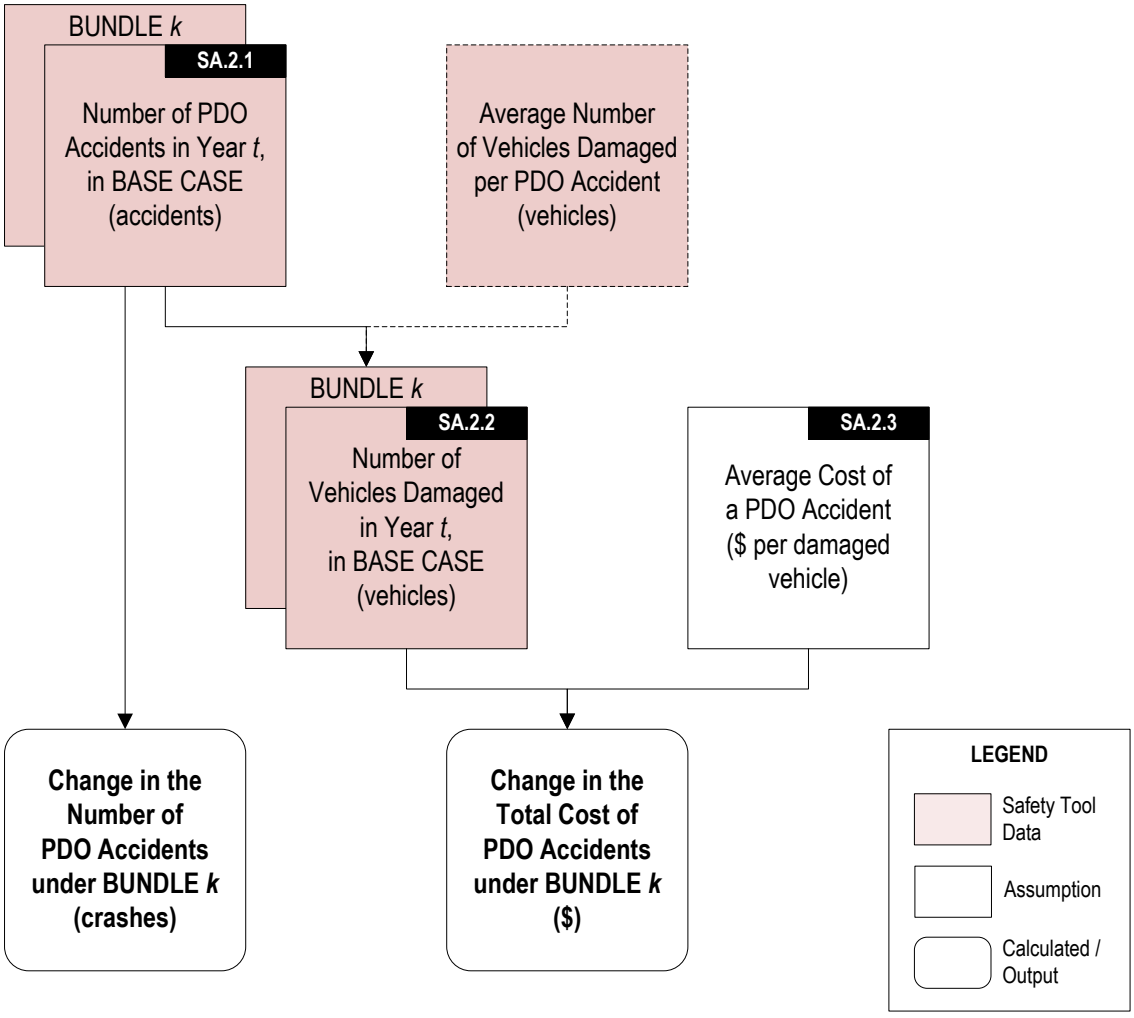
Specific Indicator: SA.2 – Property Damage Only Accidents

This specific indicator is the number of Property Damage Only (PDO) crashes, across all modes and for the entire study area.

Structure & Logic Diagram

Figure SA.2.38 is a graphical representation of the variables and calculations required to estimate this indicator. As with Indicator SA.1, it is expected that Mosaic users will develop estimates of the number of PDO accidents (variable SA.2.1) or, preferably, the number of vehicles damaged in a PDO accident (variable SA.2.2) with the Mosaic Safety Tool or other external resources.

Figure SA.2.38: S&L Diagram for Estimating Property Damage Only Accidents



Data and Assumptions

The table below provides information on the variables used in the estimation of SA.2. The input values that must be specified by Mosaic users are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table SA.2.17: Data and Assumptions for Estimating Property Damage Only Accidents

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***SA.2.1**	Number of PDO Accidents in Year t	Accidents	Yes	Yes	Must be specified by user (with Mosaic Safety Tool or other external resources)	SAFETY, Rows 103-113
***SA.2.2**	Number of Vehicles Damaged in Year t	Vehicles	Yes	No	Must be specified by user (with Mosaic Safety Tool or other external resources)	SAFETY, Rows 103-113
SA.2.3	Average Cost of a PDO Accident	$ per damaged vehicle	No	No	Default parameter value provided in tool, based on US DOT recommendations	MODEL PARAMETERS, Rows 134-136

Category: Safety & Security (SAFETY Worksheet)

General Indicator: System Security

Specific Indicator:

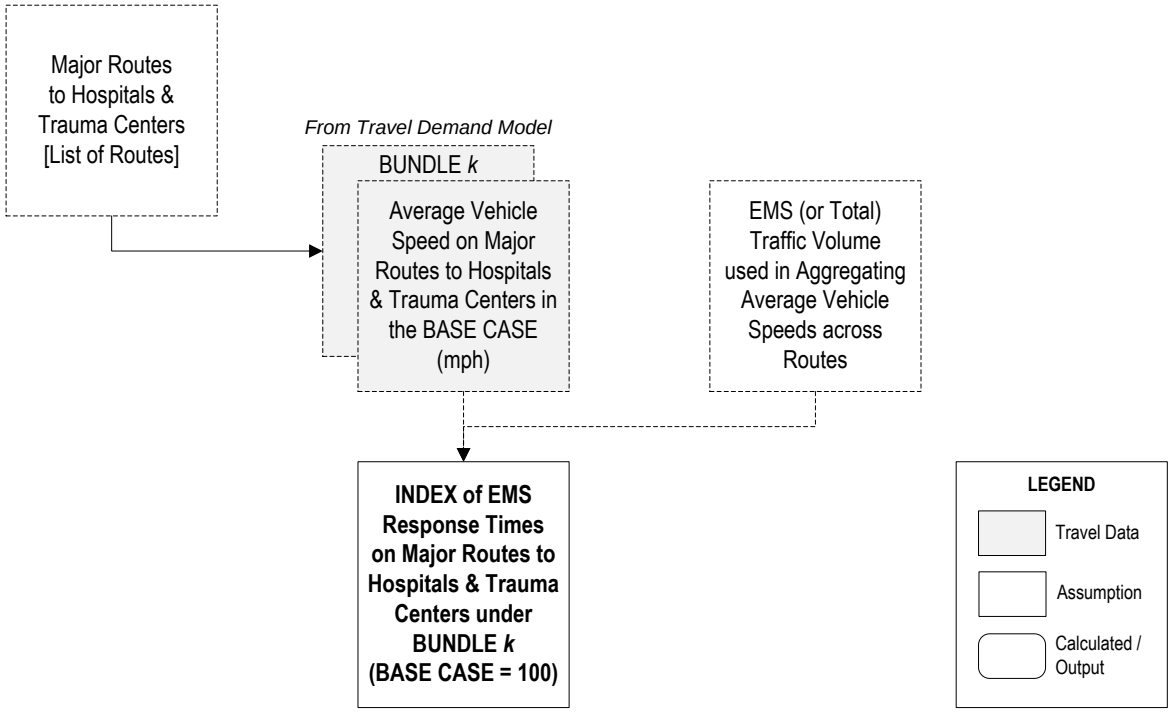
SA.3 – Emergency Management Systems Response Times

This specific indicator examines the change in Emergency Management Systems (EMS) response times by bundle of actions. This measure does not represent actual response time. Rather, the indicator is intended to capture differences in emergency vehicle travel times, on selected major routes.

Structure & Logic Diagram

The figure below provides a graphical representation of the main variables and calculations required to develop the indicator. The variables involved in operations performed *outside* the Mosaic tool are represented in boxes with dotted lines.

Figure SA.3.39: S&L Diagram for Estimating Emergency Management Systems Response Times



Mosaic users would first need to identify major routes to police and fire stations, hospitals, and trauma centers located within the study area. Output from a travel demand model would then be used to estimate the change in vehicle speeds and/or travel times on these major routes, under alternative planning options. These estimates would be weighted with traffic volumes to develop a summary measure of the expected change in EMS response times associated with a bundle. This measure would be expressed as an index, with a value of 100 in the Base Case.

An alternative approach for estimating this indicator is described in the Mosaic Users' Guide (Step 4: Populating the Mosaic Tool, page 76). In this alternative approach, users would first identify emergency facilities located within the study area. They would then use GIS capabilities and output from a travel demand model to determine the size of the geographic area within 5 minutes of an EMS facility under alternative planning options. The EMS Response Time index for a given bundle *k* would then be calculated as *Area Size under Bundle k / Area Size in the Base Case.* Larger values of the index would be associated with higher scores.

Data and Assumptions

This specific indicator (EMS Response Times index) must be estimated *outside* the Mosaic workbook for each bundle and for at least one forecast year. Users must then enter the value of the indicator in the relevant table of the SAFETY worksheet, for quantitative scoring and/or reporting. Alternatively, a System Security score can be assigned directly to each bundle (using qualitative scoring).

Category: Safety & Security (SAFETY Worksheet)

General Indicator: System Security

Specific Indicator:

SA.4 – Resiliency of the Network

This specific indicator examines the resiliency of the transportation network, defined as the ability to maintain critical operations in case of a natural event. It is scored qualitatively using a scale of -5 to +5.

The assessment of network resiliency under alternative planning options would rely on GIS data and capabilities *outside* the Mosaic tool. This would be done with a series of GIS layers, including:

- Identification/listing of lifeline routes;
- Location of hospitals and medical emergency centers in relation to these lifeline routes;
- Current condition of the lifeline routes; and
- Assessment of what damage would occur with a natural event of a given size.

For each bundle, users would then identify which lifeline routes would be *improved* to withstand the natural event, and assess the extent of the improvement. This assessment may be complemented with inputs from the Oregon Health Authority, the EMS program in the Transportation Safety Division of ODOT, or law enforcement agencies in the region.

Based on all this information, Mosaic users would have to develop a score for each bundle. The following table is provided within the tool to assist users in scoring. Additional guidance can be found in the Mosaic Users' Guide.

Table SA.4.18: Qualitative Scoring of Resiliency of the Network

EXPECTED IMPACT OF BUNDLE	SCORE
Large Adverse	-5
	-4
Moderate Adverse	-3
	-2
Slight Adverse	-1
Neutral	0
Slight Beneficial	1
Moderate Beneficial	2
	3
Large Beneficial	4
	5

Source: Mosaic Tool, Version 2.0, SAFETY & SECURITY worksheet

Land Use & Growth Management

Documentation sheets for the following specific indicators can be found in this section:

- LU.1 – Population and Employment Change and Distribution
- LU.2 – Relative Land Value Change Compared to Base Case

Category: Land Use & Growth Management (LAND USE Worksheet)

General Indicator: Population and Employment Density

Specific Indicator:

LU.1 – Population and Employment Change and Distribution

This specific indicator examines the changes in population and employment (total number and distribution) resulting from a bundle of actions. It provides insight into how future land use patterns may change in response to a bundle.

The indicator must be developed outside the Mosaic tool, with Oregon's State Wide Integrated Model (SWIM), or with a regional transportation model that considers land use in an integrated manner, so that the impacts of transportation improvements (comprised within a bundle) on population and employment can be assessed. Population and employment change and distribution may be summarized in, and presented as a series of maps and tables.

The Mosaic tool offers the possibility of developing a qualitative score for this indicator. If users select this option, it is recommended that they follow the general guidance set forth by the UK Department for Transport[13], where plans or projects are scored for their adherence to and consistency with local, regional or state land use policy goals. Suggestions for determining a qualitative score adapted from UK DfT guidance are provided in the table below. Additional guidance is available in the Mosaic User's Guide (Step 4: Populating the Tool).

Table LU.1.19: Qualitative Scoring of Population and Employment Change and Distribution

EXPECTED IMPACT OF BUNDLE	SCORE
Adverse: The bundle is not consistent with, and would hinder the realization of local, regional or state land use policy goals.	-5
	-4
	-3
	-2
	-1
Neutral	0
Beneficial: The Bundle is consistent with, and would facilitate the realization of local, regional and state land use policy goals.	1
	2
	3
	4
	5

Source: Mosaic Tool, Version 2.0, LAND USE worksheet

[13] UK Department for Transport, The Land Use Policy Sub-Objective, TAG Unit 3.7.2, June 2003

Category: Land Use & Growth Management
(LAND USE Worksheet)

General Indicator: Land Value

Specific Indicator:

LU.2 – Relative Land Value Change Compared to Base Case

This indicator examines the change in land values associated with a bundle of actions, as compared to the Base Case. It looks at the connections between transportation improvements and changes in land values, as compared to the Base Case.

Estimating changes in land value requires use of Oregon's State Wide Integrated Model (SWIM), or of a regional transportation model that considers land use in an integrated manner, so that the impacts of transportation improvements on land values can be assessed.

Changes in land value must be reported on a relative scale. The formula to be used in the development of the indicator is:

Change in Land Value = Land Value under BUNDLE k / Land Value in BASE CASE

This would be estimated outside the Mosaic tool for one or multiple forecast years, and for a pre-determined geography (e.g., Traffic Analysis Zone, or TAZ).

The results may be reported through mapping at the geographic scale used in the analysis (e.g., TAZ). Results can also be reported at different levels, by aggregating land values to the desired geography.

This indicator is only available as "Report Only" in Version 2.0 of the Mosaic tool.

Quality of Life & Livability

Documentation sheets for the following specific indicators can be found in this section:

- QL.1 – Health Benefits of Active Transportation
- QL.2 – Quality of the Travel Environment
- QL.3 – Noise Impacts

Category: Quality of Life & Livability
(QUALITY OF LIFE Worksheet)

General Indicator: Physical Activity

Specific Indicator:

QL.1 – Health Benefits of Active Transportation

This section describes how the specific indicator Health Benefits of Active Transportation (QL.1) is estimated within Version 2.0 of the Mosaic tool.

Four sketch-planning models are provided in the Mosaic workbook to assist users with the estimation and monetization of QL.1:

- The first two models can be used to estimate changes in physical activity resulting from the implementation of a project or bundle of actions.
- The third model estimates the number of lives expected to be saved due to active transportation (walking and cycling). It is based on relationships and parameter values from the 2014 Europe Health Economic Assessment Tool, by the World Health Organization[14].
- The fourth sketch-planning model can be used to estimate changes in the incidence of six diseases (Breast Cancer, Colon Cancer, Cardiovascular Diseases, Dementia, Depression and Diabetes). The calculations in this model are based on the Integrated Transport & Health Impacts Model (ITHIM) and rely on a number of additional data sources and assumptions.[15]

Structure & Logic Diagrams

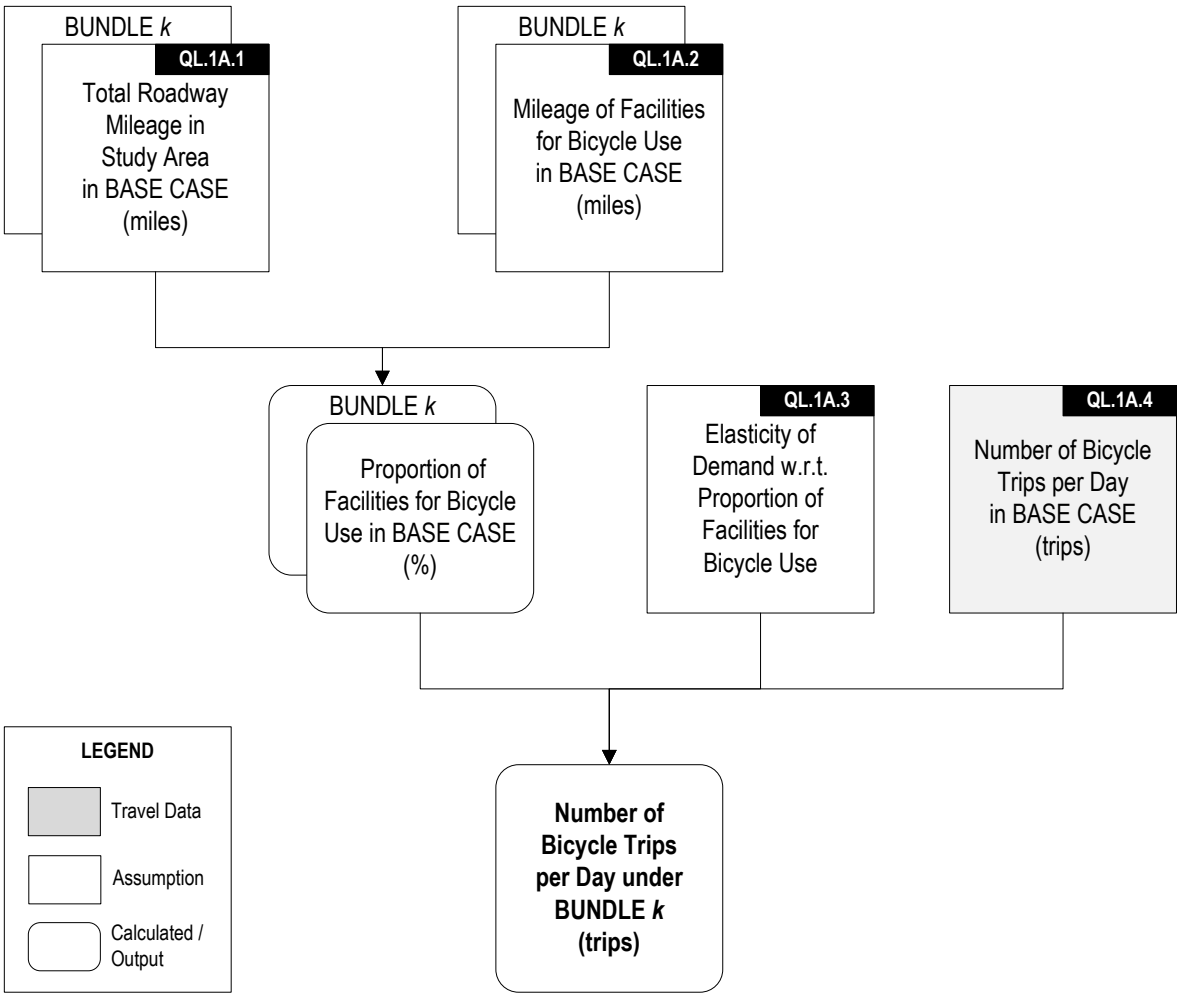
The figures below provide a graphical representation of the equations coded in the workbook to estimate changes in physical activity, using two different methods:

- Figure QL.1.40 illustrates an elasticity-based approach for estimating the change in the number of bicycle trips given a change in the proportion of facilities for bicycle use within a study area. It is based on guidance from the UK Department for Transport.
- Figure QL.1.41 illustrates a biking-likelihood multiplier approach, set forth in a 2006 guidance document prepared for the National Cooperative Highway Research Program (NCHRP Report 552, Guidelines for Analysis of Investments in Bicycle Facilities).

[14] http://www.heatwalkingcycling.org (last accessed November 6, 2014)

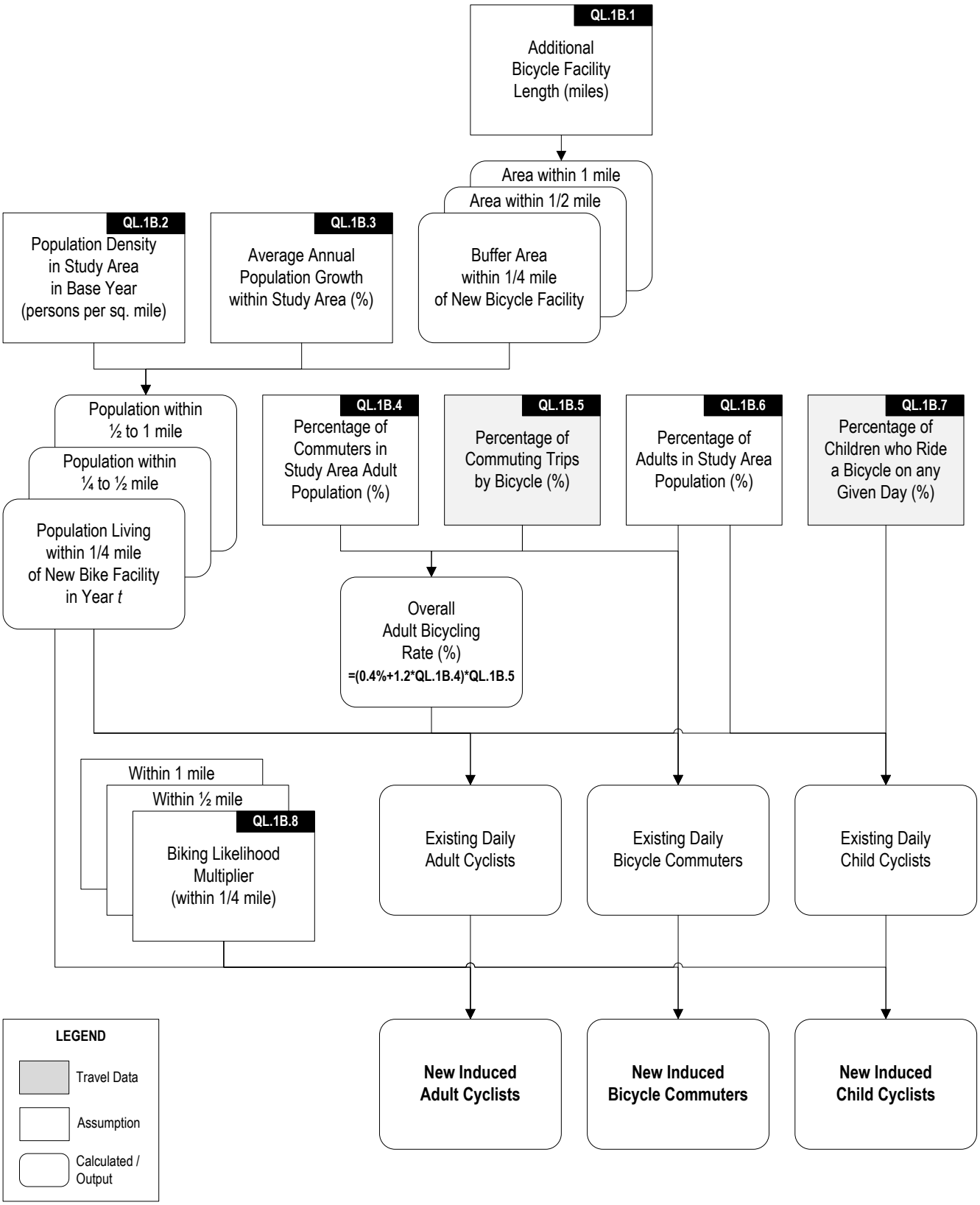
[15] James Woodcock, Integrated Transport & Health Impacts Model, Version Nov. 1, 2011. See Specific Indicator Data Sources and Estimation Methods from the Quality of Life and Livability IDT Team for detail.

Figure QL.1.40: S&L Diagram for Estimating Additional Bike Use, UK DfT Method



Source: CH2M Hill & HDR, based on UK Department for Transport, TAG Unit 3.14.1 page 13

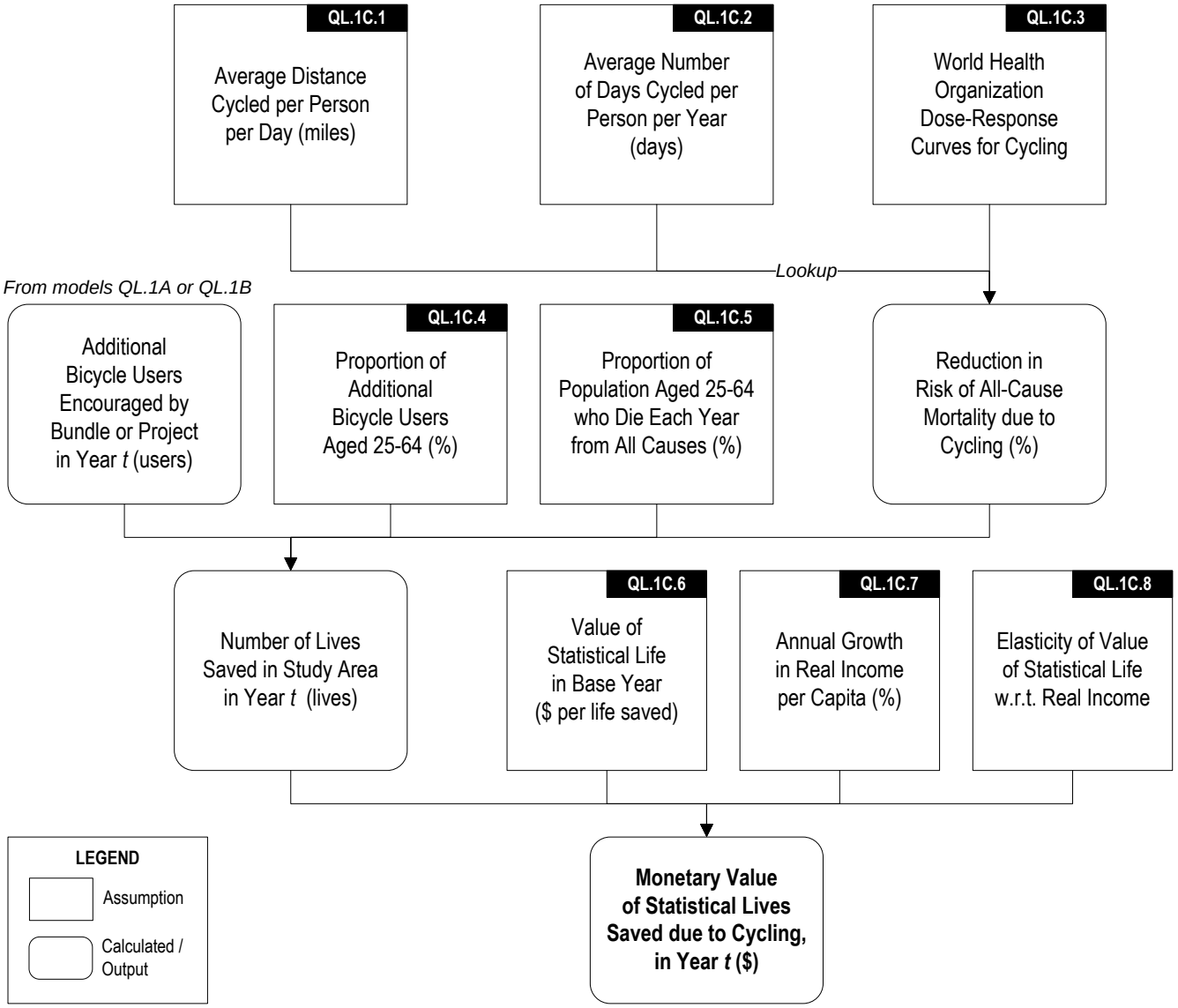
Figure QL.1.41: S&L Diagram for Estimating Additional Bike Use, NCHRP Method



Source: CH2M Hill & HDR, based on Transportation Research Board, National Cooperative Highway Research Program, Guidelines for Analysis of Investments in Bicycle Facilities (2006)

The figure below illustrates how the Mosaic tool estimates changes in the risk of mortality from cycling (using dose-response curves relating mortality risk reductions to the average distance cycled daily) and derives the resulting monetary benefits for inclusion in Benefit-Cost Analysis. Similar variables and calculations are available within the tool for walking.

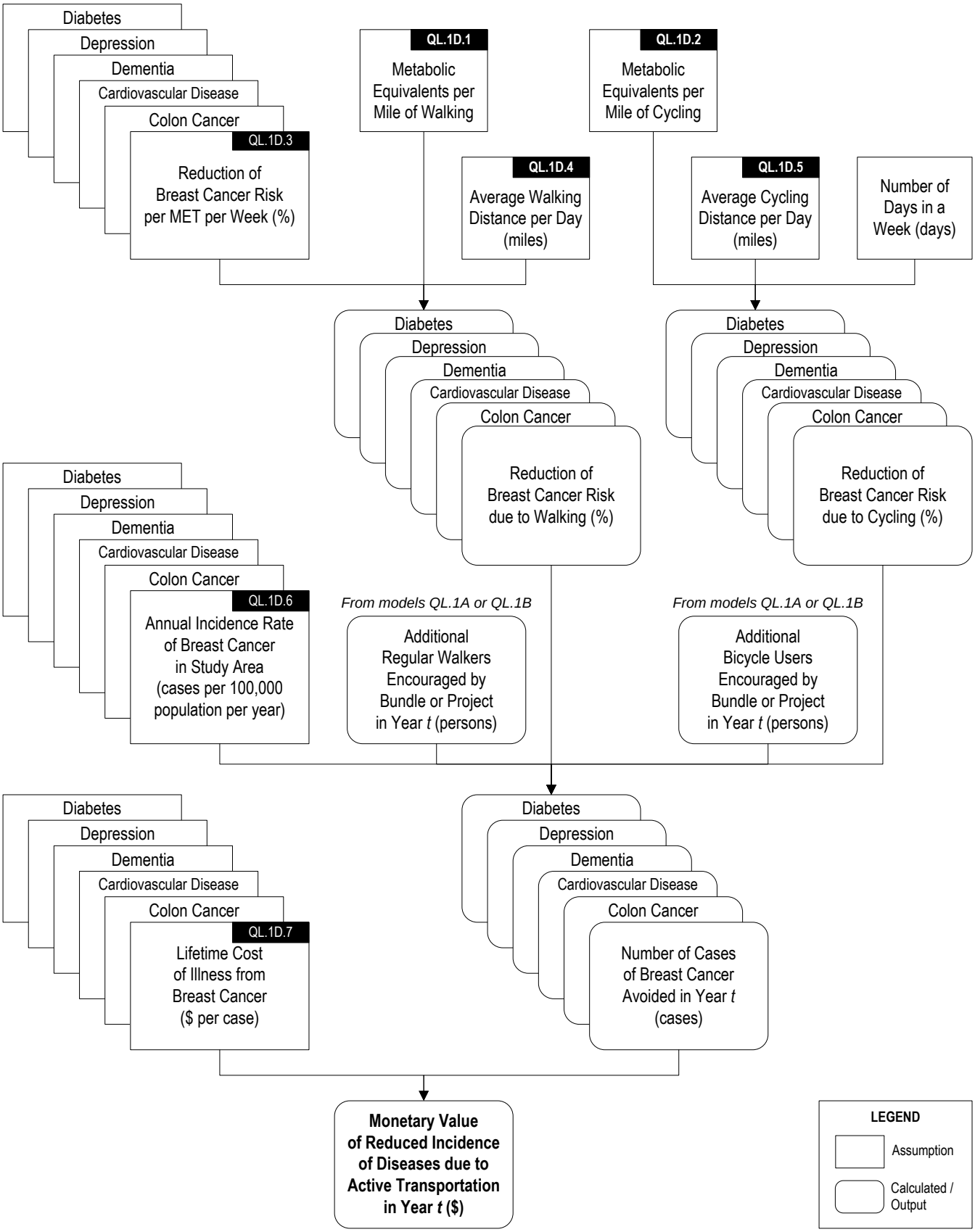
Figure QL.1.42: S&L Diagram for Estimating the Monetary Value of Statistical Lives Saved due to Cycling



Source: CH2M Hill & HDR, based on World Health Organization, Europe Health Economic Assessment Tool (2014)

Figure QL.1.43 on the next page provides a graphical representation of the equations coded in the Mosaic workbook to estimate changes in the incidence of diseases from cycling or walking, along with the resulting monetary benefits. The same calculations are repeated six times in the workbook, with assumptions and parameter values specific to each disease. This is illustrated with stacks of boxes in Figure QL.1.43, where each box represents a disease-specific value.

Figure QL.1.43: S&L Diagram for Estimating Reductions in the Incidence of Diseases due to Active Transportation



Source: CH2M Hill & HDR, based on ITHIM

Data and Assumptions

The tables below provide information on the input variables used in the estimation of additional bicycle use (Table QL.1.20 for the UK DfT Method; Table QL.1.21 for the NCHRP Method).

The input values that must be specified by users of Mosaic are identified with an asterisk * in the Variable ID column. The last column of the tables indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table QL.1.20: Data and Assumptions for Estimating Additional Bike Use, UK DfT Method

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***QL.1A.1**	Total Roadway Mileage in Study Area	Miles	Yes	Yes	Must be specified by user	SKETCH MODELS, Row 14
***QL.1A.2**	Mileage of Facilities for Bicycle Use	Miles	Yes	Yes	Must be specified by user	SKETCH MODELS, Row 15
QL.1A.3	Elasticity of Demand with respect to Proportion of Facilities for Bicycle Use	n/a	No	No	Default parameter value provided in tool	SKETCH MODELS, Row 17
***QL.1A.4**	Number of Bicycle Trips per Day in Base Case	Trips	Yes	No (Base Case Only)	Must be specified by user	SKETCH MODELS, Row 18

Table QL.1.21: Data and Assumptions for Estimating Additional Bike Use, NCHRP Method

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***QL.1B.1**	Additional Bicycle Facility Length	Miles	Yes	Yes	Must be specified by user	SKETCH MODELS, Row 29
***QL.1B.2**	Population Density in Study Area in Base Year	Persons per square mile	Yes	No	Must be specified by user	SKETCH MODELS, Row 27
QL.1B.3	Average Annual Population Growth within Study Area	Percent	Yes	No	Should be specified by user (default value provided in tool)	SKETCH MODELS, Row 28
QL.1B.4	Percentage of Commuters in Study Area Adult Population	Percent	Yes	No	Should be specified by user (default value provided in tool)	SKETCH MODELS, Row 37
***QL.1B.5**	Percentage of Commuting Trips by Bicycle	Percent	Yes	No (Existing Conditions Only)	Must be specified by user	SKETCH MODELS, Row 38
QL.1B.6	Percentage of Adults in Study Area Population	Percent	Yes	No	Should be specified by user (default value provided in tool)	SKETCH MODELS, Row 36
QL.1B.7	Percentage of Children who Ride a Bicycle on any Given Day	Percent	Yes	No (Existing Conditions Only)	Should be specified by user (default value provided in tool)	SKETCH MODELS, Row 39
QL.1B.8	Biking Likelihood Multipliers	n/a	No	No	Default parameter values provided in tool	SKETCH MODELS, Rows 44-46

The table below provides information on the input variables used in the estimation of reduced mortality benefits from cycling. Similar variables are specified in the tool for walking.

Table QL.1.22: Data and Assumptions for Estimating the Monetary Value of Statistical Lives Saved due to Cycling

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***QL.1C.1**	Average Distance Cycled per Person per Day	Miles	Yes	Yes	Must be specified by user	QUALITY OF LIFE, Rows 15-25
QL.1C.2	Average Number of Days Cycled per Person per Year	Days	Yes	No	Should be specified by user (default value provided in tool)	SKETCH MODELS, Cell J 119
QL.1C.3	World Health Organization Dose-Response Curves for Cycling	Percent Risk Reduction	No	No	Curves provided in tool	SKETCH MODELS, Rows 122-137
QL.1C.4	Proportion of Additional Bicycle Users Aged 25-64	Percent of all New Bicycle Users	Yes	No	Should be specified by user (default value provided in tool)	SKETCH MODELS, Rows 148-158
QL.1C.5	Proportion of Population Aged 25-64 who Die Each Year from All Causes	Percent	Yes (State)	No	Default value provided in tool	SKETCH MODELS, Cell J79
QL.1C.6	Value of Statistical Life in Base Year	2012 Dollars per life saved	No	No	Default value provided in tool	MODEL PARAMETERS, Rows 118-120
QL.1C.7	Annual Growth in Real Income per Capita	Percent	No	No	Default value provided in tool	MODEL PARAMETERS, Rows 56-58
QL.1C.8	Elasticity of Value of Statistical Life with respect to Real Income	n/a	No	No	Default parameter value provided in tool	MODEL PARAMETERS, Rows 122-124

The table below provides information on the input variables used in the estimation of reduced morbidity benefits, from walking and/or cycling.

Table QL.1.23: Data and Assumptions for Estimating Reductions in the Incidence of Diseases due to Active Transportation

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
QL.1D.1	Metabolic Equivalents per Mile of Walking	MET (kcal per kg per hour)	No	No	Values provided in tool	SKETCH MODELS, Cell F195
QL.1D.2	Metabolic Equivalents per Mile of Cycling	MET (kcal per kg per hour)	No	No	Values provided in tool	SKETCH MODELS, Cell F197
QL.1D.3	Reduction of Disease Risks per MET per Week	Percent change	No	No	Values provided in tool	SKETCH MODELS, Rows 204-211, Column E
***QL.1D.4**	Average Walking Distance per Day	Miles	Yes	Yes	Must be specified by user	QUALITY OF LIFE, Rows 15-25, Column M
***QL.1D.5**	Average Cycling Distance per Day	Miles	Yes	Yes	Must be specified by user	QUALITY OF LIFE, Rows 15-25, Column H
QL.1D.6	Annual Incidence Rate of Diseases in Study Area	Cases per 100,000 population per year	Yes (State)	No	Some values for Oregon provided in tool	SKETCH MODELS, Rows 184-189, Column D
QL.1D.7	Lifetime or Annual Cost of Illness	Dollars per case	No	No	Values provided in tool	MODEL PARAMETERS, Rows 139-161

Category: Quality of Life & Livability
(QUALITY OF LIFE Worksheet)

General Indicator: Journey Ambience

Specific Indicator:

QL.2 – Quality of the Travel Environment

This section describes how the specific indicator Quality of the Travel Environment (QL.2) is estimated within Version 2.0 of the Mosaic tool. The calculations are based on guidance available from the UK Department for Transport.[16]

Structure & Logic Diagrams

The figures on the next two pages provide a graphical representation of the equations coded in the Mosaic workbook to estimate journey ambience benefits associated with pedestrian and bicycle travel.

For improvements in the Pedestrian Environment, six improvement types are considered:

- Street lighting,
- Curb level,
- Information panels,
- Pavement evenness,
- Directional signage, and
- Benches.

The same calculations are repeated six times, with different parameter values for each improvement type. This is illustrated with stacks of boxes in Figure QL.2.44, where each box represents a different improvement.

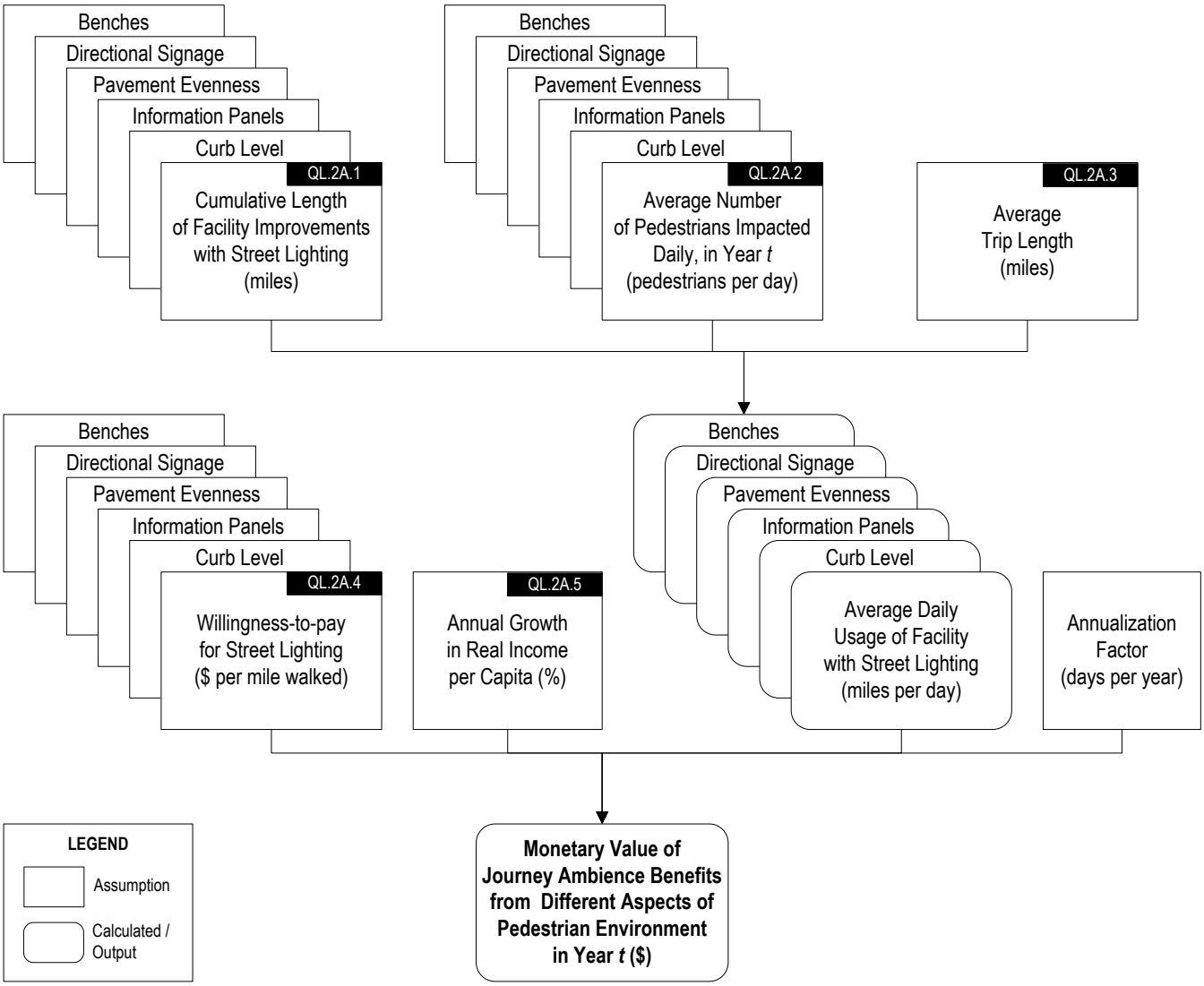
For investments in Bicycle Facilities, five facility types are assessed against a baseline of "No Facility":

- Off-road segregated cycle track,
- On-road segregated cycle lane,
- On-road non-segregated cycle lane,
- Wider lane, and
- Shared bus lane.

The same calculations are repeated five times with different parameter values for each improvement type (see Figure QL.2.45).

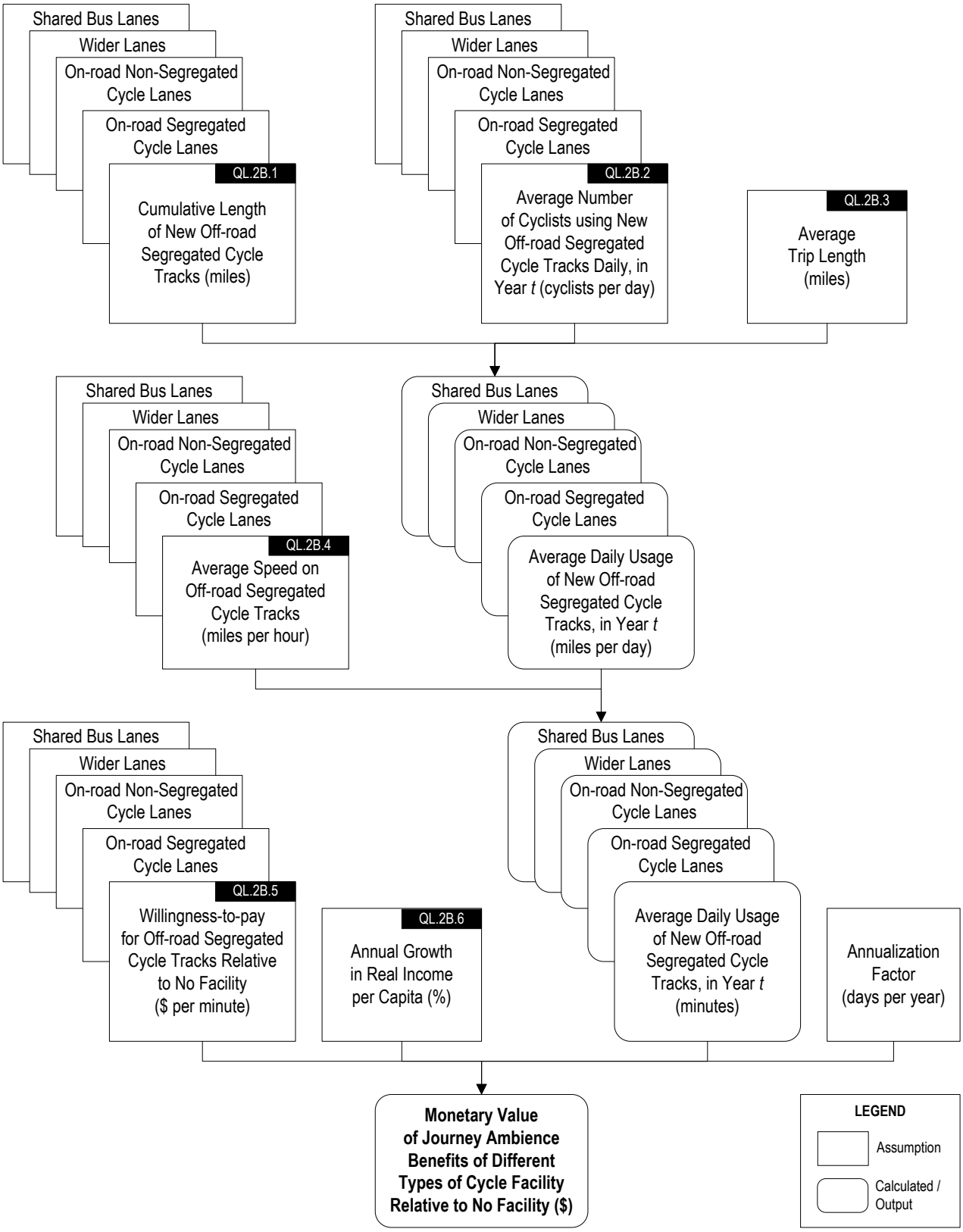
[16] UK Department of Transport, Transport Analysis Guidance (TAG) Unit 3.14, Guidance on the monetizing of Journey Ambiance, January, 2010; see Specific Indicator Data Sources and Estimation Methods from the Quality of Life and Livability IDT Team for detail.

Figure QL.2.44: S&L Diagram for Estimating the Monetary Value of Different Aspects of the Pedestrian Environment



Source: CH2M Hill & HDR, based on UK Department of Transport, Transport Analysis Guidance Unit 3.14

Figure QL.2.45: S&L Diagram for Estimating the Monetary Value of Journey Ambience Benefits from Different Types of Bicycle Facilities Relative to No Facility



Source: CH2M Hill & HDR, based on UK Department of Transport, Transport Analysis Guidance Unit 3.14

Data and Assumptions

The tables below provide information on the input variables used in the estimation of Specific Indicator QL.2. The input values that must be specified by users of Mosaic are identified with an asterisk * in the Variable ID column. The last column of the tables indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table QL.2.24: Data and Assumptions for Estimating the Monetary Value of Different Aspects of the Pedestrian Environment

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***QL.2A.1**	Cumulative Length of Facility Improvements	Miles	Yes	Yes	Must be specified by user	QUALITY OF LIFE, Rows 172-257, Columns D-G
***QL.2A.2**	Average Number of Pedestrians Impacted Daily, in Year t	Number of pedestrians per day	Yes	Yes	Must be specified by user	QUALITY OF LIFE, Rows 172-257, Columns K-N
***QL.2A.3**	Average Trip Length, miles	Percent	Yes	Yes	Must be specified by user	QUALITY OF LIFE, Rows 172-257, Column I
QL.2A.4	Willingness-to-pay for Improvement, in Base Year	Dollars per mile walked	No	No	Values provided in tool	MODEL PARAMETERS, Rows 163-185
QL.2A.5	Annual Growth in Real Income per Capita	Percent	No	No	Default value provided in tool	MODEL PARAMETERS, Rows 56-58

Table QL.2.25: Data and Assumptions for Estimating the Monetary Value of Journey Ambience Benefits from Different Types of Bicycle Facilities Relative to No Facility

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***QL.2B.1**	Cumulative Length of New Off-road Segregated Cycle Tracks	Miles	Yes	Yes	Must be specified by user	QUALITY OF LIFE, Rows 279-349, Columns D-G
***QL.2B.2**	Average Number of Cyclists using New Off-road Segregated Cycle Tracks Daily, in Year t	Number of cyclists per day	Yes	Yes	Must be specified by user	QUALITY OF LIFE, Rows 279-349, Columns K-N
***QL.2B.3**	Average Trip Length, miles	Percent	Yes	Yes	Must be specified by user	QUALITY OF LIFE, Rows 279-349, Column H
QL.2B.4	Average Speed on Facility	Miles per hour	Yes	No	Default values provided in tool	QUALITY OF LIFE, Rows 279-349, Column I
QL.2B.5	Willingness-to-pay for Facility Relative to No Facility, in Base Year	Dollars per minute of use	No	No	Values provided in tool	MODEL PARAMETERS, Rows 187-205
QL.2B.6	Annual Growth in Real Income per Capita	Percent	No	No	Default value provided in tool	MODEL PARAMETERS, Rows 56-58

Category: Quality of Life & Livability
(QUALITY OF LIFE Worksheet)

General Indicator: Noise

Specific Indicator: QL.3 – Noise Impacts

This section describes how the specific indicator Noise Impacts (QL.3) can be estimated within Version 2.0 of the Mosaic tool. Two methods can be used:

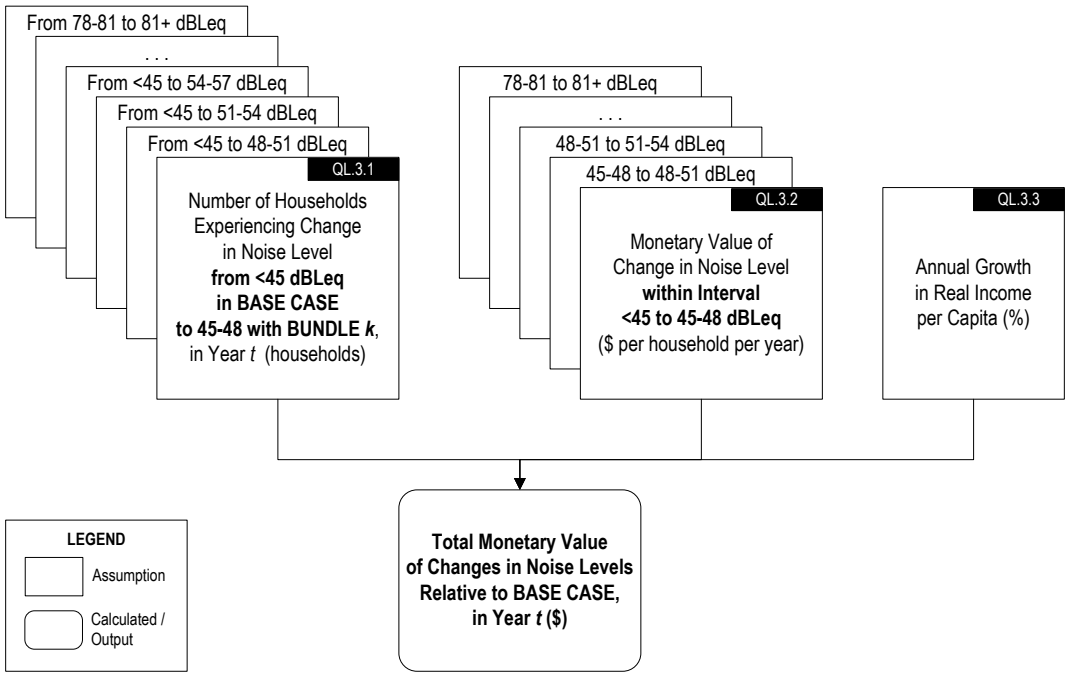
- **Method 1:** Use of sketch-planning tool available within the Mosaic workbook and developed on the basis of guidance from the UK Department for Transport[17].
- **Method 2:** Use of estimates of average noise cost per VMT by vehicle type and roadway type (published by the Federal Highway Administration) in combination with projected changes in VMT under alternative planning options.

This section focuses on Method 1. Users should refer to the Mosaic User's Guide (Step 4: Populating the Mosaic Tool) for additional information on Method 2.

Structure & Logic Diagram

Method 1 is illustrated in Figure QL.3.46 below. The boxes in the upper left corner of the figure are structured as a set of Noise Impact Matrices, as illustrated in Figure QL.3.47 on the next page.

Figure QL.3.46: S&L Diagram for Estimating the Monetary Value of Noise Impacts



Source: CH2M Hill & HDR, based on UK Department of Transport, Transport Analysis Guidance Unit 3.3.2

[17] UK Department of Transport, Transport Analysis Guidance (TAG) Unit 3.3.2, The Noise Sub-Objective, August 2012; and Excel Workbook, U3_3_2S-noise120326.xls.

Users of Mosaic must fill the matrix below to specify the number of households impacted by changes in noise levels from the BASE CASE (noise levels in rows) to a given BUNDLE (noise levels in column). Thus, in the example below, ten thousand households would see their average noise exposure increase from less than 45 dBLeq in the BASE CASE to over 75 dBLeq under BUNDLE k; and no other households would be affected by the bundle.

Figure QL.3.47: Example of a Noise Impact Matrix

2035	BUNDLE *k*	<45	45-47.9	48-50.9	51-53.9	54-56.9	57-59.9	60-62.9	63-65.9	66-68.9	69-71.9	72-74.9	75-77.9	78-80.9	81+
BASE CASE 2035 Low Build															
<45			0	0	0	0	0	0	0	0	0	0	0	0	0
45-47.9		0		0	0	0	0	0	0	0	0	0	0	0	0
48-50.9		0	0		0	0	0	0	0	0	0	0	0	0	0
51-53.9		0	0	0		0	0	0	0	0	0	0	0	0	0
54-56.9		0	0	0	0		0	0	0	0	0	0	0	0	0
57-59.9		0	0	0	0	0		0	0	0	0	0	0	0	0
60-62.9		0	0	0	0	0	0		0	0	0	0	0	0	0
63-65.9		0	0	0	0	0	0	0		0	0	0	0	0	0
66-68.9		0	0	0	0	0	0	0	0		0	0	0	0	0
69-71.9		0	0	0	0	0	0	0	0	0		0	0	0	0
72-74.9		0	0	0	0	0	0	0	0	0	0		0	0	0
75-77.9		0	0	0	0	0	0	0	0	0	0	0		0	0
78-80.9		0	0	0	0	0	0	0	0	0	0	0	0		0
81+		0	0	0	0	0	0	0	0	0	0	0	0	0	

Source: Mosaic Tool, Version 2.0, SKETCH MODELS worksheet

The information collected in the above matrix is combined with assumptions on the monetary value of changes in noise levels (Variable QL.3.2) structured as a look-up table in the Mosaic tool.

Figure QL.3.48: Look-up Table used in the Monetization of Noise Impacts

2035	TO	<45	45-47.9	48-50.9	51-53.9	54-56.9	57-59.9	60-62.9	63-65.9	66-68.9	69-71.9	72-74.9	75-77.9	78-80.9	81+
FROM															
<45		$0	-$39	-$166	-$359	-$618	-$926	-$1,319	-$1,778	-$2,304	-$2,897	-$3,556	-$4,281	-$5,074	-$5,899
45-47.9		$39	$0	-$126	-$319	-$579	-$887	-$1,280	-$1,739	-$2,265	-$2,857	-$3,516	-$4,242	-$5,035	-$5,860
48-50.9		$166	$126	$0	-$193	-$453	-$760	-$1,153	-$1,612	-$2,138	-$2,731	-$3,390	-$4,116	-$4,908	-$5,733
51-53.9		$359	$319	$193	$0	-$260	-$567	-$960	-$1,419	-$1,945	-$2,538	-$3,197	-$3,923	-$4,715	-$5,540
54-56.9		$618	$579	$453	$260	$0	-$308	-$701	-$1,160	-$1,686	-$2,278	-$2,937	-$3,663	-$4,456	-$5,281
57-59.9		$926	$887	$760	$567	$308	$0	-$393	-$852	-$1,378	-$1,971	-$2,630	-$3,356	-$4,148	-$4,973
60-62.9		$1,319	$1,280	$1,153	$960	$701	$393	$0	-$459	-$985	-$1,578	-$2,237	-$2,963	-$3,755	-$4,580
63-65.9		$1,778	$1,739	$1,612	$1,419	$1,160	$852	$459	$0	-$526	-$1,118	-$1,777	-$2,503	-$3,296	-$4,121
66-68.9		$2,304	$2,265	$2,138	$1,945	$1,686	$1,378	$985	$526	$0	-$592	-$1,251	-$1,977	-$2,770	-$3,595
69-71.9		$2,897	$2,857	$2,731	$2,538	$2,278	$1,971	$1,578	$1,118	$592	$0	-$659	-$1,385	-$2,177	-$3,002
72-74.9		$3,556	$3,516	$3,390	$3,197	$2,937	$2,630	$2,237	$1,777	$1,251	$659	$0	-$726	-$1,518	-$2,343
75-77.9		$4,281	$4,242	$4,116	$3,923	$3,663	$3,356	$2,963	$2,503	$1,977	$1,385	$726	$0	-$792	-$1,618
78-80.9		$5,074	$5,035	$4,908	$4,715	$4,456	$4,148	$3,755	$3,296	$2,770	$2,177	$1,518	$792	$0	-$825
81+		$5,899	$5,860	$5,733	$5,540	$5,281	$4,973	$4,580	$4,121	$3,595	$3,002	$2,343	$1,618	$825	$0

Source: Mosaic Tool, Version 2.0, SKETCH MODELS worksheet

Data and Assumptions

The table below provides information on the input variables used in the estimation of Specific Indicator QL.3 using Method 1. The input values that must be specified by users of Mosaic are identified with an asterisk * in the Variable ID column. The last column of the table indicates where in the tool the input values must be entered (Worksheet name in upper case, followed by Row and/or Column number).

Table QL.3.26: Data and Assumptions for Estimating the Monetary Value of Noise Impacts (Method 1)

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
***QL.3.1**	Number of Households Experiencing Change in Noise Level in Year t	Households	Yes	Yes	Must be specified by user	SKETCH MODELS, Rows 443-479 (must be repeated for each bundle, for two years)
QL.3.2	Monetary Value of Change in Noise Level by 3 dBLeq interval	Dollars per household per year	No	No	Default values provided in tool	SKETCH MODELS, Rows 320-332
QL.3.3	Annual Growth in Real Income per Capita	Percent	No	No	Default value provided in tool	MODEL PARAMETERS, Rows 50-52

The model parameters used in the monetization of Noise Impacts (QL.3) with Method 2 are identified in the table below.

Table QL.3.27: Data and Assumptions for Estimating the Monetary Value of Noise Impacts (Method 2)

Variable ID	Variable Name	Units	Area Specific	Bundle Specific	User Input	Where in Tool (Version 2.0)
N/A	Marginal External Costs for Noise, All Highways, Autos	Dollars per VMT	Yes	No	Default values provided in tool	MODEL PARAMETERS, Rows 207-209
N/A	Marginal External Costs for Noise, All Highways, Trucks & Buses	Dollars per VMT	Yes	No	Default values provided in tool	MODEL PARAMETERS, Rows 211-213

Equity

Documentation sheets for the following specific indicators can be found in this section:

- EQ.1 – Distribution of User Benefits across Population Groups
- EQ.2 – Distribution of PM Emissions across Population Groups
- EQ.3 – Distribution of Health Benefits from Active Transportation across Population Groups
- EQ.4 – Distribution of Accident Rates across Population Groups

Category: Equity
(EQUITY Worksheet)

General Indicator: Equity Analysis of Accessibility

Specific Indicator:

EQ.1 – Distribution of User Benefits across Population Groups

This specific indicator examines the distribution of user benefits (travel time savings and out-of-pocket cost savings) by geographic area and population group. User benefits for the entire study area are estimated under the MOBILITY category (indicators MO.1 and MO.5).

User benefits may be estimated and reported for the following areas:

- Urban and rural areas;
- Areas with low/high median household income; and
- Areas with low/high proportions of racial and ethnic minorities.

The disaggregated travel date loaded in Mosaic (see "Load Travel Data and Travel Data Calculations" in the Mosaic Users' Guide) may be used *outside* the tool to estimate user benefits by zone. User benefits should be allocated to the zones where the trips are produced (or originate).

Socio-demographic data at the TAZ level must be used *outside* the Mosaic workbook to characterize all traffic analysis zones considered in the analysis, and to estimate user benefits for zones:

- Within urban or rural areas – based on official county designations;
- With median household incomes greater/lower than the average across all TAZ's plus or minus 1 Standard Deviation; and
- With proportions of racial and ethnic minorities greater/lower than the average across all TAZ's plus or minus 1 Standard Deviation.

Estimates of user benefits should be reported for all six groups defined above, under the Base Case and for all bundles of actions being assessed. The following summary table template is available within the Mosaic tool:

USER BENEFITS (TRAVEL TIME SAVINGS AND USER COST SAVINGS FOR PERSONAL TRAVEL)		PRESENT VALUE OF USER BENEFITS*	IN URBAN/RURAL AREAS		IN AREAS WITH LOW/HIGH INCOME**		IN AREAS WITH LOW/HIGH PROPORTION OF MINORITIES***	
			URBAN	RURAL	LOW INCOME	HIGH INCOME	LOW SHARE	HIGH SHARE
Base Case								
Bundle_1								
Bundle_2								
Bundle_3								
Bundle_4								
Bundle_5								
Bundle_6								
Bundle_7								
Bundle_8								
Bundle_9								
Bundle_10								

** In millions of present-day, discounted dollars*

*** Median household income within TAZ greater/lower than Average +/- 1 StDev*

**** Proportion within TAZ greater/lower than Average +/- 1 StDev*

Source: Mosaic Tool, Version 2.0, EQUITY worksheet

In situations where quantitative information cannot be developed for all groups and areas, Mosaic users may choose to develop a set of qualitative scores directly, using the scale shown in the table below, where impacts are expressed *in relation to other groups*. The scale ranges from "a lot worse" (with an assigned score of -5) to "a lot better" (with a score of +5).

CODE	EXPECTED IMPACT RELATIVE TO OTHER GROUP(S)	SCORE
Wrs_Lg	A Lot Worse	-5
		-4
Wrs_Md	Moderately Worse	-3
		-2
Wrs_Sl	Slightly Worse	-1
Equal	Equal	0
Btt_Sl	Slightly Better	1
Btt_Md	Moderately Better	2
		3
Btt_Lg	A Lot Better	4
		5

Source: Mosaic Tool, Version 2.0, EQUITY worksheet

A summary score for indicator EQ.1 (summarizing the distribution of user benefits across all areas and groups) must then be developed by the user, and entered into the Mosaic workbook.

Category: Equity
(EQUITY Worksheet)

General Indicator: Equity Analysis of Environmental Stewardship

Specific Indicator:

EQ.2 – Distribution of PM 2.5 Emissions across Population Groups

This specific indicator examines the distribution of emissions of fine Particulate Matter (PM 2.5) by geographic area and population group. PM 2.5 emissions for the entire study area are considered under the Environmental Stewardship category, and included in indicator ES.1 (Criteria Air Contaminants).

Users may develop estimates of PM 2.5 emissions for the following areas, based on the geographic distribution of emissions:

- Areas with low/high median household income; and
- Areas with low/high proportions of racial and ethnic minorities.

Traffic volumes by vehicle type and associated amounts of air contaminants (see documentation for indicator ES.1) must be estimated *outside* the Mosaic workbook for all traffic analysis zones considered in the analysis. Socio-demographic information at the TAZ level must then be used *outside* the tool to calculate total emissions in zones:

- With median household incomes greater/lower than the average across all TAZ's plus or minus one Standard Deviation; and
- With proportions of racial and ethnic minorities greater/lower than the average across all TAZ's plus or minus one Standard Deviation.

Emission volumes should be reported for all four groups defined above, under the Base Case and for all bundles of actions being assessed. The following summary table template is available within the Mosaic workbook:

ANNUAL PM 2.5 EMISSION VOLUMES, IN THOUSANDS OF SHORT TONS		OVERALL	IN AREAS WITH LOW/HIGH INCOME**		IN AREAS WITH LOW/HIGH PROPORTION OF MINORITIES***	
			LOW INCOME	HIGH INCOME	LOW SHARE	HIGH SHARE
Base Case						
Bundle_1						
Bundle_2						
Bundle_3						
Bundle_4						
Bundle_5						
Bundle_6						
Bundle_7						
Bundle_8						
Bundle_9						
Bundle_10						

* See Environmental Stewardship Worksheet

** Median household income within TAZ greater/lower than Average +/- 1 StDev

*** Proportion within TAZ greater/lower than Average +/- 1 StDev

Source: Mosaic Tool, Version 2.0, EQUITY worksheet

As with other EQUITY indicators, if quantitative information cannot be developed for all groups and areas, Mosaic users can score the indicator qualitatively, using the “a lot worse” to “a lot better” qualitative scale.

Category: Equity
(EQUITY Worksheet)

General Indicator: Equity Analysis of Quality of Life

Specific Indicator:

EQ.3 – Distribution of Health Benefits across Population Groups

This specific indicator examines how the health benefits of active transportation are distributed across geographic areas and population groups. Health benefits for the entire population of the study area are reported under the Quality of Life category (Specific Indicator QL.1, Health Benefits of Active Transportation).

In this version of the tool, it is assumed that the health benefits of active transportation would be distributed across areas and population groups proportionately with the increase in physical activity (i.e., number of additional person-miles traveled, or PMT), and irrespectively of differences in baseline health conditions, access to preventive care and other socio-demographic factors.

Using data on walking and cycling trips by origin-destination and GIS capabilities *outside* the tool, users of Mosaic may develop estimates of additional pedestrian and bicycle PMT for the following areas:

- Urban and rural areas;
- Areas with low/high median household income; and
- Areas with low/high proportions of racial and ethnic minorities.

PMT projections must be developed for all traffic analysis zones included in the study area. Socio-demographic information at the TAZ level must then be used to calculate additional PMT in zones:

- Within urban or rural areas – based on official county designations;
- With median household incomes greater/lower than the average across all TAZ's plus or minus one Standard Deviation; and
- With proportions of racial and ethnic minorities greater/lower than the average across all TAZ's plus or minus one Standard Deviation.

Additional pedestrian and bicycle PMT should be reported for all six groups defined above, under the Base Case and for all bundles of actions being assessed. A summary table is available within the tool for that purpose.

As with other EQUITY indicators, if quantitative information cannot be developed for all groups and areas, Mosaic users can score the indicator qualitatively, using the "a lot worse" to "a lot better" qualitative scale.

Category: Equity
(EQUITY Worksheet)

General Indicator: Equity Analysis of Safety

Specific Indicator:

EQ.4 – Distribution of Accident Rates across Population Groups

This specific indicator examines the distribution of changes in accident rates (for Fatal, Injury A and Injury B Crashes) by geographic area and population group. System safety and changes in the number of fatal and severe injury crashes for the entire population of the study area are reported under the Safety & Security category (Specific Indicator SA.1).

This indicator can only be assessed qualitatively in this version of Mosaic, using the “a lot worse” to “a lot better” qualitative scale.

The methods and tools required to distribute the quantitative estimates of the Safety analysis by geographic area and population group have not been established yet.

Appendix: Visual Basic Code

The Mosaic tool includes a number of functions and procedures coded in Visual Basic for Applications (VBA) and organized in modules. The full code is reproduced below.

Module Buttons_Chart

```
Option Explicit
Option Base 1

Sub MOSAIC_Lock_Unlock_Button()
MsgBox "This button will allow users to lock or unlock features and cells of MOSAIC."
End Sub

Sub MOSAIC_Email_Button()
MsgBox "To contact the MOSAIC support team, please send an email to unknown@oregonmosaic.com."
End Sub

Sub MOSAIC_Chart_XY_Labels_All(ByRef mysheet As Worksheet, ByVal myChartName As String, ByVal myOffset As Integer)
' This macro assigns names to markers in an XY chart.

'Dimension variables.
   Dim Counter As Integer, ChartName As String, xVals As String

'Disable screen updating while the subroutine is run.
   Application.ScreenUpdating = False

Dim myChart As Chart
'Store the formula for the first series in "xVals".
Set myChart = mysheet.Shapes(myChartName).Chart
   xVals = myChart.SeriesCollection(1).Formula

'Extract the range for the data from xVals.
   xVals = Mid(xVals, InStr(InStr(xVals, ","), xVals, _
      Mid(Left(xVals, InStr(xVals, "!") - 1), 9)))
   xVals = Left(xVals, InStr(InStr(xVals, "!"), xVals, ",") - 1)
   Do While Left(xVals, 1) = ","
      xVals = Mid(xVals, 2)
   Loop

'Attach a label to each data point in the chart.
   For Counter = 1 To Range(xVals).Cells.Count
     myChart.SeriesCollection(1).Points(Counter).HasDataLabel = _
         True
     myChart.SeriesCollection(1).Points(Counter).DataLabel.Text = _
         Range(xVals).Cells(1, Counter).Offset(-myOffset, 0).value
   Next Counter

End Sub
```

```
Sub MOSAIC_Chart_XY_Labels()
' This macro assigns names to markers in an XY chart.

'Dimension variables.
   Dim Counter As Integer, ChartName As String, xVals As String

'Disable screen updating while the subroutine is run.
   Application.ScreenUpdating = False

Dim myChart As Chart
'Store the formula for the first series in "xVals".
Set myChart = Sheet21.Shapes("Chart 3").Chart
   xVals = myChart.SeriesCollection(1).Formula

'Extract the range for the data from xVals.
   xVals = Mid(xVals, InStr(InStr(xVals, ","), xVals, _
      Mid(Left(xVals, InStr(xVals, "!") - 1), 9)))
   xVals = Left(xVals, InStr(InStr(xVals, "!"), xVals, ",") - 1)
   Do While Left(xVals, 1) = ","
      xVals = Mid(xVals, 2)
   Loop

'Attach a label to each data point in the chart.
   For Counter = 1 To Range(xVals).Cells.Count
     myChart.SeriesCollection(1).Points(Counter).HasDataLabel = _
         True
     myChart.SeriesCollection(1).Points(Counter).DataLabel.Text = _
         Range(xVals).Cells(1, Counter).Offset(-3, 0).value
   Next Counter

End Sub

Sub MOSAIC_Chart_XY_Bubble_Labels()
' This macro assigns names to markers in an XY chart.

'Dimension variables.
   Dim Counter As Integer, ChartName As String, xVals As String

'Disable screen updating while the subroutine is run.
   Application.ScreenUpdating = False

Dim myChart As Chart
'Store the formula for the first series in "xVals".
Set myChart = Sheet21.Shapes("Chart 4").Chart
   xVals = myChart.SeriesCollection(1).Formula

'Extract the range for the data from xVals.
   xVals = Mid(xVals, InStr(InStr(xVals, ","), xVals, _
      Mid(Left(xVals, InStr(xVals, "!") - 1), 9)))
   xVals = Left(xVals, InStr(InStr(xVals, "!"), xVals, ",") - 1)
   Do While Left(xVals, 1) = ","
      xVals = Mid(xVals, 2)
   Loop

'Attach a label to each data point in the chart.
```

```
   For Counter = 1 To Range(xVals).Cells.Count
     myChart.SeriesCollection(1).Points(Counter).HasDataLabel = _
         True
     myChart.SeriesCollection(1).Points(Counter).DataLabel.Text = _
         Range(xVals).Cells(1, Counter).Offset(-3, 0).value
   Next Counter

End Sub
```

Module ClearAll

```
Sub MOSAIC_ClearTravelBrowse()

Sheet31.TextBox1.value = ""
Sheet31.TextBox2.value = ""

Sheet31.TextBox3.value = ""
Sheet31.TextBox4.value = ""

Sheet31.TextBox5.value = ""
Sheet31.TextBox6.value = ""

Sheet31.TextBox7.value = ""
Sheet31.TextBox8.value = ""

Sheet31.TextBox9.value = ""
Sheet31.TextBox10.value = ""

Sheet31.TextBox11.value = ""
Sheet31.TextBox12.value = ""

Sheet31.TextBox13.value = ""
Sheet31.TextBox14.value = ""

Sheet31.TextBox15.value = ""
Sheet31.TextBox16.value = ""

Sheet31.TextBox17.value = ""
Sheet31.TextBox18.value = ""

Sheet31.TextBox19.value = ""
Sheet31.TextBox20.value = ""

Sheet31.TextBox21.value = ""
Sheet31.TextBox22.value = ""

End Sub
```

Module LoadAggData

```
Type Agg_Travel_Data
    'Daily Trips
    Trips_Auto(1 To 4) As Double
    Trips_DriveAlone(1 To 4) As Double    'Auto = Drive + Drive Passenger + Passenger
```

```
Trips_DrivePass(1 To 4) As Double
Trips_Pass(1 To 4) As Double

Trips_Transit(1 To 4) As Double
Trips_BusWalk(1 To 4) As Double 'Transit = Bus Walk + Park&Ride Bus
Trips_ParkandRideBus(1 To 4) As Double

Trips_Other(1 To 4) As Double
Trips_Bike(1 To 4) As Double   'Other = Bike + Walk
Trips_Walk(1 To 4) As Double

Trips_Truck(1 To 4) As Double

Trips_Total(1 To 4) As Double   'Total Number of trips

'Peak Period
Trips_Auto_Peak(1 To 4) As Double
Trips_DriveAlone_Peak(1 To 4) As Double    'Auto = Drive + Drive Passenger + Passenger
Trips_DrivePass_Peak(1 To 4) As Double
Trips_Pass_Peak(1 To 4) As Double

Trips_Transit_Peak(1 To 4) As Double
Trips_BusWalk_Peak(1 To 4) As Double 'Transit = Bus Walk + Park&Ride Bus
Trips_ParkandRideBus_Peak(1 To 4) As Double

Trips_Other_Peak(1 To 4) As Double
Trips_Bike_Peak(1 To 4) As Double   'Other = Bike + Walk
Trips_Walk_Peak(1 To 4) As Double

Trips_Truck_Peak(1 To 4) As Double

Trips_Total_Peak(1 To 4) As Double   'Total Number of trips

AvgDistanceTraveled_Auto(1 To 4) As Double
AvgDistanceTraveled_DriveAlone(1 To 4) As Double
AvgDistanceTraveled_DrivePass(1 To 4) As Double
AvgDistanceTraveled_Pass(1 To 4) As Double

AvgDistanceTraveled_Transit(1 To 4) As Double
AvgDistanceTraveled_BusWalk(1 To 4) As Double
AvgDistanceTraveled_ParkandRideBus(1 To 4) As Double

AvgDistanceTraveled_Other(1 To 4) As Double
AvgDistanceTraveled_Bike(1 To 4) As Double
AvgDistanceTraveled_Walk(1 To 4) As Double

AvgDistanceTraveled_Truck(1 To 4) As Double

AvgTravelTime_Auto(1 To 4) As Double
AvgTravelTime_DriveAlone(1 To 4) As Double
AvgTravelTime_DrivePass(1 To 4) As Double
AvgTravelTime_Pass(1 To 4) As Double

AvgTravelTime_Transit(1 To 4) As Double
AvgTravelTime_BusWalk(1 To 4) As Double
```

```
    AvgTravelTime_ParkandRideBus(1 To 4) As Double

    AvgTravelTime_Other(1 To 4) As Double
    AvgTravelTime_Bike(1 To 4) As Double
    AvgTravelTime_Walk(1 To 4) As Double

    AvgTravelTime_Truck(1 To 4) As Double

    AvgAccessTime_Auto(1 To 4) As Double
    AvgAccessTime_DriveAlone(1 To 4) As Double
    AvgAccessTime_DrivePass(1 To 4) As Double
    AvgAccessTime_Pass(1 To 4) As Double

    AvgAccessTime_Transit(1 To 4) As Double
    AvgAccessTime_BusWalk(1 To 4) As Double
    AvgAccessTime_ParkandRideBus(1 To 4) As Double

    AvgAccessTime_Other(1 To 4) As Double
    AvgAccessTime_Bike(1 To 4) As Double
    AvgAccessTime_Walk(1 To 4) As Double

    AvgAccessTime_Truck(1 To 4) As Double

    AvgWaitTime_Auto(1 To 4) As Double
    AvgWaitTime_DriveAlone(1 To 4) As Double
    AvgWaitTime_DrivePass(1 To 4) As Double
    AvgWaitTime_Pass(1 To 4) As Double

    AvgWaitTime_Transit(1 To 4) As Double
    AvgWaitTime_BusWalk(1 To 4) As Double
    AvgWaitTime_ParkandRideBus(1 To 4) As Double

    AvgWaitTime_Other(1 To 4) As Double
    AvgWaitTime_Bike(1 To 4) As Double
    AvgWaitTime_Walk(1 To 4) As Double

    AvgWaitTime_Truck(1 To 4) As Double

'    AvgCost_Auto(1 To 4) As Double
'    AvgCost_Transit(1 To 4) As Double
'    AvgCost_Other(1 To 4) As Double
'    AvgCost_Truck(1 To 4) As Double
'
    CongestedTravel_Auto(1 To 4) As Double
    CongestedTravel_DriveAlone(1 To 4) As Double
    CongestedTravel_DrivePass(1 To 4) As Double
    CongestedTravel_Pass(1 To 4) As Double

    CongestedTravel_Transit(1 To 4) As Double
    CongestedTravel_BusWalk(1 To 4) As Double
    CongestedTravel_ParkandRideBus(1 To 4) As Double

    CongestedTravel_Other(1 To 4) As Double
    CongestedTravel_Bike(1 To 4) As Double
    CongestedTravel_Walk(1 To 4) As Double
```

```
CongestedTravel_Truck(1 To 4) As Double

TravelTimeSavings_Auto(1 To 4) As Double
TravelTimeSavings_DriveAlone(1 To 4) As Double
TravelTimeSavings_DrivePass(1 To 4) As Double
TravelTimeSavings_Pass(1 To 4) As Double

TravelTimeSavings_Transit(1 To 4) As Double
TravelTimeSavings_BusWalk(1 To 4) As Double
TravelTimeSavings_ParkandRideBus(1 To 4) As Double

TravelTimeSavings_Other(1 To 4) As Double
TravelTimeSavings_Bike(1 To 4) As Double
TravelTimeSavings_Walk(1 To 4) As Double

TravelTimeSavings_Truck(1 To 4) As Double

AccessTimeSavings_Auto(1 To 4) As Double
AccessTimeSavings_DriveAlone(1 To 4) As Double
AccessTimeSavings_DrivePass(1 To 4) As Double
AccessTimeSavings_Pass(1 To 4) As Double

AccessTimeSavings_Transit(1 To 4) As Double
AccessTimeSavings_BusWalk(1 To 4) As Double
AccessTimeSavings_ParkandRideBus(1 To 4) As Double

AccessTimeSavings_Other(1 To 4) As Double
AccessTimeSavings_Bike(1 To 4) As Double
AccessTimeSavings_Walk(1 To 4) As Double

AccessTimeSavings_Truck(1 To 4) As Double

WaitTimeSavings_Auto(1 To 4) As Double
WaitTimeSavings_DriveAlone(1 To 4) As Double
WaitTimeSavings_DrivePass(1 To 4) As Double
WaitTimeSavings_Pass(1 To 4) As Double

WaitTimeSavings_Transit(1 To 4) As Double
WaitTimeSavings_BusWalk(1 To 4) As Double
WaitTimeSavings_ParkandRideBus(1 To 4) As Double

WaitTimeSavings_Other(1 To 4) As Double
WaitTimeSavings_Bike(1 To 4) As Double
WaitTimeSavings_Walk(1 To 4) As Double

WaitTimeSavings_Truck(1 To 4) As Double

BufferTimeRecurring_Auto(1 To 4) As Double
BufferTimeRecurring_DriveAlone(1 To 4) As Double
BufferTimeRecurring_DrivePass(1 To 4) As Double
BufferTimeRecurring_Pass(1 To 4) As Double

BufferTimeRecurring_Transit(1 To 4) As Double
BufferTimeRecurring_BusWalk(1 To 4) As Double
```

```
    BufferTimeRecurring_ParkandRideBus(1 To 4) As Double

    BufferTimeRecurring_Other(1 To 4) As Double
    BufferTimeRecurring_Bike(1 To 4) As Double
    BufferTimeRecurring_Walk(1 To 4) As Double

    BufferTimeRecurring_Truck(1 To 4) As Double

End Type

Type Trips
    Auto(1 To 4) As Double
    DriveAlone(1 To 4) As Double
    DrivePass(1 To 4) As Double
    Pass(1 To 4) As Double

    Transit(1 To 4) As Double
    BusWalk(1 To 4) As Double
    ParkandRideBus(1 To 4) As Double

    Other(1 To 4) As Double
    Bike(1 To 4) As Double
    Walk(1 To 4) As Double

    truck(1 To 4)
End Type

Type Benefits
    Auto(1 To 4) As Double
    DriveAlone(1 To 4) As Double
    DrivePass(1 To 4) As Double
    Pass(1 To 4) As Double

    Transit(1 To 4) As Double
    BusWalk(1 To 4) As Double
    ParkandRideBus(1 To 4) As Double

    Other(1 To 4) As Double
    Bike(1 To 4) As Double
    Walk(1 To 4) As Double

    truck(1 To 4) As Double

    Total(1 To 4) As Double
End Type

Public AggTravelData_Bdl(1 To 11) As Agg_Travel_Data

Sub MOSAIC_Load_Aggregate()

Dim i, Bdli, line As Integer
Dim Col, Row As Integer
Const MaxBundle = 11   'BaseCase and 10 Bundles
```

```
Dim ExistingTrips(1 To MaxBundle) As Trips
Dim NewTrips(1 To MaxBundle) As Trips

Dim TravelTimeSaving_minpertrip(1 To MaxBundle) As Benefits
Dim TravelTimeSaving_ExistingTrips(1 To MaxBundle) As Benefits
Dim TravelTimeSaving_NewTrips(1 To MaxBundle) As Benefits
Dim TravelTimeSaving_Total(1 To MaxBundle) As Benefits

Dim OutofPocketSaving_DollarperTrip(1 To MaxBundle) As Benefits
Dim OutofPocketSaving_ExistingTrips(1 To MaxBundle) As Benefits
Dim OutofPocketSaving_NewTrips(1 To MaxBundle) As Benefits
Dim OutofPocketSaving_Total(1 To MaxBundle) As Benefits

Dim ChangeDistanceTraveled(1 To MaxBundle) As Benefits

Dim FreeFlowTravel As Double    'Free flow speed used to estimate hours of congestion
    FreeFlowTravel = 1 / 35 * 60

Dim Sh As Worksheet

'For Progress Bar
Dim j As Long
Dim Diag As New ProgressDialogue


''Select file with aggregate travel data
    Dim FileName As String
    FileName = Application.GetOpenFilename

    If FileExists(FileName) = False Then
        Sheet31.TextBox1.value = ""
        MsgBox ("File not Selected")
        Exit Sub
    End If

    'Empty array for input for new aggregate travel data
    Erase AggTravelData_Bdl

    'Call Procedure To Load Base Trip and Base Cost matrices

    'Progress Bar
    Diag.Configure "Loading Aggregate Travel Data", "Loading...", 0, 4
    Diag.Show
    j = 1
    Diag.SetValue j
    Diag.SetStatus "Loading Base Case Data ..."
    If Diag.cancelIsPressed Then Exit Sub



''Read selected file
    Dim iFNumber As Integer

    'Headings for the imported trip file
    Dim Import_Bdl As Agg_Travel_Data
```

```
    'Load the file into BaseTrips
    iFNumber = FreeFile
    'Prepare the file for input
    Open FileName For Input As #iFNumber

    'Read text file line by line
    i = 1
    line = 1

    Do
        'All mode have the same Input (trips, avg travel time, avg disctance traveled) that's why the headings don't need to be
        'update to reflect a new mode.

        Input #iFNumber, Import_Bdl.Trips_DriveAlone(1), Import_Bdl.Trips_DriveAlone(2), Import_Bdl.Trips_DriveAlone(3),
Import_Bdl.Trips_DriveAlone(4), _
        Import_Bdl.Trips_DrivePass(1), Import_Bdl.Trips_DrivePass(2), Import_Bdl.Trips_DrivePass(3), Import_Bdl.Trips_DrivePass(4), _
        Import_Bdl.Trips_Pass(1), Import_Bdl.Trips_Pass(2), Import_Bdl.Trips_Pass(3), Import_Bdl.Trips_Pass(4), _
        Import_Bdl.Trips_BusWalk(1), Import_Bdl.Trips_BusWalk(2), Import_Bdl.Trips_BusWalk(3), Import_Bdl.Trips_BusWalk(4), _
        Import_Bdl.Trips_ParkandRideBus(1), Import_Bdl.Trips_ParkandRideBus(2), Import_Bdl.Trips_ParkandRideBus(3),
Import_Bdl.Trips_ParkandRideBus(4), _
        Import_Bdl.Trips_Bike(1), Import_Bdl.Trips_Bike(2), Import_Bdl.Trips_Bike(3), Import_Bdl.Trips_Bike(4), _
        Import_Bdl.Trips_Walk(1), Import_Bdl.Trips_Walk(2), Import_Bdl.Trips_Walk(3), Import_Bdl.Trips_Walk(4), _
        Import_Bdl.Trips_Truck(1), Import_Bdl.Trips_Truck(2), Import_Bdl.Trips_Truck(3), Import_Bdl.Trips_Truck(4)

        'Put the data in AggTravelData_Bdl array

            i = line
            'Auto-DriveAlone Trips
        If line <= 11 Then
             AggTravelData_Bdl(i).Trips_DriveAlone(1) = Import_Bdl.Trips_DriveAlone(1)
             AggTravelData_Bdl(i).Trips_DriveAlone(2) = Import_Bdl.Trips_DriveAlone(2)
             AggTravelData_Bdl(i).Trips_DriveAlone(3) = Import_Bdl.Trips_DriveAlone(3)
             AggTravelData_Bdl(i).Trips_DriveAlone(4) = Import_Bdl.Trips_DriveAlone(4)

             'Auto-DrivePass Trips
             AggTravelData_Bdl(i).Trips_DrivePass(1) = Import_Bdl.Trips_DrivePass(1)
             AggTravelData_Bdl(i).Trips_DrivePass(2) = Import_Bdl.Trips_DrivePass(2)
             AggTravelData_Bdl(i).Trips_DrivePass(3) = Import_Bdl.Trips_DrivePass(3)
             AggTravelData_Bdl(i).Trips_DrivePass(4) = Import_Bdl.Trips_DrivePass(4)

             'Auto-Pass Trips
             AggTravelData_Bdl(i).Trips_Pass(1) = Import_Bdl.Trips_Pass(1)
             AggTravelData_Bdl(i).Trips_Pass(2) = Import_Bdl.Trips_Pass(2)
             AggTravelData_Bdl(i).Trips_Pass(3) = Import_Bdl.Trips_Pass(3)
             AggTravelData_Bdl(i).Trips_Pass(4) = Import_Bdl.Trips_Pass(4)

             'Auto-Total Trips
             AggTravelData_Bdl(i).Trips_Auto(1) = AggTravelData_Bdl(i).Trips_DriveAlone(1) + _
                                                  AggTravelData_Bdl(i).Trips_DrivePass(1) + AggTravelData_Bdl(i).Trips_Pass(1)

             AggTravelData_Bdl(i).Trips_Auto(2) = AggTravelData_Bdl(i).Trips_DriveAlone(2) + _
                                                  AggTravelData_Bdl(i).Trips_DrivePass(2) + AggTravelData_Bdl(i).Trips_Pass(2)

             AggTravelData_Bdl(i).Trips_Auto(3) = AggTravelData_Bdl(i).Trips_DriveAlone(3) + _
```

```
                                        AggTravelData_Bdl(i).Trips_DrivePass(3) + AggTravelData_Bdl(i).Trips_Pass(3)

     AggTravelData_Bdl(i).Trips_Auto(4) = AggTravelData_Bdl(i).Trips_DriveAlone(4) + _
                                        AggTravelData_Bdl(i).Trips_DrivePass(4) + AggTravelData_Bdl(i).Trips_Pass(4)


      'Transit-BusWalk Trips
      AggTravelData_Bdl(i).Trips_BusWalk(1) = Import_Bdl.Trips_BusWalk(1)
      AggTravelData_Bdl(i).Trips_BusWalk(2) = Import_Bdl.Trips_BusWalk(2)
      AggTravelData_Bdl(i).Trips_BusWalk(3) = Import_Bdl.Trips_BusWalk(3)
      AggTravelData_Bdl(i).Trips_BusWalk(4) = Import_Bdl.Trips_BusWalk(4)

      'Transit-ParkandRideBus Trips
      AggTravelData_Bdl(i).Trips_ParkandRideBus(1) = Import_Bdl.Trips_ParkandRideBus(1)
      AggTravelData_Bdl(i).Trips_ParkandRideBus(2) = Import_Bdl.Trips_ParkandRideBus(2)
      AggTravelData_Bdl(i).Trips_ParkandRideBus(3) = Import_Bdl.Trips_ParkandRideBus(3)
      AggTravelData_Bdl(i).Trips_ParkandRideBus(4) = Import_Bdl.Trips_ParkandRideBus(4)

      AggTravelData_Bdl(i).Trips_Transit(1) = AggTravelData_Bdl(i).Trips_BusWalk(1) + AggTravelData_Bdl(i).Trips_ParkandRideBus(1)
      AggTravelData_Bdl(i).Trips_Transit(2) = AggTravelData_Bdl(i).Trips_BusWalk(2) + AggTravelData_Bdl(i).Trips_ParkandRideBus(2)
      AggTravelData_Bdl(i).Trips_Transit(3) = AggTravelData_Bdl(i).Trips_BusWalk(3) + AggTravelData_Bdl(i).Trips_ParkandRideBus(3)
      AggTravelData_Bdl(i).Trips_Transit(4) = AggTravelData_Bdl(i).Trips_BusWalk(4) + AggTravelData_Bdl(i).Trips_ParkandRideBus(4)

     'Other-Bike Trips
     AggTravelData_Bdl(i).Trips_Bike(1) = Import_Bdl.Trips_Bike(1)
     AggTravelData_Bdl(i).Trips_Bike(2) = Import_Bdl.Trips_Bike(2)
     AggTravelData_Bdl(i).Trips_Bike(3) = Import_Bdl.Trips_Bike(3)
     AggTravelData_Bdl(i).Trips_Bike(4) = Import_Bdl.Trips_Bike(4)

      'Other-Walk Trips
      AggTravelData_Bdl(i).Trips_Walk(1) = Import_Bdl.Trips_Walk(1)
      AggTravelData_Bdl(i).Trips_Walk(2) = Import_Bdl.Trips_Walk(2)
      AggTravelData_Bdl(i).Trips_Walk(3) = Import_Bdl.Trips_Walk(3)
      AggTravelData_Bdl(i).Trips_Walk(4) = Import_Bdl.Trips_Walk(4)

      'Other-Total Trips
      AggTravelData_Bdl(i).Trips_Other(1) = AggTravelData_Bdl(i).Trips_Bike(1) + AggTravelData_Bdl(i).Trips_Walk(1)
      AggTravelData_Bdl(i).Trips_Other(2) = AggTravelData_Bdl(i).Trips_Bike(2) + AggTravelData_Bdl(i).Trips_Walk(2)
      AggTravelData_Bdl(i).Trips_Other(3) = AggTravelData_Bdl(i).Trips_Bike(3) + AggTravelData_Bdl(i).Trips_Walk(3)
      AggTravelData_Bdl(i).Trips_Other(4) = AggTravelData_Bdl(i).Trips_Bike(4) + AggTravelData_Bdl(i).Trips_Walk(4)

      'Trucks Trips
      AggTravelData_Bdl(i).Trips_Truck(1) = Import_Bdl.Trips_Truck(1)
      AggTravelData_Bdl(i).Trips_Truck(2) = Import_Bdl.Trips_Truck(2)
      AggTravelData_Bdl(i).Trips_Truck(3) = Import_Bdl.Trips_Truck(3)
      AggTravelData_Bdl(i).Trips_Truck(4) = Import_Bdl.Trips_Truck(4)
End If

'Peak Period Trips
If line > 11 And line <= 22 Then
    i = line - 11

    AggTravelData_Bdl(i).Trips_DriveAlone_Peak(1) = Import_Bdl.Trips_DriveAlone(1)
     AggTravelData_Bdl(i).Trips_DriveAlone_Peak(2) = Import_Bdl.Trips_DriveAlone(2)
     AggTravelData_Bdl(i).Trips_DriveAlone_Peak(3) = Import_Bdl.Trips_DriveAlone(3)
```

```
AggTravelData_Bdl(i).Trips_DriveAlone_Peak(4) = Import_Bdl.Trips_DriveAlone(4)

'Auto-DrivePass Trips
AggTravelData_Bdl(i).Trips_DrivePass_Peak(1) = Import_Bdl.Trips_DrivePass(1)
AggTravelData_Bdl(i).Trips_DrivePass_Peak(2) = Import_Bdl.Trips_DrivePass(2)
AggTravelData_Bdl(i).Trips_DrivePass_Peak(3) = Import_Bdl.Trips_DrivePass(3)
AggTravelData_Bdl(i).Trips_DrivePass_Peak(4) = Import_Bdl.Trips_DrivePass(4)

'Auto-Pass Trips
AggTravelData_Bdl(i).Trips_Pass_Peak(1) = Import_Bdl.Trips_Pass(1)
AggTravelData_Bdl(i).Trips_Pass_Peak(2) = Import_Bdl.Trips_Pass(2)
AggTravelData_Bdl(i).Trips_Pass_Peak(3) = Import_Bdl.Trips_Pass(3)
AggTravelData_Bdl(i).Trips_Pass_Peak(4) = Import_Bdl.Trips_Pass(4)

'Auto-Total Trips
AggTravelData_Bdl(i).Trips_Auto_Peak(1) = AggTravelData_Bdl(i).Trips_DriveAlone(1) + _
                                    AggTravelData_Bdl(i).Trips_DrivePass(1) + AggTravelData_Bdl(i).Trips_Pass(1)

AggTravelData_Bdl(i).Trips_Auto_Peak(2) = AggTravelData_Bdl(i).Trips_DriveAlone(2) + _
                                    AggTravelData_Bdl(i).Trips_DrivePass(2) + AggTravelData_Bdl(i).Trips_Pass(2)

AggTravelData_Bdl(i).Trips_Auto_Peak(3) = AggTravelData_Bdl(i).Trips_DriveAlone(3) + _
                                    AggTravelData_Bdl(i).Trips_DrivePass(3) + AggTravelData_Bdl(i).Trips_Pass(3)

AggTravelData_Bdl(i).Trips_Auto_Peak(4) = AggTravelData_Bdl(i).Trips_DriveAlone(4) + _
                                    AggTravelData_Bdl(i).Trips_DrivePass(4) + AggTravelData_Bdl(i).Trips_Pass(4)


'Transit-BusWalk Trips
AggTravelData_Bdl(i).Trips_BusWalk_Peak(1) = Import_Bdl.Trips_BusWalk(1)
AggTravelData_Bdl(i).Trips_BusWalk_Peak(2) = Import_Bdl.Trips_BusWalk(2)
AggTravelData_Bdl(i).Trips_BusWalk_Peak(3) = Import_Bdl.Trips_BusWalk(3)
AggTravelData_Bdl(i).Trips_BusWalk_Peak(4) = Import_Bdl.Trips_BusWalk(4)

'Transit-ParkandRideBus Trips
AggTravelData_Bdl(i).Trips_ParkandRideBus_Peak(1) = Import_Bdl.Trips_ParkandRideBus(1)
AggTravelData_Bdl(i).Trips_ParkandRideBus_Peak(2) = Import_Bdl.Trips_ParkandRideBus(2)
AggTravelData_Bdl(i).Trips_ParkandRideBus_Peak(3) = Import_Bdl.Trips_ParkandRideBus(3)
AggTravelData_Bdl(i).Trips_ParkandRideBus_Peak(4) = Import_Bdl.Trips_ParkandRideBus(4)

AggTravelData_Bdl(i).Trips_Transit_Peak(1) = AggTravelData_Bdl(i).Trips_BusWalk(1) + AggTravelData_Bdl(i).Trips_ParkandRideBus(1)
AggTravelData_Bdl(i).Trips_Transit_Peak(2) = AggTravelData_Bdl(i).Trips_BusWalk(2) + AggTravelData_Bdl(i).Trips_ParkandRideBus(2)
AggTravelData_Bdl(i).Trips_Transit_Peak(3) = AggTravelData_Bdl(i).Trips_BusWalk(3) + AggTravelData_Bdl(i).Trips_ParkandRideBus(3)
AggTravelData_Bdl(i).Trips_Transit_Peak(4) = AggTravelData_Bdl(i).Trips_BusWalk(4) + AggTravelData_Bdl(i).Trips_ParkandRideBus(4)

'Other-Bike Trips
AggTravelData_Bdl(i).Trips_Bike_Peak(1) = Import_Bdl.Trips_Bike(1)
AggTravelData_Bdl(i).Trips_Bike_Peak(2) = Import_Bdl.Trips_Bike(2)
AggTravelData_Bdl(i).Trips_Bike_Peak(3) = Import_Bdl.Trips_Bike(3)
AggTravelData_Bdl(i).Trips_Bike_Peak(4) = Import_Bdl.Trips_Bike(4)

'Other-Walk Trips
AggTravelData_Bdl(i).Trips_Walk_Peak(1) = Import_Bdl.Trips_Walk(1)
AggTravelData_Bdl(i).Trips_Walk_Peak(2) = Import_Bdl.Trips_Walk(2)
AggTravelData_Bdl(i).Trips_Walk_Peak(3) = Import_Bdl.Trips_Walk(3)
```

```
      AggTravelData_Bdl(i).Trips_Walk_Peak(4) = Import_Bdl.Trips_Walk(4)

      'Other-Total Trips
      AggTravelData_Bdl(i).Trips_Other_Peak(1) = AggTravelData_Bdl(i).Trips_Bike(1) + AggTravelData_Bdl(i).Trips_Walk(1)
      AggTravelData_Bdl(i).Trips_Other_Peak(2) = AggTravelData_Bdl(i).Trips_Bike(2) + AggTravelData_Bdl(i).Trips_Walk(2)
      AggTravelData_Bdl(i).Trips_Other_Peak(3) = AggTravelData_Bdl(i).Trips_Bike(3) + AggTravelData_Bdl(i).Trips_Walk(3)
      AggTravelData_Bdl(i).Trips_Other_Peak(4) = AggTravelData_Bdl(i).Trips_Bike(4) + AggTravelData_Bdl(i).Trips_Walk(4)

      'Trucks Trips
      AggTravelData_Bdl(i).Trips_Truck_Peak(1) = Import_Bdl.Trips_Truck(1)
      AggTravelData_Bdl(i).Trips_Truck_Peak(2) = Import_Bdl.Trips_Truck(2)
      AggTravelData_Bdl(i).Trips_Truck_Peak(3) = Import_Bdl.Trips_Truck(3)
      AggTravelData_Bdl(i).Trips_Truck_Peak(4) = Import_Bdl.Trips_Truck(4)
End If

'AvgDistance Traveled
If line > 22 And line <= 33 Then
    i = line - 22

    'Auto-DriveAlone Distance
    AggTravelData_Bdl(i).AvgDistanceTraveled_DriveAlone(1) = Import_Bdl.Trips_DriveAlone(1)
    AggTravelData_Bdl(i).AvgDistanceTraveled_DriveAlone(2) = Import_Bdl.Trips_DriveAlone(2)
    AggTravelData_Bdl(i).AvgDistanceTraveled_DriveAlone(3) = Import_Bdl.Trips_DriveAlone(3)
    AggTravelData_Bdl(i).AvgDistanceTraveled_DriveAlone(4) = Import_Bdl.Trips_DriveAlone(4)

    'Auto-DrivePass Distance
    AggTravelData_Bdl(i).AvgDistanceTraveled_DrivePass(1) = Import_Bdl.Trips_DrivePass(1)
    AggTravelData_Bdl(i).AvgDistanceTraveled_DrivePass(2) = Import_Bdl.Trips_DrivePass(2)
    AggTravelData_Bdl(i).AvgDistanceTraveled_DrivePass(3) = Import_Bdl.Trips_DrivePass(3)
    AggTravelData_Bdl(i).AvgDistanceTraveled_DrivePass(4) = Import_Bdl.Trips_DrivePass(4)

    'Auto-Pass Distance
    AggTravelData_Bdl(i).AvgDistanceTraveled_Pass(1) = Import_Bdl.Trips_Pass(1)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Pass(2) = Import_Bdl.Trips_Pass(2)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Pass(3) = Import_Bdl.Trips_Pass(3)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Pass(4) = Import_Bdl.Trips_Pass(4)

    'Transit-BusWalk Distance
    AggTravelData_Bdl(i).AvgDistanceTraveled_BusWalk(1) = Import_Bdl.Trips_BusWalk(1)
    AggTravelData_Bdl(i).AvgDistanceTraveled_BusWalk(2) = Import_Bdl.Trips_BusWalk(2)
    AggTravelData_Bdl(i).AvgDistanceTraveled_BusWalk(3) = Import_Bdl.Trips_BusWalk(3)
    AggTravelData_Bdl(i).AvgDistanceTraveled_BusWalk(4) = Import_Bdl.Trips_BusWalk(4)

    'Transit-ParkandRide Bus Distance
    AggTravelData_Bdl(i).AvgDistanceTraveled_ParkandRideBus(1) = Import_Bdl.Trips_ParkandRideBus(1)
    AggTravelData_Bdl(i).AvgDistanceTraveled_ParkandRideBus(2) = Import_Bdl.Trips_ParkandRideBus(2)
    AggTravelData_Bdl(i).AvgDistanceTraveled_ParkandRideBus(3) = Import_Bdl.Trips_ParkandRideBus(3)
    AggTravelData_Bdl(i).AvgDistanceTraveled_ParkandRideBus(4) = Import_Bdl.Trips_ParkandRideBus(4)

    'Other-Bike Distance
    AggTravelData_Bdl(i).AvgDistanceTraveled_Bike(1) = Import_Bdl.Trips_Bike(1)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Bike(2) = Import_Bdl.Trips_Bike(2)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Bike(3) = Import_Bdl.Trips_Bike(3)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Bike(4) = Import_Bdl.Trips_Bike(4)
```

```
    'Other-Walk Distance
    AggTravelData_Bdl(i).AvgDistanceTraveled_Walk(1) = Import_Bdl.Trips_Walk(1)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Walk(2) = Import_Bdl.Trips_Walk(2)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Walk(3) = Import_Bdl.Trips_Walk(3)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Walk(4) = Import_Bdl.Trips_Walk(4)

    'Truck Distance
    AggTravelData_Bdl(i).AvgDistanceTraveled_Truck(1) = Import_Bdl.Trips_Truck(1)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Truck(2) = Import_Bdl.Trips_Truck(2)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Truck(3) = Import_Bdl.Trips_Truck(3)
    AggTravelData_Bdl(i).AvgDistanceTraveled_Truck(4) = Import_Bdl.Trips_Truck(4)
End If

'Avg Travel time
If line > 33 And line <= 44 Then
    i = line - 33

    'Auto-DriveAlone Travel time
    AggTravelData_Bdl(i).AvgTravelTime_DriveAlone(1) = Import_Bdl.Trips_DriveAlone(1)
    AggTravelData_Bdl(i).AvgTravelTime_DriveAlone(2) = Import_Bdl.Trips_DriveAlone(2)
    AggTravelData_Bdl(i).AvgTravelTime_DriveAlone(3) = Import_Bdl.Trips_DriveAlone(3)
    AggTravelData_Bdl(i).AvgTravelTime_DriveAlone(4) = Import_Bdl.Trips_DriveAlone(4)

    'Auto-DrivePass Travel time
    AggTravelData_Bdl(i).AvgTravelTime_DrivePass(1) = Import_Bdl.Trips_DrivePass(1)
    AggTravelData_Bdl(i).AvgTravelTime_DrivePass(2) = Import_Bdl.Trips_DrivePass(2)
    AggTravelData_Bdl(i).AvgTravelTime_DrivePass(3) = Import_Bdl.Trips_DrivePass(3)
    AggTravelData_Bdl(i).AvgTravelTime_DrivePass(4) = Import_Bdl.Trips_DrivePass(4)

    'Auto-Pass Travel time
    AggTravelData_Bdl(i).AvgTravelTime_Pass(1) = Import_Bdl.Trips_Pass(1)
    AggTravelData_Bdl(i).AvgTravelTime_Pass(2) = Import_Bdl.Trips_Pass(2)
    AggTravelData_Bdl(i).AvgTravelTime_Pass(3) = Import_Bdl.Trips_Pass(3)
    AggTravelData_Bdl(i).AvgTravelTime_Pass(4) = Import_Bdl.Trips_Pass(4)

    'Transit-Bus Walk Travel time
    AggTravelData_Bdl(i).AvgTravelTime_BusWalk(1) = Import_Bdl.Trips_BusWalk(1)
    AggTravelData_Bdl(i).AvgTravelTime_BusWalk(2) = Import_Bdl.Trips_BusWalk(2)
    AggTravelData_Bdl(i).AvgTravelTime_BusWalk(3) = Import_Bdl.Trips_BusWalk(3)
    AggTravelData_Bdl(i).AvgTravelTime_BusWalk(4) = Import_Bdl.Trips_BusWalk(4)

    'Transit-Park and Ride Bus Travel time
    AggTravelData_Bdl(i).AvgTravelTime_ParkandRideBus(1) = Import_Bdl.Trips_ParkandRideBus(1)
    AggTravelData_Bdl(i).AvgTravelTime_ParkandRideBus(2) = Import_Bdl.Trips_ParkandRideBus(2)
    AggTravelData_Bdl(i).AvgTravelTime_ParkandRideBus(3) = Import_Bdl.Trips_ParkandRideBus(3)
    AggTravelData_Bdl(i).AvgTravelTime_ParkandRideBus(4) = Import_Bdl.Trips_ParkandRideBus(4)

    'Other-Bike Travel time
    AggTravelData_Bdl(i).AvgTravelTime_Bike(1) = Import_Bdl.Trips_Bike(1)
    AggTravelData_Bdl(i).AvgTravelTime_Bike(2) = Import_Bdl.Trips_Bike(2)
    AggTravelData_Bdl(i).AvgTravelTime_Bike(3) = Import_Bdl.Trips_Bike(3)
    AggTravelData_Bdl(i).AvgTravelTime_Bike(4) = Import_Bdl.Trips_Bike(4)

    'Other-Walk Travel time
    AggTravelData_Bdl(i).AvgTravelTime_Walk(1) = Import_Bdl.Trips_Walk(1)
```

```
    AggTravelData_Bdl(i).AvgTravelTime_Walk(2) = Import_Bdl.Trips_Walk(2)
    AggTravelData_Bdl(i).AvgTravelTime_Walk(3) = Import_Bdl.Trips_Walk(3)
    AggTravelData_Bdl(i).AvgTravelTime_Walk(4) = Import_Bdl.Trips_Walk(4)

    'Truck Travel time
    AggTravelData_Bdl(i).AvgTravelTime_Truck(1) = Import_Bdl.Trips_Truck(1)
    AggTravelData_Bdl(i).AvgTravelTime_Truck(2) = Import_Bdl.Trips_Truck(2)
    AggTravelData_Bdl(i).AvgTravelTime_Truck(3) = Import_Bdl.Trips_Truck(3)
    AggTravelData_Bdl(i).AvgTravelTime_Truck(4) = Import_Bdl.Trips_Truck(4)
End If

'Avg Access time
If line > 44 And line <= 55 Then
    i = line - 44

    'Auto-DriveAlone Travel time
    AggTravelData_Bdl(i).AvgAccessTime_DriveAlone(1) = Import_Bdl.Trips_DriveAlone(1)
    AggTravelData_Bdl(i).AvgAccessTime_DriveAlone(2) = Import_Bdl.Trips_DriveAlone(2)
    AggTravelData_Bdl(i).AvgAccessTime_DriveAlone(3) = Import_Bdl.Trips_DriveAlone(3)
    AggTravelData_Bdl(i).AvgAccessTime_DriveAlone(4) = Import_Bdl.Trips_DriveAlone(4)

    'Auto-DrivePass Travel time
    AggTravelData_Bdl(i).AvgAccessTime_DrivePass(1) = Import_Bdl.Trips_DrivePass(1)
    AggTravelData_Bdl(i).AvgAccessTime_DrivePass(2) = Import_Bdl.Trips_DrivePass(2)
    AggTravelData_Bdl(i).AvgAccessTime_DrivePass(3) = Import_Bdl.Trips_DrivePass(3)
    AggTravelData_Bdl(i).AvgAccessTime_DrivePass(4) = Import_Bdl.Trips_DrivePass(4)

    'Auto-Pass Travel time
    AggTravelData_Bdl(i).AvgAccessTime_Pass(1) = Import_Bdl.Trips_Pass(1)
    AggTravelData_Bdl(i).AvgAccessTime_Pass(2) = Import_Bdl.Trips_Pass(2)
    AggTravelData_Bdl(i).AvgAccessTime_Pass(3) = Import_Bdl.Trips_Pass(3)
    AggTravelData_Bdl(i).AvgAccessTime_Pass(4) = Import_Bdl.Trips_Pass(4)

    'Transit-Bus Walk Travel time
    AggTravelData_Bdl(i).AvgAccessTime_BusWalk(1) = Import_Bdl.Trips_BusWalk(1)
    AggTravelData_Bdl(i).AvgAccessTime_BusWalk(2) = Import_Bdl.Trips_BusWalk(2)
    AggTravelData_Bdl(i).AvgAccessTime_BusWalk(3) = Import_Bdl.Trips_BusWalk(3)
    AggTravelData_Bdl(i).AvgAccessTime_BusWalk(4) = Import_Bdl.Trips_BusWalk(4)

    'Transit-Park and Ride Bus Travel time
    AggTravelData_Bdl(i).AvgAccessTime_ParkandRideBus(1) = Import_Bdl.Trips_ParkandRideBus(1)
    AggTravelData_Bdl(i).AvgAccessTime_ParkandRideBus(2) = Import_Bdl.Trips_ParkandRideBus(2)
    AggTravelData_Bdl(i).AvgAccessTime_ParkandRideBus(3) = Import_Bdl.Trips_ParkandRideBus(3)
    AggTravelData_Bdl(i).AvgAccessTime_ParkandRideBus(4) = Import_Bdl.Trips_ParkandRideBus(4)

    'Other-Bike Travel time
    AggTravelData_Bdl(i).AvgAccessTime_Bike(1) = Import_Bdl.Trips_Bike(1)
    AggTravelData_Bdl(i).AvgAccessTime_Bike(2) = Import_Bdl.Trips_Bike(2)
    AggTravelData_Bdl(i).AvgAccessTime_Bike(3) = Import_Bdl.Trips_Bike(3)
    AggTravelData_Bdl(i).AvgAccessTime_Bike(4) = Import_Bdl.Trips_Bike(4)

    'Other-Walk Travel time
    AggTravelData_Bdl(i).AvgAccessTime_Walk(1) = Import_Bdl.Trips_Walk(1)
    AggTravelData_Bdl(i).AvgAccessTime_Walk(2) = Import_Bdl.Trips_Walk(2)
    AggTravelData_Bdl(i).AvgAccessTime_Walk(3) = Import_Bdl.Trips_Walk(3)
```

```
    AggTravelData_Bdl(i).AvgAccessTime_Walk(4) = Import_Bdl.Trips_Walk(4)

    'Truck Travel time
    AggTravelData_Bdl(i).AvgAccessTime_Truck(1) = Import_Bdl.Trips_Truck(1)
    AggTravelData_Bdl(i).AvgAccessTime_Truck(2) = Import_Bdl.Trips_Truck(2)
    AggTravelData_Bdl(i).AvgAccessTime_Truck(3) = Import_Bdl.Trips_Truck(3)
    AggTravelData_Bdl(i).AvgAccessTime_Truck(4) = Import_Bdl.Trips_Truck(4)
End If

'Avg Wait time
If line > 55 And line <= 66 Then
    i = line - 55

    'Auto-DriveAlone Travel time
    AggTravelData_Bdl(i).AvgWaitTime_DriveAlone(1) = Import_Bdl.Trips_DriveAlone(1)
    AggTravelData_Bdl(i).AvgWaitTime_DriveAlone(2) = Import_Bdl.Trips_DriveAlone(2)
    AggTravelData_Bdl(i).AvgWaitTime_DriveAlone(3) = Import_Bdl.Trips_DriveAlone(3)
    AggTravelData_Bdl(i).AvgWaitTime_DriveAlone(4) = Import_Bdl.Trips_DriveAlone(4)

    'Auto-DrivePass Travel time
    AggTravelData_Bdl(i).AvgWaitTime_DrivePass(1) = Import_Bdl.Trips_DrivePass(1)
    AggTravelData_Bdl(i).AvgWaitTime_DrivePass(2) = Import_Bdl.Trips_DrivePass(2)
    AggTravelData_Bdl(i).AvgWaitTime_DrivePass(3) = Import_Bdl.Trips_DrivePass(3)
    AggTravelData_Bdl(i).AvgWaitTime_DrivePass(4) = Import_Bdl.Trips_DrivePass(4)

    'Auto-Pass Travel time
    AggTravelData_Bdl(i).AvgWaitTime_Pass(1) = Import_Bdl.Trips_Pass(1)
    AggTravelData_Bdl(i).AvgWaitTime_Pass(2) = Import_Bdl.Trips_Pass(2)
    AggTravelData_Bdl(i).AvgWaitTime_Pass(3) = Import_Bdl.Trips_Pass(3)
    AggTravelData_Bdl(i).AvgWaitTime_Pass(4) = Import_Bdl.Trips_Pass(4)

    'Transit-Bus Walk Travel time
    AggTravelData_Bdl(i).AvgWaitTime_BusWalk(1) = Import_Bdl.Trips_BusWalk(1)
    AggTravelData_Bdl(i).AvgWaitTime_BusWalk(2) = Import_Bdl.Trips_BusWalk(2)
    AggTravelData_Bdl(i).AvgWaitTime_BusWalk(3) = Import_Bdl.Trips_BusWalk(3)
    AggTravelData_Bdl(i).AvgWaitTime_BusWalk(4) = Import_Bdl.Trips_BusWalk(4)

    'Transit-Park and Ride Bus Travel time
    AggTravelData_Bdl(i).AvgWaitTime_ParkandRideBus(1) = Import_Bdl.Trips_ParkandRideBus(1)
    AggTravelData_Bdl(i).AvgWaitTime_ParkandRideBus(2) = Import_Bdl.Trips_ParkandRideBus(2)
    AggTravelData_Bdl(i).AvgWaitTime_ParkandRideBus(3) = Import_Bdl.Trips_ParkandRideBus(3)
    AggTravelData_Bdl(i).AvgWaitTime_ParkandRideBus(4) = Import_Bdl.Trips_ParkandRideBus(4)

    'Other-Bike Travel time
    AggTravelData_Bdl(i).AvgWaitTime_Bike(1) = Import_Bdl.Trips_Bike(1)
    AggTravelData_Bdl(i).AvgWaitTime_Bike(2) = Import_Bdl.Trips_Bike(2)
    AggTravelData_Bdl(i).AvgWaitTime_Bike(3) = Import_Bdl.Trips_Bike(3)
    AggTravelData_Bdl(i).AvgWaitTime_Bike(4) = Import_Bdl.Trips_Bike(4)

    'Other-Walk Travel time
    AggTravelData_Bdl(i).AvgWaitTime_Walk(1) = Import_Bdl.Trips_Walk(1)
    AggTravelData_Bdl(i).AvgWaitTime_Walk(2) = Import_Bdl.Trips_Walk(2)
    AggTravelData_Bdl(i).AvgWaitTime_Walk(3) = Import_Bdl.Trips_Walk(3)
    AggTravelData_Bdl(i).AvgWaitTime_Walk(4) = Import_Bdl.Trips_Walk(4)
```

```
    'Truck Travel time
    AggTravelData_Bdl(i).AvgWaitTime_Truck(1) = Import_Bdl.Trips_Truck(1)
    AggTravelData_Bdl(i).AvgWaitTime_Truck(2) = Import_Bdl.Trips_Truck(2)
    AggTravelData_Bdl(i).AvgWaitTime_Truck(3) = Import_Bdl.Trips_Truck(3)
    AggTravelData_Bdl(i).AvgWaitTime_Truck(4) = Import_Bdl.Trips_Truck(4)
End If

'Congested Travel
If line > 66 And line <= 77 Then
    i = line - 66
    'Auto-DriveAlone Travel time
    AggTravelData_Bdl(i).CongestedTravel_DriveAlone(1) = Import_Bdl.Trips_DriveAlone(1)
    AggTravelData_Bdl(i).CongestedTravel_DriveAlone(2) = Import_Bdl.Trips_DriveAlone(2)
    AggTravelData_Bdl(i).CongestedTravel_DriveAlone(3) = Import_Bdl.Trips_DriveAlone(3)
    AggTravelData_Bdl(i).CongestedTravel_DriveAlone(4) = Import_Bdl.Trips_DriveAlone(4)

    'Auto-DrivePass Travel time
    AggTravelData_Bdl(i).CongestedTravel_DrivePass(1) = Import_Bdl.Trips_DrivePass(1)
    AggTravelData_Bdl(i).CongestedTravel_DrivePass(2) = Import_Bdl.Trips_DrivePass(2)
    AggTravelData_Bdl(i).CongestedTravel_DrivePass(3) = Import_Bdl.Trips_DrivePass(3)
    AggTravelData_Bdl(i).CongestedTravel_DrivePass(4) = Import_Bdl.Trips_DrivePass(4)

    'Auto-Pass Travel time
    AggTravelData_Bdl(i).CongestedTravel_Pass(1) = Import_Bdl.Trips_Pass(1)
    AggTravelData_Bdl(i).CongestedTravel_Pass(2) = Import_Bdl.Trips_Pass(2)
    AggTravelData_Bdl(i).CongestedTravel_Pass(3) = Import_Bdl.Trips_Pass(3)
    AggTravelData_Bdl(i).CongestedTravel_Pass(4) = Import_Bdl.Trips_Pass(4)

    'Transit-Bus Walk Travel time
    AggTravelData_Bdl(i).CongestedTravel_BusWalk(1) = Import_Bdl.Trips_BusWalk(1)
    AggTravelData_Bdl(i).CongestedTravel_BusWalk(2) = Import_Bdl.Trips_BusWalk(2)
    AggTravelData_Bdl(i).CongestedTravel_BusWalk(3) = Import_Bdl.Trips_BusWalk(3)
    AggTravelData_Bdl(i).CongestedTravel_BusWalk(4) = Import_Bdl.Trips_BusWalk(4)

    'Transit-Park and Ride Bus Travel time
    AggTravelData_Bdl(i).CongestedTravel_ParkandRideBus(1) = Import_Bdl.Trips_ParkandRideBus(1)
    AggTravelData_Bdl(i).CongestedTravel_ParkandRideBus(2) = Import_Bdl.Trips_ParkandRideBus(2)
    AggTravelData_Bdl(i).CongestedTravel_ParkandRideBus(3) = Import_Bdl.Trips_ParkandRideBus(3)
    AggTravelData_Bdl(i).CongestedTravel_ParkandRideBus(4) = Import_Bdl.Trips_ParkandRideBus(4)

    'Other-Bike Travel time
    AggTravelData_Bdl(i).CongestedTravel_Bike(1) = Import_Bdl.Trips_Bike(1)
    AggTravelData_Bdl(i).CongestedTravel_Bike(2) = Import_Bdl.Trips_Bike(2)
    AggTravelData_Bdl(i).CongestedTravel_Bike(3) = Import_Bdl.Trips_Bike(3)
    AggTravelData_Bdl(i).CongestedTravel_Bike(4) = Import_Bdl.Trips_Bike(4)

    'Other-Walk Travel time
    AggTravelData_Bdl(i).CongestedTravel_Walk(1) = Import_Bdl.Trips_Walk(1)
    AggTravelData_Bdl(i).CongestedTravel_Walk(2) = Import_Bdl.Trips_Walk(2)
    AggTravelData_Bdl(i).CongestedTravel_Walk(3) = Import_Bdl.Trips_Walk(3)
    AggTravelData_Bdl(i).CongestedTravel_Walk(4) = Import_Bdl.Trips_Walk(4)

    'Truck Travel time
    AggTravelData_Bdl(i).CongestedTravel_Truck(1) = Import_Bdl.Trips_Truck(1)
    AggTravelData_Bdl(i).CongestedTravel_Truck(2) = Import_Bdl.Trips_Truck(2)
```

```
    AggTravelData_Bdl(i).CongestedTravel_Truck(3) = Import_Bdl.Trips_Truck(3)
    AggTravelData_Bdl(i).CongestedTravel_Truck(4) = Import_Bdl.Trips_Truck(4)
End If

'Travel Time Savings
If line > 77 And line <= 88 Then
    i = line - 77
    'Auto-DriveAlone Travel time
    AggTravelData_Bdl(i).TravelTimeSavings_DriveAlone(1) = Import_Bdl.Trips_DriveAlone(1)
    AggTravelData_Bdl(i).TravelTimeSavings_DriveAlone(2) = Import_Bdl.Trips_DriveAlone(2)
    AggTravelData_Bdl(i).TravelTimeSavings_DriveAlone(3) = Import_Bdl.Trips_DriveAlone(3)
    AggTravelData_Bdl(i).TravelTimeSavings_DriveAlone(4) = Import_Bdl.Trips_DriveAlone(4)

    'Auto-DrivePass Travel time
    AggTravelData_Bdl(i).TravelTimeSavings_DrivePass(1) = Import_Bdl.Trips_DrivePass(1)
    AggTravelData_Bdl(i).TravelTimeSavings_DrivePass(2) = Import_Bdl.Trips_DrivePass(2)
    AggTravelData_Bdl(i).TravelTimeSavings_DrivePass(3) = Import_Bdl.Trips_DrivePass(3)
    AggTravelData_Bdl(i).TravelTimeSavings_DrivePass(4) = Import_Bdl.Trips_DrivePass(4)

    'Auto-Pass Travel time
    AggTravelData_Bdl(i).TravelTimeSavings_Pass(1) = Import_Bdl.Trips_Pass(1)
    AggTravelData_Bdl(i).TravelTimeSavings_Pass(2) = Import_Bdl.Trips_Pass(2)
    AggTravelData_Bdl(i).TravelTimeSavings_Pass(3) = Import_Bdl.Trips_Pass(3)
    AggTravelData_Bdl(i).TravelTimeSavings_Pass(4) = Import_Bdl.Trips_Pass(4)

    'Transit-Bus Walk Travel time
    AggTravelData_Bdl(i).TravelTimeSavings_BusWalk(1) = Import_Bdl.Trips_BusWalk(1)
    AggTravelData_Bdl(i).TravelTimeSavings_BusWalk(2) = Import_Bdl.Trips_BusWalk(2)
    AggTravelData_Bdl(i).TravelTimeSavings_BusWalk(3) = Import_Bdl.Trips_BusWalk(3)
    AggTravelData_Bdl(i).TravelTimeSavings_BusWalk(4) = Import_Bdl.Trips_BusWalk(4)

    'Transit-Park and Ride Bus Travel time
    AggTravelData_Bdl(i).TravelTimeSavings_ParkandRideBus(1) = Import_Bdl.Trips_ParkandRideBus(1)
    AggTravelData_Bdl(i).TravelTimeSavings_ParkandRideBus(2) = Import_Bdl.Trips_ParkandRideBus(2)
    AggTravelData_Bdl(i).TravelTimeSavings_ParkandRideBus(3) = Import_Bdl.Trips_ParkandRideBus(3)
    AggTravelData_Bdl(i).TravelTimeSavings_ParkandRideBus(4) = Import_Bdl.Trips_ParkandRideBus(4)

    'Other-Bike Travel time
    AggTravelData_Bdl(i).TravelTimeSavings_Bike(1) = Import_Bdl.Trips_Bike(1)
    AggTravelData_Bdl(i).TravelTimeSavings_Bike(2) = Import_Bdl.Trips_Bike(2)
    AggTravelData_Bdl(i).TravelTimeSavings_Bike(3) = Import_Bdl.Trips_Bike(3)
    AggTravelData_Bdl(i).TravelTimeSavings_Bike(4) = Import_Bdl.Trips_Bike(4)

    'Other-Walk Travel time
    AggTravelData_Bdl(i).TravelTimeSavings_Walk(1) = Import_Bdl.Trips_Walk(1)
    AggTravelData_Bdl(i).TravelTimeSavings_Walk(2) = Import_Bdl.Trips_Walk(2)
    AggTravelData_Bdl(i).TravelTimeSavings_Walk(3) = Import_Bdl.Trips_Walk(3)
    AggTravelData_Bdl(i).TravelTimeSavings_Walk(4) = Import_Bdl.Trips_Walk(4)

    'Truck Travel time
    AggTravelData_Bdl(i).TravelTimeSavings_Truck(1) = Import_Bdl.Trips_Truck(1)
    AggTravelData_Bdl(i).TravelTimeSavings_Truck(2) = Import_Bdl.Trips_Truck(2)
    AggTravelData_Bdl(i).TravelTimeSavings_Truck(3) = Import_Bdl.Trips_Truck(3)
    AggTravelData_Bdl(i).TravelTimeSavings_Truck(4) = Import_Bdl.Trips_Truck(4)
```

```
End If

'Access Time Savings
If line > 88 And line <= 99 Then
    i = line - 88
    'Auto-DriveAlone Travel time
    AggTravelData_Bdl(i).AccessTimeSavings_DriveAlone(1) = Import_Bdl.Trips_DriveAlone(1)
    AggTravelData_Bdl(i).AccessTimeSavings_DriveAlone(2) = Import_Bdl.Trips_DriveAlone(2)
    AggTravelData_Bdl(i).AccessTimeSavings_DriveAlone(3) = Import_Bdl.Trips_DriveAlone(3)
    AggTravelData_Bdl(i).AccessTimeSavings_DriveAlone(4) = Import_Bdl.Trips_DriveAlone(4)

    'Auto-DrivePass Travel time
    AggTravelData_Bdl(i).AccessTimeSavings_DrivePass(1) = Import_Bdl.Trips_DrivePass(1)
    AggTravelData_Bdl(i).AccessTimeSavings_DrivePass(2) = Import_Bdl.Trips_DrivePass(2)
    AggTravelData_Bdl(i).AccessTimeSavings_DrivePass(3) = Import_Bdl.Trips_DrivePass(3)
    AggTravelData_Bdl(i).AccessTimeSavings_DrivePass(4) = Import_Bdl.Trips_DrivePass(4)

    'Auto-Pass Travel time
    AggTravelData_Bdl(i).AccessTimeSavings_Pass(1) = Import_Bdl.Trips_Pass(1)
    AggTravelData_Bdl(i).AccessTimeSavings_Pass(2) = Import_Bdl.Trips_Pass(2)
    AggTravelData_Bdl(i).AccessTimeSavings_Pass(3) = Import_Bdl.Trips_Pass(3)
    AggTravelData_Bdl(i).AccessTimeSavings_Pass(4) = Import_Bdl.Trips_Pass(4)

    'Transit-Bus Walk Travel time
    AggTravelData_Bdl(i).AccessTimeSavings_BusWalk(1) = Import_Bdl.Trips_BusWalk(1)
    AggTravelData_Bdl(i).AccessTimeSavings_BusWalk(2) = Import_Bdl.Trips_BusWalk(2)
    AggTravelData_Bdl(i).AccessTimeSavings_BusWalk(3) = Import_Bdl.Trips_BusWalk(3)
    AggTravelData_Bdl(i).AccessTimeSavings_BusWalk(4) = Import_Bdl.Trips_BusWalk(4)

    'Transit-Park and Ride Bus Travel time
    AggTravelData_Bdl(i).AccessTimeSavings_ParkandRideBus(1) = Import_Bdl.Trips_ParkandRideBus(1)
    AggTravelData_Bdl(i).AccessTimeSavings_ParkandRideBus(2) = Import_Bdl.Trips_ParkandRideBus(2)
    AggTravelData_Bdl(i).AccessTimeSavings_ParkandRideBus(3) = Import_Bdl.Trips_ParkandRideBus(3)
    AggTravelData_Bdl(i).AccessTimeSavings_ParkandRideBus(4) = Import_Bdl.Trips_ParkandRideBus(4)

    'Other-Bike Travel time
    AggTravelData_Bdl(i).AccessTimeSavings_Bike(1) = Import_Bdl.Trips_Bike(1)
    AggTravelData_Bdl(i).AccessTimeSavings_Bike(2) = Import_Bdl.Trips_Bike(2)
    AggTravelData_Bdl(i).AccessTimeSavings_Bike(3) = Import_Bdl.Trips_Bike(3)
    AggTravelData_Bdl(i).AccessTimeSavings_Bike(4) = Import_Bdl.Trips_Bike(4)

    'Other-Walk Travel time
    AggTravelData_Bdl(i).AccessTimeSavings_Walk(1) = Import_Bdl.Trips_Walk(1)
    AggTravelData_Bdl(i).AccessTimeSavings_Walk(2) = Import_Bdl.Trips_Walk(2)
    AggTravelData_Bdl(i).AccessTimeSavings_Walk(3) = Import_Bdl.Trips_Walk(3)
    AggTravelData_Bdl(i).AccessTimeSavings_Walk(4) = Import_Bdl.Trips_Walk(4)

    'Truck Travel time
    AggTravelData_Bdl(i).AccessTimeSavings_Truck(1) = Import_Bdl.Trips_Truck(1)
    AggTravelData_Bdl(i).AccessTimeSavings_Truck(2) = Import_Bdl.Trips_Truck(2)
    AggTravelData_Bdl(i).AccessTimeSavings_Truck(3) = Import_Bdl.Trips_Truck(3)
    AggTravelData_Bdl(i).AccessTimeSavings_Truck(4) = Import_Bdl.Trips_Truck(4)

End If
```

```
'Wait Time Savings
If line > 99 And line <= 110 Then
    i = line - 99
    'Auto-DriveAlone Travel time
    AggTravelData_Bdl(i).WaitTimeSavings_DriveAlone(1) = Import_Bdl.Trips_DriveAlone(1)
    AggTravelData_Bdl(i).WaitTimeSavings_DriveAlone(2) = Import_Bdl.Trips_DriveAlone(2)
    AggTravelData_Bdl(i).WaitTimeSavings_DriveAlone(3) = Import_Bdl.Trips_DriveAlone(3)
    AggTravelData_Bdl(i).WaitTimeSavings_DriveAlone(4) = Import_Bdl.Trips_DriveAlone(4)

    'Auto-DrivePass Travel time
    AggTravelData_Bdl(i).WaitTimeSavings_DrivePass(1) = Import_Bdl.Trips_DrivePass(1)
    AggTravelData_Bdl(i).WaitTimeSavings_DrivePass(2) = Import_Bdl.Trips_DrivePass(2)
    AggTravelData_Bdl(i).WaitTimeSavings_DrivePass(3) = Import_Bdl.Trips_DrivePass(3)
    AggTravelData_Bdl(i).WaitTimeSavings_DrivePass(4) = Import_Bdl.Trips_DrivePass(4)

    'Auto-Pass Travel time
    AggTravelData_Bdl(i).WaitTimeSavings_Pass(1) = Import_Bdl.Trips_Pass(1)
    AggTravelData_Bdl(i).WaitTimeSavings_Pass(2) = Import_Bdl.Trips_Pass(2)
    AggTravelData_Bdl(i).WaitTimeSavings_Pass(3) = Import_Bdl.Trips_Pass(3)
    AggTravelData_Bdl(i).WaitTimeSavings_Pass(4) = Import_Bdl.Trips_Pass(4)

    'Transit-Bus Walk Travel time
    AggTravelData_Bdl(i).WaitTimeSavings_BusWalk(1) = Import_Bdl.Trips_BusWalk(1)
    AggTravelData_Bdl(i).WaitTimeSavings_BusWalk(2) = Import_Bdl.Trips_BusWalk(2)
    AggTravelData_Bdl(i).WaitTimeSavings_BusWalk(3) = Import_Bdl.Trips_BusWalk(3)
    AggTravelData_Bdl(i).WaitTimeSavings_BusWalk(4) = Import_Bdl.Trips_BusWalk(4)

    'Transit-Park and Ride Bus Travel time
    AggTravelData_Bdl(i).WaitTimeSavings_ParkandRideBus(1) = Import_Bdl.Trips_ParkandRideBus(1)
    AggTravelData_Bdl(i).WaitTimeSavings_ParkandRideBus(2) = Import_Bdl.Trips_ParkandRideBus(2)
    AggTravelData_Bdl(i).WaitTimeSavings_ParkandRideBus(3) = Import_Bdl.Trips_ParkandRideBus(3)
    AggTravelData_Bdl(i).WaitTimeSavings_ParkandRideBus(4) = Import_Bdl.Trips_ParkandRideBus(4)

    'Other-Bike Travel time
    AggTravelData_Bdl(i).WaitTimeSavings_Bike(1) = Import_Bdl.Trips_Bike(1)
    AggTravelData_Bdl(i).WaitTimeSavings_Bike(2) = Import_Bdl.Trips_Bike(2)
    AggTravelData_Bdl(i).WaitTimeSavings_Bike(3) = Import_Bdl.Trips_Bike(3)
    AggTravelData_Bdl(i).WaitTimeSavings_Bike(4) = Import_Bdl.Trips_Bike(4)

    'Other-Walk Travel time
    AggTravelData_Bdl(i).WaitTimeSavings_Walk(1) = Import_Bdl.Trips_Walk(1)
    AggTravelData_Bdl(i).WaitTimeSavings_Walk(2) = Import_Bdl.Trips_Walk(2)
    AggTravelData_Bdl(i).WaitTimeSavings_Walk(3) = Import_Bdl.Trips_Walk(3)
    AggTravelData_Bdl(i).WaitTimeSavings_Walk(4) = Import_Bdl.Trips_Walk(4)

    'Truck Travel time
    AggTravelData_Bdl(i).WaitTimeSavings_Truck(1) = Import_Bdl.Trips_Truck(1)
    AggTravelData_Bdl(i).WaitTimeSavings_Truck(2) = Import_Bdl.Trips_Truck(2)
    AggTravelData_Bdl(i).WaitTimeSavings_Truck(3) = Import_Bdl.Trips_Truck(3)
    AggTravelData_Bdl(i).WaitTimeSavings_Truck(4) = Import_Bdl.Trips_Truck(4)

End If

'Buffer time recurring
If line > 110 And line <= 121 Then
```

```
        i = line - 110
        'Auto-DriveAlone Travel time
        AggTravelData_Bdl(i).BufferTimeRecurring_DriveAlone(1) = Import_Bdl.Trips_DriveAlone(1)
        AggTravelData_Bdl(i).BufferTimeRecurring_DriveAlone(2) = Import_Bdl.Trips_DriveAlone(2)
        AggTravelData_Bdl(i).BufferTimeRecurring_DriveAlone(3) = Import_Bdl.Trips_DriveAlone(3)
        AggTravelData_Bdl(i).BufferTimeRecurring_DriveAlone(4) = Import_Bdl.Trips_DriveAlone(4)

        'Auto-DrivePass Travel time
        AggTravelData_Bdl(i).BufferTimeRecurring_DrivePass(1) = Import_Bdl.Trips_DrivePass(1)
        AggTravelData_Bdl(i).BufferTimeRecurring_DrivePass(2) = Import_Bdl.Trips_DrivePass(2)
        AggTravelData_Bdl(i).BufferTimeRecurring_DrivePass(3) = Import_Bdl.Trips_DrivePass(3)
        AggTravelData_Bdl(i).BufferTimeRecurring_DrivePass(4) = Import_Bdl.Trips_DrivePass(4)

        'Auto-Pass Travel time
        AggTravelData_Bdl(i).BufferTimeRecurring_Pass(1) = Import_Bdl.Trips_Pass(1)
        AggTravelData_Bdl(i).BufferTimeRecurring_Pass(2) = Import_Bdl.Trips_Pass(2)
        AggTravelData_Bdl(i).BufferTimeRecurring_Pass(3) = Import_Bdl.Trips_Pass(3)
        AggTravelData_Bdl(i).BufferTimeRecurring_Pass(4) = Import_Bdl.Trips_Pass(4)

        'Transit-Bus Walk Travel time
        AggTravelData_Bdl(i).BufferTimeRecurring_BusWalk(1) = Import_Bdl.Trips_BusWalk(1)
        AggTravelData_Bdl(i).BufferTimeRecurring_BusWalk(2) = Import_Bdl.Trips_BusWalk(2)
        AggTravelData_Bdl(i).BufferTimeRecurring_BusWalk(3) = Import_Bdl.Trips_BusWalk(3)
        AggTravelData_Bdl(i).BufferTimeRecurring_BusWalk(4) = Import_Bdl.Trips_BusWalk(4)

        'Transit-Park and Ride Bus Travel time
        AggTravelData_Bdl(i).BufferTimeRecurring_ParkandRideBus(1) = Import_Bdl.Trips_ParkandRideBus(1)
        AggTravelData_Bdl(i).BufferTimeRecurring_ParkandRideBus(2) = Import_Bdl.Trips_ParkandRideBus(2)
        AggTravelData_Bdl(i).BufferTimeRecurring_ParkandRideBus(3) = Import_Bdl.Trips_ParkandRideBus(3)
        AggTravelData_Bdl(i).BufferTimeRecurring_ParkandRideBus(4) = Import_Bdl.Trips_ParkandRideBus(4)

        'Other-Bike Travel time
        AggTravelData_Bdl(i).BufferTimeRecurring_Bike(1) = Import_Bdl.Trips_Bike(1)
        AggTravelData_Bdl(i).BufferTimeRecurring_Bike(2) = Import_Bdl.Trips_Bike(2)
        AggTravelData_Bdl(i).BufferTimeRecurring_Bike(3) = Import_Bdl.Trips_Bike(3)
        AggTravelData_Bdl(i).BufferTimeRecurring_Bike(4) = Import_Bdl.Trips_Bike(4)

        'Other-Walk Travel time
        AggTravelData_Bdl(i).BufferTimeRecurring_Walk(1) = Import_Bdl.Trips_Walk(1)
        AggTravelData_Bdl(i).BufferTimeRecurring_Walk(2) = Import_Bdl.Trips_Walk(2)
        AggTravelData_Bdl(i).BufferTimeRecurring_Walk(3) = Import_Bdl.Trips_Walk(3)
        AggTravelData_Bdl(i).BufferTimeRecurring_Walk(4) = Import_Bdl.Trips_Walk(4)

        'Truck Travel time
        AggTravelData_Bdl(i).BufferTimeRecurring_Truck(1) = Import_Bdl.Trips_Truck(1)
        AggTravelData_Bdl(i).BufferTimeRecurring_Truck(2) = Import_Bdl.Trips_Truck(2)
        AggTravelData_Bdl(i).BufferTimeRecurring_Truck(3) = Import_Bdl.Trips_Truck(3)
        AggTravelData_Bdl(i).BufferTimeRecurring_Truck(4) = Import_Bdl.Trips_Truck(4)

    End If

    'move to the next line
    line = line + 1

'Loop until end of file
```

```
Loop Until EOF(iFNumber)

'Close the file
Close #iFNumber

'Calculate the total number of trips (All modes)


'Paste Arrays in TRAVEL DATA CALC
Call MOSAIC_Clear

'Progress Bar
j = j + 1
Diag.SetValue j
Diag.SetStatus "Pasting Travel Demand Data ..."
If Diag.cancelIsPressed Then Exit Sub



Sheet30.Select
Set Sh = Sheet30

Application.Calculation = xlCalculationManual
'Number of Trips
For i = 1 To 4
    'DriveAlone Trips
    Row = 15
    Col = 4
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_DriveAlone(i)
        Next j

    'Drive Passenger Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_DrivePass(i)
    Next j

    'Pass Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_Pass(i)
    Next j

    'Bus Walk Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_BusWalk(i)
    Next j

    'Park and Ride Bus Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_ParkandRideBus(i)
    Next j
```

```
    'Bike Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_Bike(i)
    Next j

    'Walk Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_Walk(i)
    Next j

    'Truck Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_Truck(i)
    Next j
Next i

'Progress Bar
j = j + 1
Diag.SetValue j
Diag.SetStatus "Pasting Travel Demand Data ..."
If Diag.cancelIsPressed Then Exit Sub


 'Number of Trips Peak Period
For i = 1 To 4
    'DriveAlone Trips
    Row = 30
    Col = 4
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_DriveAlone_Peak(i)
        Next j

    'Drive Passenger Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_DrivePass_Peak(i)
    Next j

    'Pass Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_Pass_Peak(i)
    Next j

    'Bus Walk Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_BusWalk_Peak(i)
    Next j

    'Park and Ride Bus Trips
```

```
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_ParkandRideBus_Peak(i)
    Next j

    'Bike Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_Bike_Peak(i)
    Next j

    'Walk Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_Walk_Peak(i)
    Next j

    'Truck Trips
    Col = Col + 5
    For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).Trips_Truck_Peak(i)
    Next j
Next i

'Average Distance Traveled
For i = 1 To 4
    Row = 45
    'Drive Alone Travel Time
    Col = 4
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgDistanceTraveled_DriveAlone(i)
    Next j

    'Drive Pass Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgDistanceTraveled_DrivePass(i)
    Next j

    'Pass Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgDistanceTraveled_Pass(i)
    Next j

    'Bus Walk Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgDistanceTraveled_BusWalk(i)
    Next j

    'Park and Ride Bus Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgDistanceTraveled_ParkandRideBus(i)
```

```
    Next j

    'Bike Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgDistanceTraveled_Bike(i)
    Next j

    'Walk Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgDistanceTraveled_Walk(i)
    Next j

    'Truck Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgDistanceTraveled_Truck(i)
    Next j
Next i

'Progress Bar
j = j + 1
Diag.SetValue j
Diag.SetStatus "Pasting Travel Demand Data ..."
If Diag.cancelIsPressed Then Exit Sub

'Average Travel Traveled
For i = 1 To 4
    Row = 60
    'Drive Alone Travel Time
    Col = 4
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgTravelTime_DriveAlone(i)
    Next j

    'Drive Pass Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgTravelTime_DrivePass(i)
    Next j

    'Pass Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgTravelTime_Pass(i)
    Next j

    'Bus Walk Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgTravelTime_BusWalk(i)
    Next j

    'Park and Ride Bus Travel Time
```

```
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgTravelTime_ParkandRideBus(i)
    Next j

    'Bike Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgTravelTime_Bike(i)
    Next j

    'Walk Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgTravelTime_Walk(i)
    Next j

    'Truck Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgTravelTime_Truck(i)
    Next j
Next i

'Average AccessTime
For i = 1 To 4
    Row = 75
    'Drive Alone Travel Time
    Col = 4
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgAccessTime_DriveAlone(i)
    Next j

    'Drive Pass Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgAccessTime_DrivePass(i)
    Next j

    'Pass Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgAccessTime_Pass(i)
    Next j

    'Bus Walk Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgAccessTime_BusWalk(i)
    Next j

    'Park and Ride Bus Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgAccessTime_ParkandRideBus(i)
```

```
    Next j

    'Bike Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgAccessTime_Bike(i)
    Next j

    'Walk Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgAccessTime_Walk(i)
    Next j

    'Truck Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgAccessTime_Truck(i)
    Next j
Next i

   'Average WaitTime
For i = 1 To 4
    Row = 90
    'Drive Alone Travel Time
    Col = 4
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgWaitTime_DriveAlone(i)
    Next j

    'Drive Pass Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgWaitTime_DrivePass(i)
    Next j

    'Pass Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgWaitTime_Pass(i)
    Next j

    'Bus Walk Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgWaitTime_BusWalk(i)
    Next j

    'Park and Ride Bus Travel Time
    Col = Col + 5
    For j = 1 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgWaitTime_ParkandRideBus(i)
    Next j

    'Bike Travel Time
```

```
        Col = Col + 5
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgWaitTime_Bike(i)
        Next j

        'Walk Travel Time
        Col = Col + 5
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgWaitTime_Walk(i)
        Next j

        'Truck Travel Time
        Col = Col + 5
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AvgWaitTime_Truck(i)
        Next j
    Next i

        'Congested Travel
        'Progress Bar
        j = j + 1
        Diag.SetValue j
        Diag.SetStatus "Pasting Travel Demand Data ..."
        If Diag.cancelIsPressed Then Exit Sub

        'Congested Travel
        Dim In_Hours_ofCong(1 To 8) As Integer

        Dim FreeFlowSpeed(1 To 8) As Double

        For i = 1 To 8
            In_Hours_ofCong(i) = Sheet31.Cells(107 + i, 7)
            FreeFlowSpeed(i) = Sheet31.Cells(107 + i, 8)
        Next i

    For i = 1 To 4
        Row = 105
        Col = 4
        'DriveAlone
        For j = 1 To MaxBundle
            If In_Hours_ofCong(1) = 0 Then
                Sh.Cells(Row + j - 1, Col + i) = ""
            Else
                Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).CongestedTravel_DriveAlone(i)
            End If
        Next j

        'Drive Pass
        Col = Col + 5
        For j = 1 To MaxBundle
            If In_Hours_ofCong(2) = 0 Then
                Sh.Cells(Row + j - 1, Col + i) = ""
            Else
                Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).CongestedTravel_DrivePass(i)
            End If
```

```
Next j

'Pass
Col = Col + 5
For j = 1 To MaxBundle
    If In_Hours_ofCong(3) = 0 Then
        Sh.Cells(Row + j - 1, Col + i) = ""
    Else
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).CongestedTravel_Pass(i)
    End If
Next j

'Bus Walk
Col = Col + 5
For j = 1 To MaxBundle
    If In_Hours_ofCong(4) = 0 Then
        Sh.Cells(Row + j - 1, Col + i) = ""
    Else
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).CongestedTravel_BusWalk(i)
    End If
Next j

'Park and Ride Bus
Col = Col + 5
For j = 1 To MaxBundle
    If In_Hours_ofCong(5) = 0 Then
        Sh.Cells(Row + j - 1, Col + i) = ""
    Else
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).CongestedTravel_ParkandRideBus(i)
    End If
Next j

'Bike
Col = Col + 5
For j = 1 To MaxBundle
    If In_Hours_ofCong(6) = 0 Then
        Sh.Cells(Row + j - 1, Col + i) = ""
    Else
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).CongestedTravel_Bike(i)
    End If
Next j

'Walk
Col = Col + 5
For j = 1 To MaxBundle
    If In_Hours_ofCong(7) = 0 Then
        Sh.Cells(Row + j - 1, Col + i) = ""
    Else
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).CongestedTravel_Walk(i)
    End If
Next j

'Truck
Col = Col + 5
For j = 1 To MaxBundle
```

```
            If In_Hours_ofCong(8) = 0 Then
                Sh.Cells(Row + j - 1, Col + i) = ""
            Else
                Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).CongestedTravel_Truck(i)
            End If
        Next j
    Next i
    '*****

  'Travel Time Savings
  'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Pasting Travel Demand Data ..."
    If Diag.cancelIsPressed Then Exit Sub

  For i = 2 To 4
    Row = 120
    'Travel Time Saving- DriveAlone
    Col = 4
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).TravelTimeSavings_DriveAlone(i)
    Next j

    'Travel Time Saving- DrivePass
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).TravelTimeSavings_DrivePass(i)
    Next j

    'Travel Time Saving- pass
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).TravelTimeSavings_Pass(i)
    Next j

    'Travel Time Saving- BusWalk
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).TravelTimeSavings_BusWalk(i)
    Next j

    'Travel Time Saving- Park and Ride Bus
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).TravelTimeSavings_ParkandRideBus(i)
    Next j

    'Travel Time Saving- Bike
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).TravelTimeSavings_Bike(i)
    Next j

    'Travel Time Saving- Walk
```

```
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).TravelTimeSavings_Walk(i)
    Next j

    'Travel Time Saving- Truck
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).TravelTimeSavings_Truck(i)
    Next j
Next i


 'Access Time Savings
  'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Pasting Travel Demand Data ..."
    If Diag.cancelIsPressed Then Exit Sub

  For i = 2 To 4
    Row = 135
    'Travel Time Saving- DriveAlone
    Col = 4
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AccessTimeSavings_DriveAlone(i)
    Next j

    'Travel Time Saving- DrivePass
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AccessTimeSavings_DrivePass(i)
    Next j

    'Travel Time Saving- pass
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AccessTimeSavings_Pass(i)
    Next j

    'Travel Time Saving- BusWalk
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AccessTimeSavings_BusWalk(i)
    Next j

    'Travel Time Saving- Park and Ride Bus
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AccessTimeSavings_ParkandRideBus(i)
    Next j

    'Travel Time Saving- Bike
    Col = Col + 5
    For j = 2 To MaxBundle
```

```
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AccessTimeSavings_Bike(i)
    Next j

    'Travel Time Saving- Walk
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AccessTimeSavings_Walk(i)
    Next j

    'Travel Time Saving- Truck
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).AccessTimeSavings_Truck(i)
    Next j
Next i


'Wait Time Savings
  'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Pasting Travel Demand Data ..."
    If Diag.cancelIsPressed Then Exit Sub

  For i = 2 To 4
    Row = 150
    'Travel Time Saving- DriveAlone
    Col = 4
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).WaitTimeSavings_DriveAlone(i)
    Next j

    'Travel Time Saving- DrivePass
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).WaitTimeSavings_DrivePass(i)
    Next j

    'Travel Time Saving- pass
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).WaitTimeSavings_Pass(i)
    Next j

    'Travel Time Saving- BusWalk
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).WaitTimeSavings_BusWalk(i)
    Next j

    'Travel Time Saving- Park and Ride Bus
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).WaitTimeSavings_ParkandRideBus(i)
    Next j
```

```
    'Travel Time Saving- Bike
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).WaitTimeSavings_Bike(i)
    Next j

    'Travel Time Saving- Walk
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).WaitTimeSavings_Walk(i)
    Next j

    'Travel Time Saving- Truck
    Col = Col + 5
    For j = 2 To MaxBundle
        Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).WaitTimeSavings_Truck(i)
    Next j
Next i

  'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Pasting Travel Demand Data ..."
    If Diag.cancelIsPressed Then Exit Sub

    'Buffer Time Recurring
    For i = 1 To 4
        Row = 165
        'Drive Alone Travel Time
        Col = 4
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).BufferTimeRecurring_DriveAlone(i)
        Next j

        'Drive Pass Travel Time
        Col = Col + 5
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).BufferTimeRecurring_DrivePass(i)
        Next j

        'Pass Travel Time
        Col = Col + 5
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).BufferTimeRecurring_Pass(i)
        Next j

        'Bus Walk Travel Time
        Col = Col + 5
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).BufferTimeRecurring_BusWalk(i)
        Next j

        'Park and Ride Bus Travel Time
        Col = Col + 5
```

```
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).BufferTimeRecurring_ParkandRideBus(i)
        Next j

        'Bike Travel Time
        Col = Col + 5
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).BufferTimeRecurring_Bike(i)
        Next j

        'Walk Travel Time
        Col = Col + 5
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).BufferTimeRecurring_Walk(i)
        Next j

        'Truck Travel Time
        Col = Col + 5
        For j = 1 To MaxBundle
            Sh.Cells(Row + j - 1, Col + i) = AggTravelData_Bdl(j).BufferTimeRecurring_Truck(i)
        Next j
    Next i

'Put 1 if the user is using Aggregate data instead of O-D Travel Data
    Sheet30.Range("A4") = 1
    Sheet31.Range("A4") = 1

    Diag.Hide

    Application.Calculation = xlCalculationAutomatic

End Sub

Sub MOSAIC_Load_Estimates()
       MsgBox ("This feature will be implemented in future versions of the MOSAIC tool.")
End Sub
```

Module LoadTDM

```
Option Base 1
Option Explicit

Public LoadedBundles As Integer

Sub MOSAIC_AtLeastOneBundle()
'This sub goes true the text box of each bundle to check if the correct path is selected.
'If LoadedBundles is greater than 1 then at least 1 bundle is selected.
'If LoadedBundles is equal to 0 then no bundles have been selected.

    'Initialize Loadedbundles to 0
    LoadedBundles = 0

    'Check if Bundle 1 is loaded
    If FileExists(Sheet31.TextBox3.value) = True And FileExists(Sheet31.TextBox4.value) = True Then
        LoadedBundles = LoadedBundles + 1
```

```
    End If


    'Check if Bundle 2 is loaded
    If FileExists(Sheet31.TextBox5.value) = True And FileExists(Sheet31.TextBox6.value) = True Then
        LoadedBundles = LoadedBundles + 1
    End If

    'Check if Bundle 3 is loaded
    If FileExists(Sheet31.TextBox7.value) = True And FileExists(Sheet31.TextBox8.value) = True Then
        LoadedBundles = LoadedBundles + 1
    End If

    'Check if Bundle 4 is loaded
    If FileExists(Sheet31.TextBox9.value) = True And FileExists(Sheet31.TextBox10.value) = True Then
        LoadedBundles = LoadedBundles + 1
    End If

    'Check if Bundle 5 is loaded
    If FileExists(Sheet31.TextBox11.value) = True And FileExists(Sheet31.TextBox12.value) = True Then
        LoadedBundles = LoadedBundles + 1
    End If

    'Check if Bundle 6 is loaded
    If FileExists(Sheet31.TextBox13.value) = True And FileExists(Sheet31.TextBox14.value) = True Then
        LoadedBundles = LoadedBundles + 1
    End If

    'Check if Bundle 7 is loaded
    If FileExists(Sheet31.TextBox15.value) = True And FileExists(Sheet31.TextBox16.value) = True Then
        LoadedBundles = LoadedBundles + 1
    End If

    'Check if Bundle 8 is loaded
    If FileExists(Sheet31.TextBox17.value) = True And FileExists(Sheet31.TextBox18.value) = True Then
        LoadedBundles = LoadedBundles + 1
    End If

    'Check if Bundle 9 is loaded
    If FileExists(Sheet31.TextBox19.value) = True And FileExists(Sheet31.TextBox20.value) = True Then
        LoadedBundles = LoadedBundles + 1
    End If

    'Check if Bundle 10 is loaded
    If FileExists(Sheet31.TextBox21.value) = True And FileExists(Sheet31.TextBox22.value) = True Then
        LoadedBundles = LoadedBundles + 1
    End If
End Sub

Public Function fct_ExtractElement(txt, n, Separator) As String
'Return the nth element of a text string, where the elements are seperated by a specified seperator character
    Dim AllElements As Variant
    AllElements = Split(txt, Separator)
    fct_ExtractElement = AllElements(n - 1)
End Function
```

```
Public Function FileExists(fname) As Boolean
    If Len(fname) <> 0 Then
        FileExists = Dir(fname) <> ""
    End If
End Function

Sub MOSAIC_NotActive()

MsgBox ("This functionality is not active at the moment.")

End Sub

Sub MOSAIC_Clear()

Sheet30.Select

Range("E15:AQ175").ClearContents
End Sub
```

Module LoadTDM2

```
Option Base 1

'Public Variable used to store input data into arrays:
Type Trips_Mtx2
    Origin As Integer
    Destination As Integer
    Trips(1 To 8, 1 To 2, 1 To 4) As Double
End Type

Type Time_MTX2
    Origin As Integer
    Destination As Integer
    Distance As Double
    UncongestedTime(1 To 4) As Double

    WaitTime(1 To 8, 1 To 2, 1 To 4) As Double
    AccessTime(1 To 8, 1 To 2, 1 To 4) As Double
    InVehTime(1 To 8, 1 To 2, 1 To 4) As Double

    'For WeightedAvg Travel Time
    WaitTimesTrips(1 To 8, 1 To 2, 1 To 4) As Double
    AccessTimesTrips(1 To 8, 1 To 2, 1 To 4) As Double
    InVehTimesTrips(1 To 8, 1 To 2, 1 To 4) As Double

    'For Weighted Avg Distsance Traveled
    DistanceTimesTrips(1 To 8, 1 To 2, 1 To 4) As Double

    'For Weighted Uncongested Time
    UncongestedTimesTrips(1 To 8, 1 To 2, 1 To 4) As Double
End Type

'Base Case
Dim Base_Trips() As Trips_Mtx2
```

```
Dim Base_Times(1 To 21316) As Time_MTX2
Dim Base_TripsByMode() As Double
Dim Base_TripTimeByMode() As Double
Dim Base_AvgDistanceTraveled() As Double
Dim Base_AvgUncongestedTime() As Double
'Bundle 1
Dim Bdl1_Trips() As Trips_Mtx2
Dim Bdl1_Times(1 To 21316) As Time_MTX2
Dim Bdl1_TripsByMode() As Double
Dim Bdl1_TripTimeByMode() As Double
Dim Bdl1_AvgDistanceTraveled() As Double
Dim Bdl1_AvgUncongestedtime() As Double
'Bundle2
 Dim Bdl2_Trips() As Trips_Mtx2
 'Dim Bdl2_Times() As Time_MTX2
 Dim Bdl2_TripsByMode() As Double
 Dim Bdl2_TripTimeByMode() As Double
 Dim Bdl2_AvgDistanceTraveled() As Double
 Dim Bdl2_AvgUncongestedtime() As Double
 'Bundle3
 Dim Bdl3_Trips() As Trips_Mtx2
 'Dim Bdl3_Times(1 To 21316) As Time_MTX2
 Dim Bdl3_TripsByMode() As Double
 Dim Bdl3_TripTimeByMode() As Double
 Dim Bdl3_AvgDistanceTraveled() As Double
 Dim Bdl3_AvgUncongestedtime() As Double
 'Bundle4
 Dim Bdl4_Trips() As Trips_Mtx2
 'Dim Bdl4_Times() As Time_MTX2
 Dim Bdl4_TripsByMode() As Double
 Dim Bdl4_TripTimeByMode() As Double
 Dim Bdl4_AvgDistanceTraveled() As Double
 Dim Bdl4_AvgUncongestedtime() As Double
 'Bundle5
 Dim Bdl5_Trips() As Trips_Mtx2
 'Dim Bdl5_Times() As Time_MTX2
 Dim Bdl5_TripsByMode() As Double
 Dim Bdl5_TripTimeByMode() As Double
 Dim Bdl5_AvgDistanceTraveled() As Double
 Dim Bdl5_AvgUncongestedtime() As Double
 'Bundle6
 Dim Bdl6_Trips() As Trips_Mtx2
 'Dim Bdl6_Times() As Time_MTX2
 Dim Bdl6_TripsByMode() As Double
 Dim Bdl6_TripTimeByMode() As Double
 Dim Bdl6_AvgDistanceTraveled() As Double
 Dim Bdl6_AvgUncongestedtime() As Double
 'Bundle7
 Dim Bdl7_Trips() As Trips_Mtx2
 'Dim Bdl7_Times() As Time_MTX2
 Dim Bdl7_TripsByMode() As Double
 Dim Bdl7_TripTimeByMode() As Double
 Dim Bdl7_AvgDistanceTraveled() As Double
 Dim Bdl7_AvgUncongestedtime() As Double
 'Bundle8
```

```
 Dim Bdl8_Trips() As Trips_Mtx2
 'Dim Bdl8_Times() As Time_MTX2
 Dim Bdl8_TripsByMode() As Double
 Dim Bdl8_TripTimeByMode() As Double
 Dim Bdl8_AvgDistanceTraveled() As Double
 Dim Bdl8_AvgUncongestedtime() As Double
 'Bundle9
 Dim Bdl9_Trips() As Trips_Mtx2
 'Dim Bdl9_Times() As Time_MTX2
 Dim Bdl9_TripsByMode() As Double
 Dim Bdl9_TripTimeByMode() As Double
 Dim Bdl9_AvgDistanceTraveled() As Double
 Dim Bdl9_AvgUncongestedtime() As Double
 'Bundle10
 Dim Bdl10_Trips() As Trips_Mtx2
 'Dim Bdl10_Times() As Time_MTX2
 Dim Bdl10_TripsByMode() As Double
 Dim Bdl10_TripTimeByMode() As Double
 Dim Bdl10_AvgDistanceTraveled() As Double
 Dim Bdl10_AvgUncongestedtime() As Double

Public Const ODPairs = 21316
Public Const MaxCol = 45

Sub MOSAIC_Select_File_Base_Trip()
'Extract Base Trip matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox1.value = FileName
    If FileExists(FileName) = False Then
        Sheet31.TextBox1.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Base_Cost()
'Extract Base Cost Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox2.value = FileName
    BaseCostExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox2.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_1_Trip()
'Extract Bundle1 Trip Matrix
    Dim FileName As String
    Dim Warning As Variant
    FileName = Application.GetOpenFilename
    Sheet31.TextBox3.value = FileName
    Bdl1TripExists = FileExists(FileName)
```

```
    If FileExists(FileName) = False Then
        Sheet31.TextBox3.value = ""
        MsgBox ("File not Selected")
    End If

End Sub

Sub MOSAIC_Select_File_Bundle_1_Cost()
'Extract Bundle1 Cost Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox4.value = FileName
    Bdl1CostExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox4.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_2_Trip()
'Extract Bundle2 Trip Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox5.value = FileName
    Bdl2TripExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox5.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_2_Cost()
'Extract Bundle2 Cost Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox6.value = FileName
    Bdl2CostExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox6.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_3_Trip()
'Extract Bundle3 Trip Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox7.value = FileName
    Bdl3TripExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox7.value = ""
```

```
        MsgBox ("File not Selected")
    End If

End Sub

Sub MOSAIC_Select_File_Bundle_3_Cost()
'Extract Bundle3 Cost Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox8.value = FileName
    Bdl3CostExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox8.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_4_Trip()
'Extract Bundle4 Trip Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox9.value = FileName
    Bdl4TripExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox9.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_4_Cost()
'Extract Bundle4 Cost Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox10.value = FileName
    Bdl4CostExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox10.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_5_Trip()
'Extract Bundle5 Trip Matrix
    Dim FileName As String

    FileName = Application.GetOpenFilename
    Sheet31.TextBox11.value = FileName
    Bdl5TripExists = FileExists(FileName)
    If FileExists(FileName) = False Then
        Sheet31.TextBox11.value = ""
        MsgBox ("File not Selected")
    End If
```

```
End Sub

Sub MOSAIC_Select_File_Bundle_5_Cost()
'Extract Bundle5 Cost Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox12.value = FileName
    Bdl5CostExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox12.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_6_Trip()
'Extract Bundle6 Trip Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox13.value = FileName
    Bdl6TripExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox13.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_6_Cost()
'Extract Bundle6 Cost Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox14.value = FileName
    Bdl6CostExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox14.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_7_Trip()
'Extract Bundle7 Trip Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox15.value = FileName
    Bdl7TripExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox15.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_7_Cost()
```

```
'Extract Bundle7 Cost Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox16.value = FileName
    Bdl7CostExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox16.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_8_Trip()
'Extract Bundle8 Trip Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox17.value = FileName
    Bdl8TripExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox17.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_8_Cost()
'Extract Bundle8 Cost Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox18.value = FileName
    Bdl8CostExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox18.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_9_Trip()
'Extract Bundle9 Trip Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox19.value = FileName
    Bdl9TripExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox19.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_9_Cost()
'Extract Bundle9 Cost Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
```

```
    Sheet31.TextBox20.value = FileName
    Bdl9CostExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox20.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_10_Trip()
'Extract Bundle10 Trip Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox21.value = FileName
    Bdl10TripExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox21.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_Select_File_Bundle_10_Cost()
'Extract Bundle9 Cost Matrix
    Dim FileName As String
    FileName = Application.GetOpenFilename
    Sheet31.TextBox22.value = FileName
    Bdl10CostExists = FileExists(FileName)

    If FileExists(FileName) = False Then
        Sheet31.TextBox22.value = ""
        MsgBox ("File not Selected")
    End If
End Sub

Sub MOSAIC_LoadData2()

    Dim NoBase As Variant           'Warning Message when the base files are not loaded
    Dim NoBdl As Variant            'Warning Message when no bundles are loaded
    Dim Sh As Worksheet             'Assigned to current worksheet
    Dim g As Variant
    Dim ODPairs As Long
    Dim Fy, y, k As Integer
    Dim Modes As Integer

    Erase Base_Times
    Erase Base_TripsByMode
    Erase Base_TripTimeByMode
    Erase Base_AvgDistanceTraveled
    Erase Base_AvgUncongestedTime

    Dim Bdl1_TimeBenefits() As Double
    Dim Bdl2_TimeBenefits() As Double
    Dim Bdl3_TimeBenefits() As Double
    Dim Bdl4_TimeBenefits() As Double
```

```
    Dim Bdl5_TimeBenefits() As Double
    Dim Bdl6_TimeBenefits() As Double
    Dim Bdl7_TimeBenefits() As Double
    Dim Bdl8_TimeBenefits() As Double
    Dim Bdl9_TimeBenefits() As Double
    Dim Bdl10_TimeBenefits() As Double

    Dim BufferTimeIndex(1 To 11, 1 To 4) As Double
    Dim VehOccupancy As Double

    Dim Row, Col As Integer

    Dim CurrentStatus As String
    CurrentStatus = Sheet2.Range("F21")

    'Unprotect Model
    MOSAIC_UnprotectModel



    'For Progress Bar
    Dim j As Long
    Dim Diag As New ProgressDialogue

    Application.Calculation = xlCalculationManual

    For i = 1 To 11
        For k = 1 To 4
            BufferTimeIndex(i, k) = Sheet9.Cells(111 + i, 3 + k)
        Next k
    Next i

    Set Sh = Sheet30
    Col = 4
    'Erase Previous Content
    Sh.Range("E15:AQ175").ClearContents
    '*******************************************
    'Put 1 if the user is using Aggregate data instead of O-D Travel Data
    Sheet30.Range("A4") = 0
    Sheet31.Range("A4") = 0

    On Error Resume Next

    'User click on Load Data button
    '****1- Need to make sure that the base case matrix is in
    'Check if Path Exist
    If FileExists(Sheet31.TextBox1.value) = False Or FileExists(Sheet31.TextBox2.value) = False Then
        'Base Matrix files not selected
        NoBase = MsgBox("Base Case Files Not Selected", vbCritical, "Base Case Not Selected")
        Exit Sub
    End If
'    '****2- Load Selected Bundles
    'Check if at least one bundle is delected
    MOSAIC_AtLeastOneBundle
    If LoadedBundles = 0 Then
```

```
    'Need to load at least 1 bundles
    NoBdl = MsgBox("Need To Add At Least One Bundle", vbExclamation, "No Bundle Selected")
    Exit Sub
End If

Fy = Sheet27.Range("A18")
Modes = Sheet9.Range("C12")

'Reading BaseCase Trips

'Progress Bar
Diag.Configure "Loading Data", "Loading...", 0, LoadedBundles + 3
Diag.Show

j = 1
Diag.SetValue j
Diag.SetStatus "Loading Base Case Data ..."
If Diag.cancelIsPressed Then Exit Sub

Base_Trips() = ImportTripMtx(Sheet31.TextBox1.value)
ODPairs = UBound(Base_Trips)
'Reading BaseCase Times

'************
ImportBaseTime
Diag.Show
Erase Base_TripsByMode
Erase Base_TripTimeByMode
Erase Base_AvgDistanceTraveled
Erase Base_AvgUncongestedTime

'Base_Times() = ImportTimeMtx(Sheet31.TextBox2.value, Base_Trips, Fy + 1)
'Total Trips By Mode
Base_TripsByMode() = TotalTripsByMode(Base_Trips, UBound(Base_Trips), Fy + 1)
'Average Trip Time by Mode
Base_TripTimeByMode() = AvgTripTimeByMode(Base_TripsByMode, Base_Times, Modes, Fy + 1)
'Average Distance Traveled by Mode
Base_AvgDistanceTraveled() = AvgDistanceTraveledByMode(Base_TripsByMode, Base_Times, Modes, Fy + 1)
'Average Uncongested Travel Time - Auto
Base_AvgUncongestedTime() = AvgUncongestedTime(Base_TripsByMode, Base_Times, Modes, Fy + 1)

'LoadBundle1 if selected
If FileExists(Sheet31.TextBox3.value) = True And FileExists(Sheet31.TextBox4.value) = True Then
    'Progress Bar
    Diag.Show

    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Loading Bundle1 Data ..."
    If Diag.cancelIsPressed Then Exit Sub
    'Load Matrices
    Bdl1_Trips() = ImportTripMtx(Sheet31.TextBox3.value)
    ImportBdl1_Time
    'Bdl1_Times() = ImportTimeMtx(Sheet31.TextBox4.value, Bdl1_Trips, Fy + 1 - 1)
    'Total Trips by mode
```

```
        Bdl1_TripsByMode() = TotalTripsByMode(Bdl1_Trips, UBound(Bdl1_Trips), 1)
        'Average trip time by mode
        Bdl1_TripTimeByMode() = AvgTripTimeByMode(Bdl1_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
        'Average Distance Traveled by Mode
        Bdl1_AvgDistanceTraveled() = AvgDistanceTraveledByMode(Bdl1_TripsByMode, Bdl1_Times, Modes, 1)
        'Average Uncongested Travel Time - Auto
        Bdl1_AvgUncongestedtime() = AvgUncongestedTime(Bdl1_TripsByMode, Bdl1_Times, Modes, 1)
        'Time Benefits
        Bdl1_TimeBenefits() = TimeBenefits(Base_Trips, Bdl1_Trips, Base_Times, Bdl1_Times, ODPairs, Modes, 1)

        Erase Bdl1_Trips
        Erase Bdl1_Times
    End If

     'Load Bundl2 if selected
     If FileExists(Sheet31.TextBox5.value) = True And FileExists(Sheet31.TextBox6.value) = True Then

        'Progress Bar
        Diag.Show

        j = j + 1
        Diag.SetValue j
        Diag.SetStatus "Loading Bundle2 Data ..."
        If Diag.cancelIsPressed Then Exit Sub

        'Load Matrices
        Bdl2_Trips() = ImportTripMtx(Sheet31.TextBox5.value)
        ImportBdl2_Time
       ' Bdl2_Times() = ImportTimeMtx(Sheet31.TextBox6.value, Bdl2_Trips, Fy + 1 - 1)
        'Total Trips by mode
        Bdl2_TripsByMode() = TotalTripsByMode(Bdl2_Trips, UBound(Bdl2_Trips), Fy + 1 - 1)
        'Average trip time by mode
        Bdl2_TripTimeByMode() = AvgTripTimeByMode(Bdl2_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
        'Average Distance Traveled by Mode
        Bdl2_AvgDistanceTraveled() = AvgDistanceTraveledByMode(Bdl2_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
        'Average Uncongested Travel Time - Auto
        Bdl2_AvgUncongestedtime() = AvgUncongestedTime(Bdl2_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
        'Time Benefits
        Bdl2_TimeBenefits() = TimeBenefits(Base_Trips, Bdl2_Trips, Base_Times, Bdl1_Times, ODPairs, Modes, 1)

        Erase Bdl2_Trips
        Erase Bdl1_Times
     End If

     'Load Bundl3 if selected
     If FileExists(Sheet31.TextBox7.value) = True And FileExists(Sheet31.TextBox8.value) = True Then

        'Progress Bar
        j = j + 1
        Diag.SetValue j
        Diag.SetStatus "Loading Bundle3 Data ..."
        If Diag.cancelIsPressed Then Exit Sub
        Diag.Show

        'Load Matrices
```

```
    Bdl3_Trips() = ImportTripMtx(Sheet31.TextBox7.value)
    ImportBdl3_Time
    'Bdl3_Times() = ImportTimeMtx(Sheet31.TextBox8.value, Bdl3_Trips, Fy + 1 - 1)
    'Total Trips by mode
    Bdl3_TripsByMode() = TotalTripsByMode(Bdl3_Trips, UBound(Bdl3_Trips), Fy + 1 - 1)
    'Average trip time by mode
    Bdl3_TripTimeByMode() = AvgTripTimeByMode(Bdl3_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Distance Traveled by Mode
    Bdl3_AvgDistanceTraveled() = AvgDistanceTraveledByMode(Bdl3_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Uncongested Travel Time - Auto
    Bdl3_AvgUncongestedtime() = AvgUncongestedTime(Bdl3_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Time Benefits
    Bdl3_TimeBenefits() = TimeBenefits(Base_Trips, Bdl3_Trips, Base_Times, Bdl1_Times, ODPairs, Modes, 1)

    Erase Bdl3_Trips
    Erase Bdl1_Times

End If

'Load Bundl4 if selected
If FileExists(Sheet31.TextBox9.value) = True And FileExists(Sheet31.TextBox10.value) = True Then

    'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Loading Bundle4 Data ..."
    If Diag.cancelIsPressed Then Exit Sub
    Diag.Show

    'Load Matrices
    Bdl4_Trips() = ImportTripMtx(Sheet31.TextBox9.value)
    ImportBdl4_Time
    'bdl1_times() = ImportTimeMtx(Sheet31.TextBox10.value, Bdl4_Trips, Fy + 1 - 1)
    'Total Trips by mode
    Bdl4_TripsByMode() = TotalTripsByMode(Bdl4_Trips, UBound(Bdl4_Trips), Fy + 1 - 1)
    'Average trip time by mode
    Bdl4_TripTimeByMode() = AvgTripTimeByMode(Bdl4_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Distance Traveled by Mode
    Bdl4_AvgDistanceTraveled() = AvgDistanceTraveledByMode(Bdl4_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Uncongested Travel Time - Auto
    Bdl4_AvgUncongestedtime() = AvgUncongestedTime(Bdl4_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Time Benefits
    Bdl4_TimeBenefits() = TimeBenefits(Base_Trips, Bdl4_Trips, Base_Times, Bdl1_Times, ODPairs, Modes, 1)

    Erase Bdl4_Trips
    Erase Bdl1_Times

End If

'Load Bundl5 if selected
If FileExists(Sheet31.TextBox11.value) = True And FileExists(Sheet31.TextBox12.value) = True Then

    'Progress Bar
    j = j + 1
    Diag.SetValue j
```

```
    Diag.SetStatus "Loading Bundle5 Data ..."
    If Diag.cancelIsPressed Then Exit Sub
    Diag.Show

    'Load Matrices
    Bdl5_Trips() = ImportTripMtx(Sheet31.TextBox11.value)
    ImportBdl5_Time
    'Bdl1_Times() = ImportTimeMtx(Sheet31.TextBox12.value, Bdl5_Trips, Fy + 1 - 1)
    'Total Trips by mode
    Bdl5_TripsByMode() = TotalTripsByMode(Bdl5_Trips, UBound(Bdl5_Trips), Fy + 1 - 1)
    'Average trip time by mode
    Bdl5_TripTimeByMode() = AvgTripTimeByMode(Bdl5_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Distance Traveled by Mode
    Bdl5_AvgDistanceTraveled() = AvgDistanceTraveledByMode(Bdl5_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Uncongested Travel Time - Auto
    Bdl5_AvgUncongestedtime() = AvgUncongestedTime(Bdl5_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Time Benefits
    Bdl5_TimeBenefits() = TimeBenefits(Base_Trips, Bdl5_Trips, Base_Times, Bdl1_Times, ODPairs, Modes, 1)

    Erase Bdl5_Trips
    Erase Bdl1_Times

End If

'Load Bundl6 if selected
If FileExists(Sheet31.TextBox13.value) = True And FileExists(Sheet31.TextBox14.value) = True Then

    'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Loading Bundle6 Data ..."
    If Diag.cancelIsPressed Then Exit Sub
    Diag.Show

    'Load Matrices
    Bdl6_Trips() = ImportTripMtx(Sheet31.TextBox13.value)
    ImportBdl6_Time
    'Bdl1_Times() = ImportTimeMtx(Sheet31.TextBox12.value, Bdl5_Trips, Fy + 1 - 1)
    'Total Trips by mode
    Bdl6_TripsByMode() = TotalTripsByMode(Bdl6_Trips, UBound(Bdl6_Trips), Fy + 1 - 1)
    'Average trip time by mode
    Bdl6_TripTimeByMode() = AvgTripTimeByMode(Bdl6_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Distance Traveled by Mode
    Bdl6_AvgDistanceTraveled() = AvgDistanceTraveledByMode(Bdl6_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Uncongested Travel Time - Auto
    Bdl6_AvgUncongestedtime() = AvgUncongestedTime(Bdl6_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Time Benefits
    Bdl6_TimeBenefits() = TimeBenefits(Base_Trips, Bdl6_Trips, Base_Times, Bdl1_Times, ODPairs, Modes, 1)

    Erase Bdl6_Trips
    Erase Bdl1_Times

End If

'Load Bundl7 if selected
```

```
If FileExists(Sheet31.TextBox15.value) = True And FileExists(Sheet31.TextBox16.value) = True Then

    'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Loading Bundle7 Data ..."
    If Diag.cancelIsPressed Then Exit Sub
    Diag.Show

    'Load Matrices
    Bdl7_Trips() = ImportTripMtx(Sheet31.TextBox15.value)
    ImportBdl7_Time
    'Bdl1_Times() = ImportTimeMtx(Sheet31.TextBox12.value, Bdl5_Trips, Fy + 1 - 1)
    'Total Trips by mode
    Bdl7_TripsByMode() = TotalTripsByMode(Bdl7_Trips, UBound(Bdl7_Trips), Fy + 1 - 1)
    'Average trip time by mode
    Bdl7_TripTimeByMode() = AvgTripTimeByMode(Bdl7_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Distance Traveled by Mode
    Bdl7_AvgDistanceTraveled() = AvgDistanceTraveledByMode(Bdl7_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Uncongested Travel Time - Auto
    Bdl7_AvgUncongestedtime() = AvgUncongestedTime(Bdl7_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Time Benefits
    Bdl7_TimeBenefits() = TimeBenefits(Base_Trips, Bdl7_Trips, Base_Times, Bdl1_Times, ODPairs, Modes, 1)

    Erase Bdl7_Trips
    Erase Bdl1_Times

End If

'Load Bundl8 if selected
If FileExists(Sheet31.TextBox17.value) = True And FileExists(Sheet31.TextBox18.value) = True Then

    'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Loading Bundle8 Data ..."
    If Diag.cancelIsPressed Then Exit Sub
    Diag.Show

    'Load Matrices
    Bdl8_Trips() = ImportTripMtx(Sheet31.TextBox17.value)
    Importbdl8_Time
    'Bdl1_Times() = ImportTimeMtx(Sheet31.TextBox12.value, Bdl5_Trips, Fy + 1 - 1)
    'Total Trips by mode
    Bdl8_TripsByMode() = TotalTripsByMode(Bdl8_Trips, UBound(Bdl8_Trips), Fy + 1 - 1)
    'Average trip time by mode
    Bdl8_TripTimeByMode() = AvgTripTimeByMode(Bdl8_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Distance Traveled by Mode
    Bdl8_AvgDistanceTraveled() = AvgDistanceTraveledByMode(Bdl8_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Uncongested Travel Time - Auto
    Bdl8_AvgUncongestedtime() = AvgUncongestedTime(Bdl8_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Time Benefits
    Bdl8_TimeBenefits() = TimeBenefits(Base_Trips, Bdl8_Trips, Base_Times, Bdl1_Times, ODPairs, Modes, 1)

    Erase Bdl8_Trips
```

```
    Erase Bdl1_Times

End If

'Load Bundl9 if selected
If FileExists(Sheet31.TextBox19.value) = True And FileExists(Sheet31.TextBox20.value) = True Then

    'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Loading Bundle9 Data ..."
    If Diag.cancelIsPressed Then Exit Sub
    Diag.Show

    'Load Matrices
    Bdl9_Trips() = ImportTripMtx(Sheet31.TextBox19.value)
    Importbdl9_Time
    'Bdl1_Times() = ImportTimeMtx(Sheet31.TextBox12.value, Bdl5_Trips, Fy + 1 - 1)
    'Total Trips by mode
    Bdl9_TripsByMode() = TotalTripsByMode(Bdl9_Trips, UBound(Bdl9_Trips), Fy + 1 - 1)
    'Average trip time by mode
    Bdl9_TripTimeByMode() = AvgTripTimeByMode(Bdl9_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Distance Traveled by Mode
    Bdl9_AvgDistanceTraveled() = AvgDistanceTraveledByMode(Bdl9_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Uncongested Travel Time - Auto
    Bdl9_AvgUncongestedtime() = AvgUncongestedTime(Bdl9_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Time Benefits
    Bdl9_TimeBenefits() = TimeBenefits(Base_Trips, Bdl9_Trips, Base_Times, Bdl1_Times, ODPairs, Modes, 1)

    Erase Bdl9_Trips
    Erase Bdl1_Times

End If

    'Load Bundl10 if selected
If FileExists(Sheet31.TextBox21.value) = True And FileExists(Sheet31.TextBox22.value) = True Then

    'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Loading Bundle10 Data ..."
    If Diag.cancelIsPressed Then Exit Sub
    Diag.Show

    'Load Matrices
    Bdl10_Trips() = ImportTripMtx(Sheet31.TextBox21.value)
    Importbdl10_Time
    'Bdl1_Times() = ImportTimeMtx(Sheet31.TextBox12.value, Bdl5_Trips, Fy + 1 - 1)
    'Total Trips by mode
    Bdl10_TripsByMode() = TotalTripsByMode(Bdl10_Trips, UBound(Bdl10_Trips), Fy + 1 - 1)
    'Average trip time by mode
    Bdl10_TripTimeByMode() = AvgTripTimeByMode(Bdl10_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Distance Traveled by Mode
    Bdl10_AvgDistanceTraveled() = AvgDistanceTraveledByMode(Bdl10_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
    'Average Uncongested Travel Time - Auto
```

```
        Bdl10_AvgUncongestedtime() = AvgUncongestedTime(Bdl10_TripsByMode, Bdl1_Times, Modes, Fy + 1 - 1)
        'Time Benefits
        Bdl10_TimeBenefits() = TimeBenefits(Base_Trips, Bdl10_Trips, Base_Times, Bdl1_Times, ODPairs, Modes, 1)

        Erase Bdl10_Trips
        Erase Bdl1_Times

    End If



    'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Calculating Mobility Benefits ..."
    If Diag.cancelIsPressed Then Exit Sub

    Set Sh = Sheet30
    Col = 4
    'Erase Previous Content
    Sh.Range("E15:AQ175").ClearContents
    Sh.Range("E348:AQ358").ClearContents

For Mj = 1 To Modes

    'Determine Vehicle Occupancy Rate
           VehOccupancy = 1

    'Number of Daily Trips

    For y = 1 To Fy + 1
        Row = 15
        If y = 1 Then
            Sh.Cells(Row, Col + y) = Base_TripsByMode(Mj, 1, y) + Base_TripsByMode(Mj, 2, y)
        Else
            Sh.Cells(Row, Col + y) = Base_TripsByMode(Mj, 1, y) + Base_TripsByMode(Mj, 2, y)
            Sh.Cells(Row + 1, Col + y) = Bdl1_TripsByMode(Mj, 1, y - 1) + Bdl1_TripsByMode(Mj, 2, y - 1)
            Sh.Cells(Row + 2, Col + y) = Bdl2_TripsByMode(Mj, 1, y - 1) + Bdl2_TripsByMode(Mj, 2, y - 1)
            Sh.Cells(Row + 3, Col + y) = Bdl3_TripsByMode(Mj, 1, y - 1) + Bdl3_TripsByMode(Mj, 2, y - 1)
            Sh.Cells(Row + 4, Col + y) = Bdl4_TripsByMode(Mj, 1, y - 1) + Bdl4_TripsByMode(Mj, 2, y - 1)
            Sh.Cells(Row + 5, Col + y) = Bdl5_TripsByMode(Mj, 1, y - 1) + Bdl5_TripsByMode(Mj, 2, y - 1)
            Sh.Cells(Row + 6, Col + y) = Bdl6_TripsByMode(Mj, 1, y - 1) + Bdl6_TripsByMode(Mj, 2, y - 1)
            Sh.Cells(Row + 7, Col + y) = Bdl7_TripsByMode(Mj, 1, y - 1) + Bdl7_TripsByMode(Mj, 2, y - 1)
            Sh.Cells(Row + 8, Col + y) = Bdl8_TripsByMode(Mj, 1, y - 1) + Bdl8_TripsByMode(Mj, 2, y - 1)
            Sh.Cells(Row + 9, Col + y) = Bdl9_TripsByMode(Mj, 1, y - 1) + Bdl9_TripsByMode(Mj, 2, y - 1)
            Sh.Cells(Row + 10, Col + y) = Bdl10_TripsByMode(Mj, 1, y - 1) + Bdl10_TripsByMode(Mj, 2, y - 1)
        End If
    'Number of Peak Period Trips
        Row = Row + 15 '30
        If y = 1 Then
            Sh.Cells(Row, Col + y) = Base_TripsByMode(Mj, 1, y)
        Else
            Sh.Cells(Row, Col + y) = Base_TripsByMode(Mj, 1, y)
            Sh.Cells(Row + 1, Col + y) = Bdl1_TripsByMode(Mj, 1, y - 1)
            Sh.Cells(Row + 2, Col + y) = Bdl2_TripsByMode(Mj, 1, y - 1)
```

```
            Sh.Cells(Row + 3, Col + y) = Bdl3_TripsByMode(Mj, 1, y - 1)
            Sh.Cells(Row + 4, Col + y) = Bdl4_TripsByMode(Mj, 1, y - 1)
            Sh.Cells(Row + 5, Col + y) = Bdl5_TripsByMode(Mj, 1, y - 1)
            Sh.Cells(Row + 6, Col + y) = Bdl6_TripsByMode(Mj, 1, y - 1)
            Sh.Cells(Row + 7, Col + y) = Bdl7_TripsByMode(Mj, 1, y - 1)
            Sh.Cells(Row + 8, Col + y) = Bdl8_TripsByMode(Mj, 1, y - 1)
            Sh.Cells(Row + 9, Col + y) = Bdl9_TripsByMode(Mj, 1, y - 1)
            Sh.Cells(Row + 10, Col + y) = Bdl10_TripsByMode(Mj, 1, y - 1)
        End If
    'Average Distance Traveled
        Row = Row + 15 '45
        If y = 1 Then
            Sh.Cells(Row, Col + y) = ((Base_AvgDistanceTraveled(Mj, 1, y) * Base_TripsByMode(Mj, 1, y)) + _
            (Base_AvgDistanceTraveled(Mj, 2, y) * Base_TripsByMode(Mj, 2, y))) / (Base_TripsByMode(Mj, 1, y) + Base_TripsByMode(Mj, 2, y))
        Else
            Sh.Cells(Row, Col + y) = ((Base_AvgDistanceTraveled(Mj, 1, y) * Base_TripsByMode(Mj, 1, y)) + _
                (Base_AvgDistanceTraveled(Mj, 2, y) * Base_TripsByMode(Mj, 2, y))) / (Base_TripsByMode(Mj, 1, y) + Base_TripsByMode(Mj, 2, y))

            Sh.Cells(Row + 1, Col + y) = ((Bdl1_AvgDistanceTraveled(Mj, 1, y - 1) * Bdl1_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl1_AvgDistanceTraveled(Mj, 2, y - 1) * Bdl1_TripsByMode(Mj, 2, y - 1))) / (Bdl1_TripsByMode(Mj, 1, y - 1) +
Bdl1_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 2, Col + y) = ((Bdl2_AvgDistanceTraveled(Mj, 1, y - 1) * Bdl2_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl2_AvgDistanceTraveled(Mj, 2, y - 1) * Bdl2_TripsByMode(Mj, 2, y - 1))) / (Bdl2_TripsByMode(Mj, 1, y - 1) +
Bdl2_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 3, Col + y) = ((Bdl3_AvgDistanceTraveled(Mj, 1, y - 1) * Bdl3_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl3_AvgDistanceTraveled(Mj, 2, y - 1) * Bdl3_TripsByMode(Mj, 2, y - 1))) / (Bdl3_TripsByMode(Mj, 1, y - 1) +
Bdl3_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 4, Col + y) = ((Bdl4_AvgDistanceTraveled(Mj, 1, y - 1) * Bdl4_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl4_AvgDistanceTraveled(Mj, 2, y - 1) * Bdl4_TripsByMode(Mj, 2, y - 1))) / (Bdl4_TripsByMode(Mj, 1, y - 1) +
Bdl4_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 5, Col + y) = ((Bdl5_AvgDistanceTraveled(Mj, 1, y - 1) * Bdl5_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl5_AvgDistanceTraveled(Mj, 2, y - 1) * Bdl5_TripsByMode(Mj, 2, y - 1))) / (Bdl5_TripsByMode(Mj, 1, y - 1) +
Bdl5_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 6, Col + y) = ((Bdl6_AvgDistanceTraveled(Mj, 1, y - 1) * Bdl6_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl6_AvgDistanceTraveled(Mj, 2, y - 1) * Bdl6_TripsByMode(Mj, 2, y - 1))) / (Bdl6_TripsByMode(Mj, 1, y - 1) +
Bdl6_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 7, Col + y) = ((Bdl7_AvgDistanceTraveled(Mj, 1, y - 1) * Bdl7_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl7_AvgDistanceTraveled(Mj, 2, y - 1) * Bdl7_TripsByMode(Mj, 2, y - 1))) / (Bdl7_TripsByMode(Mj, 1, y - 1) +
Bdl7_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 8, Col + y) = ((Bdl8_AvgDistanceTraveled(Mj, 1, y - 1) * Bdl8_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl8_AvgDistanceTraveled(Mj, 2, y - 1) * Bdl8_TripsByMode(Mj, 2, y - 1))) / (Bdl8_TripsByMode(Mj, 1, y - 1) +
Bdl8_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 9, Col + y) = ((Bdl9_AvgDistanceTraveled(Mj, 1, y - 1) * Bdl9_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl9_AvgDistanceTraveled(Mj, 2, y - 1) * Bdl9_TripsByMode(Mj, 2, y - 1))) / (Bdl9_TripsByMode(Mj, 1, y - 1) +
Bdl9_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 10, Col + y) = ((Bdl10_AvgDistanceTraveled(Mj, 1, y - 1) * Bdl10_TripsByMode(Mj, 1, y - 1)) + _
```

```
                (Bdl10_AvgDistanceTraveled(Mj, 2, y - 1) * Bdl10_TripsByMode(Mj, 2, y - 1))) / (Bdl10_TripsByMode(Mj, 1, y - 1) +
Bdl10_TripsByMode(Mj, 2, y - 1))
        End If

    'Average Travel Time - InVehicle
        Row = Row + 15 '60
        If y = 1 Then
            Sh.Cells(Row, Col + y) = ((Base_TripTimeByMode(Mj, 1, 1, y) * Base_TripsByMode(Mj, 1, y)) + _
            (Base_TripTimeByMode(Mj, 1, 2, y) * Base_TripsByMode(Mj, 2, y))) / (Base_TripsByMode(Mj, 1, y) + Base_TripsByMode(Mj, 2, y))
        Else
            Sh.Cells(Row, Col + y) = ((Base_TripTimeByMode(Mj, 1, 1, y) * Base_TripsByMode(Mj, 1, y)) + _
                (Base_TripTimeByMode(Mj, 1, 2, y) * Base_TripsByMode(Mj, 2, y))) / (Base_TripsByMode(Mj, 1, y) + Base_TripsByMode(Mj, 2, y))

            Sh.Cells(Row + 1, Col + y) = ((Bdl1_TripTimeByMode(Mj, 1, 1, y - 1) * Bdl1_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl1_TripTimeByMode(Mj, 1, 2, y - 1) * Bdl1_TripsByMode(Mj, 2, y - 1))) / (Bdl1_TripsByMode(Mj, 1, y - 1) +
Bdl1_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 2, Col + y) = ((Bdl2_TripTimeByMode(Mj, 1, 1, y - 1) * Bdl2_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl2_TripTimeByMode(Mj, 1, 2, y - 1) * Bdl2_TripsByMode(Mj, 2, y - 1))) / (Bdl2_TripsByMode(Mj, 1, y - 1) +
Bdl2_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 3, Col + y) = ((Bdl3_TripTimeByMode(Mj, 1, 1, y - 1) * Bdl3_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl3_TripTimeByMode(Mj, 1, 2, y - 1) * Bdl3_TripsByMode(Mj, 2, y - 1))) / (Bdl3_TripsByMode(Mj, 1, y - 1) +
Bdl3_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 4, Col + y) = ((Bdl4_TripTimeByMode(Mj, 1, 1, y - 1) * Bdl4_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl4_TripTimeByMode(Mj, 1, 2, y - 1) * Bdl4_TripsByMode(Mj, 2, y - 1))) / (Bdl4_TripsByMode(Mj, 1, y - 1) +
Bdl4_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 5, Col + y) = ((Bdl5_TripTimeByMode(Mj, 1, 1, y - 1) * Bdl5_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl5_TripTimeByMode(Mj, 1, 2, y - 1) * Bdl5_TripsByMode(Mj, 2, y - 1))) / (Bdl5_TripsByMode(Mj, 1, y - 1) +
Bdl5_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 6, Col + y) = ((Bdl6_TripTimeByMode(Mj, 1, 1, y - 1) * Bdl6_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl6_TripTimeByMode(Mj, 1, 2, y - 1) * Bdl6_TripsByMode(Mj, 2, y - 1))) / (Bdl6_TripsByMode(Mj, 1, y - 1) +
Bdl6_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 7, Col + y) = ((Bdl7_TripTimeByMode(Mj, 1, 1, y - 1) * Bdl7_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl7_TripTimeByMode(Mj, 1, 2, y - 1) * Bdl7_TripsByMode(Mj, 2, y - 1))) / (Bdl7_TripsByMode(Mj, 1, y - 1) +
Bdl7_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 8, Col + y) = ((Bdl8_TripTimeByMode(Mj, 1, 1, y - 1) * Bdl8_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl8_TripTimeByMode(Mj, 1, 2, y - 1) * Bdl8_TripsByMode(Mj, 2, y - 1))) / (Bdl8_TripsByMode(Mj, 1, y - 1) +
Bdl8_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 9, Col + y) = ((Bdl9_TripTimeByMode(Mj, 1, 1, y - 1) * Bdl9_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl9_TripTimeByMode(Mj, 1, 2, y - 1) * Bdl9_TripsByMode(Mj, 2, y - 1))) / (Bdl9_TripsByMode(Mj, 1, y - 1) +
Bdl9_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 10, Col + y) = ((Bdl10_TripTimeByMode(Mj, 1, 1, y - 1) * Bdl10_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl10_TripTimeByMode(Mj, 1, 2, y - 1) * Bdl10_TripsByMode(Mj, 2, y - 1))) / (Bdl10_TripsByMode(Mj, 1, y - 1) +
Bdl10_TripsByMode(Mj, 2, y - 1))
        End If

    'Average Travel Time - Access
```

```
        Row = Row + 15 '75
        If y = 1 Then
            Sh.Cells(Row, Col + y) = ((Base_TripTimeByMode(Mj, 2, 1, y) * Base_TripsByMode(Mj, 1, y)) + _
            (Base_TripTimeByMode(Mj, 2, 2, y) * Base_TripsByMode(Mj, 2, y))) / (Base_TripsByMode(Mj, 1, y) + Base_TripsByMode(Mj, 2, y))
        Else
            Sh.Cells(Row, Col + y) = ((Base_TripTimeByMode(Mj, 2, 1, y) * Base_TripsByMode(Mj, 1, y)) + _
            (Base_TripTimeByMode(Mj, 2, 2, y) * Base_TripsByMode(Mj, 2, y))) / (Base_TripsByMode(Mj, 1, y) + Base_TripsByMode(Mj, 2, y))

            Sh.Cells(Row + 1, Col + y) = ((Bdl1_TripTimeByMode(Mj, 2, 1, y - 1) * Bdl1_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl1_TripTimeByMode(Mj, 2, 2, y - 1) * Bdl1_TripsByMode(Mj, 2, y - 1))) / (Bdl1_TripsByMode(Mj, 1, y - 1) +
Bdl1_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 2, Col + y) = ((Bdl2_TripTimeByMode(Mj, 2, 1, y - 1) * Bdl2_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl2_TripTimeByMode(Mj, 2, 2, y - 1) * Bdl2_TripsByMode(Mj, 2, y - 1))) / (Bdl2_TripsByMode(Mj, 1, y - 1) +
Bdl2_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 3, Col + y) = ((Bdl3_TripTimeByMode(Mj, 2, 1, y - 1) * Bdl3_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl3_TripTimeByMode(Mj, 2, 2, y - 1) * Bdl3_TripsByMode(Mj, 2, y - 1))) / (Bdl3_TripsByMode(Mj, 1, y - 1) +
Bdl3_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 4, Col + y) = ((Bdl4_TripTimeByMode(Mj, 2, 1, y - 1) * Bdl4_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl4_TripTimeByMode(Mj, 2, 2, y - 1) * Bdl4_TripsByMode(Mj, 2, y - 1))) / (Bdl4_TripsByMode(Mj, 1, y - 1) +
Bdl4_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 5, Col + y) = ((Bdl5_TripTimeByMode(Mj, 2, 1, y - 1) * Bdl5_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl5_TripTimeByMode(Mj, 2, 2, y - 1) * Bdl5_TripsByMode(Mj, 2, y - 1))) / (Bdl5_TripsByMode(Mj, 1, y - 1) +
Bdl5_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 6, Col + y) = ((Bdl6_TripTimeByMode(Mj, 2, 1, y - 1) * Bdl6_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl6_TripTimeByMode(Mj, 2, 2, y - 1) * Bdl6_TripsByMode(Mj, 2, y - 1))) / (Bdl6_TripsByMode(Mj, 1, y - 1) +
Bdl6_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 7, Col + y) = ((Bdl7_TripTimeByMode(Mj, 2, 1, y - 1) * Bdl7_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl7_TripTimeByMode(Mj, 2, 2, y - 1) * Bdl7_TripsByMode(Mj, 2, y - 1))) / (Bdl7_TripsByMode(Mj, 1, y - 1) +
Bdl7_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 8, Col + y) = ((Bdl8_TripTimeByMode(Mj, 2, 1, y - 1) * Bdl8_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl8_TripTimeByMode(Mj, 2, 2, y - 1) * Bdl8_TripsByMode(Mj, 2, y - 1))) / (Bdl8_TripsByMode(Mj, 1, y - 1) +
Bdl8_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 9, Col + y) = ((Bdl9_TripTimeByMode(Mj, 2, 1, y - 1) * Bdl9_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl9_TripTimeByMode(Mj, 2, 2, y - 1) * Bdl9_TripsByMode(Mj, 2, y - 1))) / (Bdl9_TripsByMode(Mj, 1, y - 1) +
Bdl9_TripsByMode(Mj, 2, y - 1))

            Sh.Cells(Row + 10, Col + y) = ((Bdl10_TripTimeByMode(Mj, 2, 1, y - 1) * Bdl10_TripsByMode(Mj, 1, y - 1)) + _
                (Bdl10_TripTimeByMode(Mj, 2, 2, y - 1) * Bdl10_TripsByMode(Mj, 2, y - 1))) / (Bdl10_TripsByMode(Mj, 1, y - 1) +
Bdl10_TripsByMode(Mj, 2, y - 1))
        End If

      'Average Travel Time - Wait
        Row = Row + 15 '90
        If y = 1 Then
            Sh.Cells(Row, Col + y) = ((Base_TripTimeByMode(Mj, 3, 1, y) * Base_TripsByMode(Mj, 1, y)) + _
            (Base_TripTimeByMode(Mj, 3, 2, y) * Base_TripsByMode(Mj, 2, y))) / (Base_TripsByMode(Mj, 1, y) + Base_TripsByMode(Mj, 2, y))
        Else
```

```
            Sh.Cells(Row, Col + y) = ((Base_TripTimeByMode(Mj, 3, 1, y) * Base_TripsByMode(Mj, 1, y)) + _
           (Base_TripTimeByMode(Mj, 3, 2, y) * Base_TripsByMode(Mj, 2, y))) / (Base_TripsByMode(Mj, 1, y) + Base_TripsByMode(Mj, 2, y))

           Sh.Cells(Row + 1, Col + y) = ((Bdl1_TripTimeByMode(Mj, 3, 1, y - 1) * Bdl1_TripsByMode(Mj, 1, y - 1)) + _
               (Bdl1_TripTimeByMode(Mj, 3, 2, y - 1) * Bdl1_TripsByMode(Mj, 2, y - 1))) / (Bdl1_TripsByMode(Mj, 1, y - 1) +
Bdl1_TripsByMode(Mj, 2, y - 1))

           Sh.Cells(Row + 2, Col + y) = ((Bdl2_TripTimeByMode(Mj, 3, 1, y - 1) * Bdl2_TripsByMode(Mj, 1, y - 1)) + _
               (Bdl2_TripTimeByMode(Mj, 3, 2, y - 1) * Bdl2_TripsByMode(Mj, 2, y - 1))) / (Bdl2_TripsByMode(Mj, 1, y - 1) +
Bdl2_TripsByMode(Mj, 2, y - 1))

           Sh.Cells(Row + 3, Col + y) = ((Bdl3_TripTimeByMode(Mj, 3, 1, y - 1) * Bdl3_TripsByMode(Mj, 1, y - 1)) + _
               (Bdl3_TripTimeByMode(Mj, 3, 2, y - 1) * Bdl3_TripsByMode(Mj, 2, y - 1))) / (Bdl3_TripsByMode(Mj, 1, y - 1) +
Bdl3_TripsByMode(Mj, 2, y - 1))

           Sh.Cells(Row + 4, Col + y) = ((Bdl4_TripTimeByMode(Mj, 3, 1, y - 1) * Bdl4_TripsByMode(Mj, 1, y - 1)) + _
               (Bdl4_TripTimeByMode(Mj, 3, 2, y - 1) * Bdl4_TripsByMode(Mj, 2, y - 1))) / (Bdl4_TripsByMode(Mj, 1, y - 1) +
Bdl4_TripsByMode(Mj, 2, y - 1))

           Sh.Cells(Row + 5, Col + y) = ((Bdl5_TripTimeByMode(Mj, 3, 1, y - 1) * Bdl5_TripsByMode(Mj, 1, y - 1)) + _
               (Bdl5_TripTimeByMode(Mj, 3, 2, y - 1) * Bdl5_TripsByMode(Mj, 2, y - 1))) / (Bdl5_TripsByMode(Mj, 1, y - 1) +
Bdl5_TripsByMode(Mj, 1, y - 1))

           Sh.Cells(Row + 6, Col + y) = ((Bdl6_TripTimeByMode(Mj, 3, 1, y - 1) * Bdl6_TripsByMode(Mj, 1, y - 1)) + _
               (Bdl6_TripTimeByMode(Mj, 3, 2, y - 1) * Bdl6_TripsByMode(Mj, 2, y - 1))) / (Bdl6_TripsByMode(Mj, 1, y - 1) +
Bdl6_TripsByMode(Mj, 1, y - 1))

           Sh.Cells(Row + 7, Col + y) = ((Bdl7_TripTimeByMode(Mj, 3, 1, y - 1) * Bdl7_TripsByMode(Mj, 1, y - 1)) + _
               (Bdl7_TripTimeByMode(Mj, 3, 2, y - 1) * Bdl7_TripsByMode(Mj, 2, y - 1))) / (Bdl7_TripsByMode(Mj, 1, y - 1) +
Bdl7_TripsByMode(Mj, 2, y - 1))

           Sh.Cells(Row + 8, Col + y) = ((Bdl8_TripTimeByMode(Mj, 3, 1, y - 1) * Bdl8_TripsByMode(Mj, 1, y - 1)) + _
               (Bdl8_TripTimeByMode(Mj, 3, 2, y - 1) * Bdl8_TripsByMode(Mj, 1, y - 1))) / (Bdl8_TripsByMode(Mj, 1, y - 1) +
Bdl8_TripsByMode(Mj, 2, y - 1))

           Sh.Cells(Row + 9, Col + y) = ((Bdl9_TripTimeByMode(Mj, 3, 1, y - 1) * Bdl9_TripsByMode(Mj, 1, y - 1)) + _
               (Bdl9_TripTimeByMode(Mj, 3, 2, y - 1) * Bdl9_TripsByMode(Mj, 2, y - 1))) / (Bdl9_TripsByMode(Mj, 1, y - 1) +
Bdl9_TripsByMode(Mj, 2, y - 1))

           Sh.Cells(Row + 10, Col + y) = ((Bdl10_TripTimeByMode(Mj, 3, 1, y - 1) * Bdl10_TripsByMode(Mj, 1, y - 1)) + _
               (Bdl10_TripTimeByMode(Mj, 3, 2, y - 1) * Bdl10_TripsByMode(Mj, 2, y - 1))) / (Bdl10_TripsByMode(Mj, 1, y - 1) +
Bdl10_TripsByMode(Mj, 2, y - 1))
       End If

       'Congested Travel - For First Mode Peak Period Only
       Row = Row + 15 '105
       If Mj = 1 Then

           If y = 1 Then
               Sh.Cells(Row, Col + y) = Base_TripTimeByMode(Mj, 1, 1, y) - Base_AvgUncongestedTime(Mj, 1, y)
           Else
               Sh.Cells(Row, Col + y) = Base_TripTimeByMode(Mj, 1, 1, y) - Base_AvgUncongestedTime(Mj, 1, y)
               Sh.Cells(Row + 1, Col + y) = Bdl1_TripTimeByMode(Mj, 1, 1, y - 1) - Bdl1_AvgUncongestedtime(Mj, 1, y - 1)
               Sh.Cells(Row + 2, Col + y) = Bdl2_TripTimeByMode(Mj, 1, 1, y - 1) - Bdl2_AvgUncongestedtime(Mj, 1, y - 1)
               Sh.Cells(Row + 3, Col + y) = Bdl3_TripTimeByMode(Mj, 1, 1, y - 1) - Bdl3_AvgUncongestedtime(Mj, 1, y - 1)
```

```
        Sh.Cells(Row + 4, Col + y) = Bdl4_TripTimeByMode(Mj, 1, 1, y - 1) - Bdl4_AvgUncongestedtime(Mj, 1, y - 1)
        Sh.Cells(Row + 5, Col + y) = Bdl5_TripTimeByMode(Mj, 1, 1, y - 1) - Bdl5_AvgUncongestedtime(Mj, 1, y - 1)
        Sh.Cells(Row + 6, Col + y) = Bdl6_TripTimeByMode(Mj, 1, 1, y - 1) - Bdl6_AvgUncongestedtime(Mj, 1, y - 1)
        Sh.Cells(Row + 7, Col + y) = Bdl7_TripTimeByMode(Mj, 1, 1, y - 1) - Bdl7_AvgUncongestedtime(Mj, 1, y - 1)
        Sh.Cells(Row + 8, Col + y) = Bdl8_TripTimeByMode(Mj, 1, 1, y - 1) - Bdl8_AvgUncongestedtime(Mj, 1, y - 1)
        Sh.Cells(Row + 9, Col + y) = Bdl9_TripTimeByMode(Mj, 1, 1, y - 1) - Bdl9_AvgUncongestedtime(Mj, 1, y - 1)
        Sh.Cells(Row + 10, Col + y) = Bdl10_TripTimeByMode(Mj, 1, 1, y - 1) - Bdl10_AvgUncongestedtime(Mj, 1, y - 1)
    End If
End If


'Travel Time Savings - Invehicle
Row = Row + 15 '120
If y > 1 Then

        Sh.Cells(Row + 1, Col + y) = (Bdl1_TimeBenefits(Mj, 1, 1, y - 1) + Bdl1_TimeBenefits(Mj, 1, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 2, Col + y) = (Bdl2_TimeBenefits(Mj, 1, 1, y - 1) + Bdl2_TimeBenefits(Mj, 1, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 3, Col + y) = (Bdl3_TimeBenefits(Mj, 1, 1, y - 1) + Bdl3_TimeBenefits(Mj, 1, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 4, Col + y) = (Bdl4_TimeBenefits(Mj, 1, 1, y - 1) + Bdl4_TimeBenefits(Mj, 1, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 5, Col + y) = (Bdl5_TimeBenefits(Mj, 1, 1, y - 1) + Bdl5_TimeBenefits(Mj, 1, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 6, Col + y) = (Bdl6_TimeBenefits(Mj, 1, 1, y - 1) + Bdl6_TimeBenefits(Mj, 1, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 7, Col + y) = (Bdl7_TimeBenefits(Mj, 1, 1, y - 1) + Bdl7_TimeBenefits(Mj, 1, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 8, Col + y) = (Bdl8_TimeBenefits(Mj, 1, 1, y - 1) + Bdl8_TimeBenefits(Mj, 1, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 9, Col + y) = (Bdl9_TimeBenefits(Mj, 1, 1, y - 1) + Bdl9_TimeBenefits(Mj, 1, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 10, Col + y) = (Bdl10_TimeBenefits(Mj, 1, 1, y - 1) + Bdl10_TimeBenefits(Mj, 1, 2, y - 1)) / 60 * VehOccupancy
End If

'Travel Time Savings - Access
Row = Row + 15 '135
If y > 1 Then

        Sh.Cells(Row + 1, Col + y) = (Bdl1_TimeBenefits(Mj, 2, 1, y - 1) + Bdl1_TimeBenefits(Mj, 2, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 2, Col + y) = (Bdl2_TimeBenefits(Mj, 2, 1, y - 1) + Bdl2_TimeBenefits(Mj, 2, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 3, Col + y) = (Bdl3_TimeBenefits(Mj, 2, 1, y - 1) + Bdl3_TimeBenefits(Mj, 2, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 4, Col + y) = (Bdl4_TimeBenefits(Mj, 2, 1, y - 1) + Bdl4_TimeBenefits(Mj, 2, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 5, Col + y) = (Bdl5_TimeBenefits(Mj, 2, 1, y - 1) + Bdl5_TimeBenefits(Mj, 2, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 6, Col + y) = (Bdl6_TimeBenefits(Mj, 2, 1, y - 1) + Bdl6_TimeBenefits(Mj, 2, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 7, Col + y) = (Bdl7_TimeBenefits(Mj, 2, 1, y - 1) + Bdl7_TimeBenefits(Mj, 2, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 8, Col + y) = (Bdl8_TimeBenefits(Mj, 2, 1, y - 1) + Bdl8_TimeBenefits(Mj, 2, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 9, Col + y) = (Bdl9_TimeBenefits(Mj, 2, 1, y - 1) + Bdl9_TimeBenefits(Mj, 2, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 10, Col + y) = (Bdl10_TimeBenefits(Mj, 2, 1, y - 1) + Bdl10_TimeBenefits(Mj, 2, 2, y - 1)) / 60 * VehOccupancy
End If

'Travel Time Savings - WAIT
Row = Row + 15 '150
If y > 1 Then

        Sh.Cells(Row + 1, Col + y) = (Bdl1_TimeBenefits(Mj, 3, 1, y - 1) + Bdl1_TimeBenefits(Mj, 3, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 2, Col + y) = (Bdl2_TimeBenefits(Mj, 3, 1, y - 1) + Bdl2_TimeBenefits(Mj, 3, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 3, Col + y) = (Bdl3_TimeBenefits(Mj, 3, 1, y - 1) + Bdl3_TimeBenefits(Mj, 3, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 4, Col + y) = (Bdl4_TimeBenefits(Mj, 3, 1, y - 1) + Bdl4_TimeBenefits(Mj, 3, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 5, Col + y) = (Bdl5_TimeBenefits(Mj, 3, 1, y - 1) + Bdl5_TimeBenefits(Mj, 3, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 6, Col + y) = (Bdl6_TimeBenefits(Mj, 3, 1, y - 1) + Bdl6_TimeBenefits(Mj, 3, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 7, Col + y) = (Bdl7_TimeBenefits(Mj, 3, 1, y - 1) + Bdl7_TimeBenefits(Mj, 3, 2, y - 1)) / 60 * VehOccupancy
        Sh.Cells(Row + 8, Col + y) = (Bdl8_TimeBenefits(Mj, 3, 1, y - 1) + Bdl8_TimeBenefits(Mj, 3, 2, y - 1)) / 60 * VehOccupancy
```

```
            Sh.Cells(Row + 9, Col + y) = (Bdl9_TimeBenefits(Mj, 3, 1, y - 1) + Bdl9_TimeBenefits(Mj, 3, 2, y - 1)) / 60 * VehOccupancy
            Sh.Cells(Row + 10, Col + y) = (Bdl10_TimeBenefits(Mj, 3, 1, y - 1) + Bdl10_TimeBenefits(Mj, 3, 2, y - 1)) / 60 * VehOccupancy
        End If

        'Average Buffer Time Due to Recurring Delay
        Row = Row + 15 '165
'        If y = 1 Then
'            Sh.Cells(Row, Col + y) = BufferTimeIndex(1, y) * Base_TripTimeByMode(Mj, 1, 1, y)
'        Else
'            Sh.Cells(Row, Col + y) = BufferTimeIndex(1, y) * Base_TripTimeByMode(Mj, 1, 1, y)
'            Sh.Cells(Row + 1, Col + y) = BufferTimeIndex(2, y) * Bdl1_TripTimeByMode(Mj, 1, 1, y - 1)
'            Sh.Cells(Row + 2, Col + y) = BufferTimeIndex(3, y) * Bdl2_TripTimeByMode(Mj, 1, 1, y - 1)
'            Sh.Cells(Row + 3, Col + y) = BufferTimeIndex(4, y) * Bdl3_TripTimeByMode(Mj, 1, 1, y - 1)
'            Sh.Cells(Row + 4, Col + y) = BufferTimeIndex(5, y) * Bdl4_TripTimeByMode(Mj, 1, 1, y - 1)
'            Sh.Cells(Row + 5, Col + y) = BufferTimeIndex(6, y) * Bdl5_TripTimeByMode(Mj, 1, 1, y - 1)
'            Sh.Cells(Row + 6, Col + y) = BufferTimeIndex(7, y) * Bdl6_TripTimeByMode(Mj, 1, 1, y - 1)
'            Sh.Cells(Row + 7, Col + y) = BufferTimeIndex(8, y) * Bdl7_TripTimeByMode(Mj, 1, 1, y - 1)
'            Sh.Cells(Row + 8, Col + y) = BufferTimeIndex(9, y) * Bdl8_TripTimeByMode(Mj, 1, 1, y - 1)
'            Sh.Cells(Row + 9, Col + y) = BufferTimeIndex(10, y) * Bdl9_TripTimeByMode(Mj, 1, 1, y - 1)
'            Sh.Cells(Row + 10, Col + y) = BufferTimeIndex(11, y) * Bdl10_TripTimeByMode(Mj, 1, 1, y - 1)
'
'        End If

        If Mj = 1 Then

            If y = 1 Then
                Sh.Cells(Row, Col + y) = Base_TripTimeByMode(Mj, 1, 1, y) / Base_AvgUncongestedTime(Mj, 1, y)
            Else
                Sh.Cells(Row, Col + y) = Base_TripTimeByMode(Mj, 1, 1, y) / Base_AvgUncongestedTime(Mj, 1, y)
                Sh.Cells(Row + 1, Col + y) = Bdl1_TripTimeByMode(Mj, 1, 1, y - 1) / Bdl1_AvgUncongestedtime(Mj, 1, y - 1)
                Sh.Cells(Row + 2, Col + y) = Bdl2_TripTimeByMode(Mj, 1, 1, y - 1) / Bdl2_AvgUncongestedtime(Mj, 1, y - 1)
                Sh.Cells(Row + 3, Col + y) = Bdl3_TripTimeByMode(Mj, 1, 1, y - 1) / Bdl3_AvgUncongestedtime(Mj, 1, y - 1)
                Sh.Cells(Row + 4, Col + y) = Bdl4_TripTimeByMode(Mj, 1, 1, y - 1) / Bdl4_AvgUncongestedtime(Mj, 1, y - 1)
                Sh.Cells(Row + 5, Col + y) = Bdl5_TripTimeByMode(Mj, 1, 1, y - 1) / Bdl5_AvgUncongestedtime(Mj, 1, y - 1)
                Sh.Cells(Row + 6, Col + y) = Bdl6_TripTimeByMode(Mj, 1, 1, y - 1) / Bdl6_AvgUncongestedtime(Mj, 1, y - 1)
                Sh.Cells(Row + 7, Col + y) = Bdl7_TripTimeByMode(Mj, 1, 1, y - 1) / Bdl7_AvgUncongestedtime(Mj, 1, y - 1)
                Sh.Cells(Row + 8, Col + y) = Bdl8_TripTimeByMode(Mj, 1, 1, y - 1) / Bdl8_AvgUncongestedtime(Mj, 1, y - 1)
                Sh.Cells(Row + 9, Col + y) = Bdl9_TripTimeByMode(Mj, 1, 1, y - 1) / Bdl9_AvgUncongestedtime(Mj, 1, y - 1)
                Sh.Cells(Row + 10, Col + y) = Bdl10_TripTimeByMode(Mj, 1, 1, y - 1) / Bdl10_AvgUncongestedtime(Mj, 1, y - 1)
            End If
        End If

        'AVERAGE BUFFER TIME DUE TO NON-RECURRING DELAY, minutes per peak-period trip
        Row = Row + 183 '348
        If Mj = 1 Then

            If y = 1 Then
            Sh.Cells(Row, Col + y) = BufferTimeIndex(1, y) * Base_TripTimeByMode(Mj, 1, 1, y)
            Else
                Sh.Cells(Row, Col + y) = BufferTimeIndex(1, y) * Base_TripTimeByMode(Mj, 1, 1, y)
                Sh.Cells(Row + 1, Col + y) = BufferTimeIndex(2, y) * Bdl1_TripTimeByMode(Mj, 1, 1, y - 1)
                Sh.Cells(Row + 2, Col + y) = BufferTimeIndex(3, y) * Bdl2_TripTimeByMode(Mj, 1, 1, y - 1)
                Sh.Cells(Row + 3, Col + y) = BufferTimeIndex(4, y) * Bdl3_TripTimeByMode(Mj, 1, 1, y - 1)
                Sh.Cells(Row + 4, Col + y) = BufferTimeIndex(5, y) * Bdl4_TripTimeByMode(Mj, 1, 1, y - 1)
                Sh.Cells(Row + 5, Col + y) = BufferTimeIndex(6, y) * Bdl5_TripTimeByMode(Mj, 1, 1, y - 1)
```

```
                Sh.Cells(Row + 6, Col + y) = BufferTimeIndex(7, y) * Bdl6_TripTimeByMode(Mj, 1, 1, y - 1)
                Sh.Cells(Row + 7, Col + y) = BufferTimeIndex(8, y) * Bdl7_TripTimeByMode(Mj, 1, 1, y - 1)
                Sh.Cells(Row + 8, Col + y) = BufferTimeIndex(9, y) * Bdl8_TripTimeByMode(Mj, 1, 1, y - 1)
                Sh.Cells(Row + 9, Col + y) = BufferTimeIndex(10, y) * Bdl9_TripTimeByMode(Mj, 1, 1, y - 1)
                Sh.Cells(Row + 10, Col + y) = BufferTimeIndex(11, y) * Bdl10_TripTimeByMode(Mj, 1, 1, y - 1)
            End If
        End If

    Next y
    Col = Col + 5
Next Mj

    'Progress Bar
    j = j + 1
    Diag.SetValue j
    Diag.SetStatus "Pasting Trip & Cost Data ..."
    Diag.Show


'Dim cbrReset As CommandBarButton
'
'Set cbrReset = Application.VBE.CommandBars(1).Controls("&Run").Controls("&Reset")
'cbrReset.Execute

    Sh.Select
    Sh.Range("A9").Select

    If CurrentStatus = "Locked for Editing" Then
        'Lock Model
        Sheet2.Range("F21") = "locked for Editing"
        MOSAIC_ProtectModel

    End If

    'Application.Calculation = xlCalculationManual
    Application.Calculation = xlCalculationAutomatic
    Application.ScreenUpdating = True
    Diag.Hide


End Sub

Public Function ImportTripMtx(FileName As String) As Trips_Mtx2()

Dim ImpRng As Range
Dim Pastetrips2() As Trips_Mtx2
Dim Year As Integer
Dim c As Integer
Dim txt As String
Dim char As String * 1  'Read the line one Char at a time
Dim data As String
Dim i, Fy, NumberOfImputs As Integer
Dim FileNum As Integer
Dim Mode As Integer
```

```
FileNum = FreeFile
ReDim Pastetrips2(1 To ODPairs)

Open FileName For Input As #FileNum

'If Err <> 0 Then
'    MsgBox "not found: " & FileName, vbCritical, "ERROR"
'    'Exit Function
'End If

r = 1
txt = ""

'Read Trip File
Do Until EOF(FileNum)
    Line Input #FileNum, data 'Read a line
    txt = ""
    For i = 1 To Len(data) + 1
        char = Mid(data, i, 1)
        If char = "," Or i = Len(data) + 1 Then 'comma
            'Store Data
            c = c + 1
            Select Case c
            Case 1
                'Origin
                    Pastetrips2(r).Origin = txt
            Case 2
                'Destination
                    Pastetrips2(r).Destination = txt
                    c = c + 1
            End Select

            Select Case Mode
            'Peak Traffic
            Case 1, 17, 33, 49
                'Mode 1 - Peak
                Year = Mode Mod 5
                Pastetrips2(r).Trips(1, 1, Year) = txt
                Mode = Mode + 1
            Case 2, 18, 34, 50
                'Mode 2
                Year = (Mode - 1) Mod 5
                Pastetrips2(r).Trips(2, 1, Year) = txt
                Mode = Mode + 1
            Case 3, 19, 35, 51
                'Mode 3
                Year = (Mode - 2) Mod 5
                Pastetrips2(r).Trips(3, 1, Year) = txt
                Mode = Mode + 1
           Case 4, 20, 36, 52
                'Mode 4
                Year = (Mode - 3) Mod 5
                Pastetrips2(r).Trips(4, 1, Year) = txt
                Mode = Mode + 1
            Case 5, 21, 37, 53
```

```
        'Mode 5
        Year = (Mode - 4) Mod 5
        Pastetrips2(r).Trips(5, 1, Year) = txt
        Mode = Mode + 1
     Case 6, 22, 38, 54
        'Mode 6
        Year = (Mode - 5) Mod 5
        Pastetrips2(r).Trips(6, 1, Year) = txt
        Mode = Mode + 1
     Case 7, 23, 39, 55
        'Mode 7
        Year = (Mode - 6) Mod 5
        Pastetrips2(r).Trips(7, 1, Year) = txt
        Mode = Mode + 1
     Case 8, 24, 40, 56
        'Mode 8
        Year = (Mode - 7) Mod 5
        Pastetrips2(r).Trips(8, 1, Year) = txt
        Mode = Mode + 1
    'Off Peak Traffic
    Case 9, 25, 41, 57
        'Mode 1
        Year = (Mode - 8) Mod 5
        Pastetrips2(r).Trips(1, 2, Year) = txt
        Mode = Mode + 1
    Case 10, 26, 42, 58
        'Mode 2
        Year = (Mode - 9) Mod 5
        Pastetrips2(r).Trips(2, 2, Year) = txt
        Mode = Mode + 1
    Case 11, 27, 43, 59
        'Mode 3
        Year = (Mode - 10) Mod 5
        Pastetrips2(r).Trips(3, 2, Year) = txt
        Mode = Mode + 1
   Case 12, 28, 44, 60
        'Mode 4
        Year = (Mode - 11) Mod 5
        Pastetrips2(r).Trips(4, 2, Year) = txt
        Mode = Mode + 1
    Case 13, 29, 45, 61
        'Mode 5
        Year = (Mode - 12) Mod 5
        Pastetrips2(r).Trips(5, 2, Year) = txt
        Mode = Mode + 1
     Case 14, 30, 46, 62
        'Mode 6
        Year = (Mode - 13) Mod 5
        Pastetrips2(r).Trips(6, 2, Year) = txt
        Mode = Mode + 1
     Case 15, 31, 47, 63
        'Mode 7
        Year = (Mode - 14) Mod 5
        Pastetrips2(r).Trips(7, 2, Year) = txt
        Mode = Mode + 1
```

```
             Case 16, 32, 48, 64
                'Mode 8
                Year = (Mode - 15) Mod 5
                Pastetrips2(r).Trips(8, 2, Year) = txt
                Mode = Mode + 1
            End Select

        txt = ""
        ElseIf i = Len(data) Then 'End of File
            If char <> Chr(34) Then
                txt = txt & char

            End If

        ElseIf char <> Chr(34) Then
            txt = txt & char
        End If

        If c = 3 Then
            Mode = Mode + 1
            c = 5
        End If
    Next i


    c = 0
    r = r + 1
    Mode = 0
Loop

Close #FileNum

'return array
ReDim Preserve Pastetrips2(1 To r - 1)
ImportTripMtx = Pastetrips2()

ReDim Pastetrips2(1 To 1)
Erase Pastetrips2

End Function
Sub ImportBaseTime()
    Dim c As Integer
    Dim txt As String
    Dim char As String * 1  'Read the line one Char at a time
    Dim data
    Dim i, Column As Integer
    Dim PasteTimes() As Double
    ReDim PasteTimes(1 To ODPairs, 1 To MaxCol)
    Dim MaxModes As Integer
    Dim k As Integer
    Dim OffPeakIncrement As Integer
    Dim FileName As String
    Dim Fy As Integer
    Dim FileNum As Integer
```

```
FileNum = FreeFile
Fy = Sheet27.Range("A18") + 1


FileName = Sheet31.TextBox2.value

MaxModes = Sheet9.Cells(12, 3)
OffPeakIncrement = 24 - Sheet9.Cells(15, 15)

Open FileName For Input As #FileNum

'Read File
r = 1
c = 0
txt = ""
Do Until EOF(FileNum)
    Line Input #FileNum, data 'Read a line
    txt = ""
    For i = 1 To Len(data) + 1

        char = Mid(data, i, 1)
        If char = "," Or i = Len(data) + 1 Then 'comma
            'Store Data
            c = c + 1
            PasteTimes(r, c) = txt
            txt = ""
            Column = c
        ElseIf i = Len(data) Then 'End of File
            If char <> Chr(34) Then txt = txt & char

        ElseIf char <> Chr(34) Then
            txt = txt & char
        End If

    Next i
    c = 0
    r = r + 1
    Mode = 0
Loop
Close #FileNum

'ReDim Base_Times(1 To r - 1)

For i = 1 To r - 1
    Base_Times(i).Origin = PasteTimes(i, 1)
    Base_Times(i).Destination = PasteTimes(i, 2)
    Base_Times(i).Distance = PasteTimes(i, 3)
    k = 4
    For j = 1 To Fy
        Base_Times(i).UncongestedTime(j) = PasteTimes(i, k)
        k = k + 1
        For Mode = 1 To MaxModes
            If Sheet9.Cells(14 + Mode, 12) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Base_Times(i).InVehTime(Mode, 1, j) = 0
```

```
    Else
        Base_Times(i).InVehTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Base_Times(i).InVehTime(Mode, 2, j) = 0
    Else
        Base_Times(i).InVehTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Base_Times(i).InVehTime(Mode, 1, j) = 0
    Base_Times(i).InVehTime(Mode, 2, j) = 0
End If
If Sheet9.Cells(14 + Mode, 13) = 1 Then
    If PasteTimes(i, k) >= 9999 Then
        Base_Times(i).AccessTime(Mode, 1, j) = 0
    Else
        Base_Times(i).AccessTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Base_Times(i).AccessTime(Mode, 2, j) = 0
    Else
        Base_Times(i).AccessTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Base_Times(i).AccessTime(Mode, 1, j) = 0
    Base_Times(i).AccessTime(Mode, 2, j) = 0
End If
If Sheet9.Cells(14 + Mode, 14) = 1 Then
    If PasteTimes(i, k) >= 9999 Then
        Base_Times(i).WaitTime(Mode, 1, j) = 0
    Else
        Base_Times(i).WaitTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Base_Times(i).WaitTime(Mode, 2, j) = 0
    Else
        Base_Times(i).WaitTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Base_Times(i).WaitTime(Mode, 1, j) = 0
    Base_Times(i).WaitTime(Mode, 2, j) = 0
End If

'Sum of Trip time multiplied by number of trips 'Stored in the row if the matrix
Base_Times(1).InVehTimesTrips(Mode, 1, j) = Base_Times(1).InVehTimesTrips(Mode, 1, j) + _
            (Base_Times(i).InVehTime(Mode, 1, j) * Base_Trips(i).Trips(Mode, 1, j))
Base_Times(1).InVehTimesTrips(Mode, 2, j) = Base_Times(1).InVehTimesTrips(Mode, 2, j) + _
            (Base_Times(i).InVehTime(Mode, 2, j) * Base_Trips(i).Trips(Mode, 2, j))

Base_Times(1).AccessTimesTrips(Mode, 1, j) = Base_Times(1).AccessTimesTrips(Mode, 1, j) + _
            (Base_Times(i).AccessTime(Mode, 1, j) * Base_Trips(i).Trips(Mode, 1, j))
Base_Times(1).AccessTimesTrips(Mode, 2, j) = Base_Times(1).AccessTimesTrips(Mode, 2, j) + _
```

```
                            (Base_Times(i).AccessTime(Mode, 2, j) * Base_Trips(i).Trips(Mode, 2, j))

                Base_Times(1).WaitTimesTrips(Mode, 1, j) = Base_Times(1).WaitTimesTrips(Mode, 1, j) + _
                            (Base_Times(i).WaitTime(Mode, 1, j) * Base_Trips(i).Trips(Mode, 1, j))
                Base_Times(1).WaitTimesTrips(Mode, 2, j) = Base_Times(1).WaitTimesTrips(Mode, 2, j) + _
                            (Base_Times(i).WaitTime(Mode, 2, j) * Base_Trips(i).Trips(Mode, 2, j))

                'Sum of distance Multiplied by Number of trips ' Stored in the first row of the matrix
                Base_Times(1).DistanceTimesTrips(Mode, 1, j) = Base_Times(1).DistanceTimesTrips(Mode, 1, j) + _
                            (Base_Times(i).Distance * Base_Trips(i).Trips(Mode, 1, j))
                Base_Times(1).DistanceTimesTrips(Mode, 2, j) = Base_Times(1).DistanceTimesTrips(Mode, 2, j) + _
                            (Base_Times(i).Distance * Base_Trips(i).Trips(Mode, 2, j))

                'Sum of uncongested time Multiplied by Number of Trips ' Stored in the first row of the matrix
                'Peak
                Base_Times(1).UncongestedTimesTrips(Mode, 1, j) = Base_Times(1).UncongestedTimesTrips(Mode, 1, j) + _
                            (Base_Times(i).UncongestedTime(j) * Base_Trips(i).Trips(Mode, 1, j))
                'OffPeak
                Base_Times(1).UncongestedTimesTrips(Mode, 2, j) = Base_Times(1).UncongestedTimesTrips(Mode, 2, j) + _
                            (Base_Times(i).UncongestedTime(j) * Base_Trips(i).Trips(Mode, 1, j))


            Next Mode
            k = k + OffPeakIncrement
        Next j
    Next i

    Erase PasteTimes

End Sub

Sub ImportBdl1_Time()
    Dim c As Integer
    Dim txt As String
    Dim char As String * 1  'Read the line one Char at a time
    Dim data
    Dim i, Column As Integer
    Dim PasteTimes() As Double
    ReDim PasteTimes(1 To ODPairs, 1 To MaxCol)
    Dim MaxModes As Integer
    Dim k As Integer
    Dim OffPeakIncrement As Integer
    Dim FileName As String
    Dim Fy As Integer
    Dim FileNum As Integer

    FileNum = FreeFile
    Fy = Sheet27.Range("A18") + 1


    FileName = Sheet31.TextBox4.value

    MaxModes = Sheet9.Cells(12, 3)
    OffPeakIncrement = 24 - Sheet9.Cells(15, 15)
```

```
Open FileName For Input As #FileNum

'Read File
r = 1
c = 0
txt = ""
Do Until EOF(FileNum)
    Line Input #FileNum, data 'Read a line
    txt = ""
    For i = 1 To Len(data) + 1

        char = Mid(data, i, 1)
        If char = "," Or i = Len(data) + 1 Then 'comma
            'Store Data
            c = c + 1
            PasteTimes(r, c) = txt
            txt = ""
            Column = c
        ElseIf i = Len(data) Then 'End of File
            If char <> Chr(34) Then txt = txt & char

        ElseIf char <> Chr(34) Then
            txt = txt & char
        End If

    Next i
    c = 0
    r = r + 1
    Mode = 0
Loop
Close #FileNum

'ReDim Bdl1_Times(1 To r - 1)

For i = 1 To r - 1
    Bdl1_Times(i).Origin = PasteTimes(i, 1)
    Bdl1_Times(i).Destination = PasteTimes(i, 2)
    Bdl1_Times(i).Distance = PasteTimes(i, 3)
    k = 4
    For j = 1 To Fy
        Bdl1_Times(i).UncongestedTime(j) = PasteTimes(i, k)
        k = k + 1
        For Mode = 1 To MaxModes
            If Sheet9.Cells(14 + Mode, 12) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
```

```
Else
    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
    Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
End If
If Sheet9.Cells(14 + Mode, 13) = 1 Then
    If PasteTimes(i, k) >= 9999 Then
        Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
    Else
        Bdl1_Times(i).AccessTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
    Else
        Bdl1_Times(i).AccessTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
    Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
End If
If Sheet9.Cells(14 + Mode, 14) = 1 Then
    If PasteTimes(i, k) >= 9999 Then
        Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
    Else
        Bdl1_Times(i).WaitTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
    Else
        Bdl1_Times(i).WaitTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
    Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
End If

'Sum of Trip time multiplied by number of trips 'Stored in the row if the matrix
Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).InVehTime(Mode, 1, j) * Bdl1_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).InVehTime(Mode, 2, j) * Bdl1_Trips(i).Trips(Mode, 2, j))

Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).AccessTime(Mode, 1, j) * Bdl1_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).AccessTime(Mode, 2, j) * Bdl1_Trips(i).Trips(Mode, 2, j))

Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).WaitTime(Mode, 1, j) * Bdl1_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).WaitTime(Mode, 2, j) * Bdl1_Trips(i).Trips(Mode, 2, j))

'Sum of distance Multiplied by Number of trips ' Stored in the first row of the matrix
Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) + _
```

```
                    (Bdl1_Times(i).Distance * Bdl1_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) + _
                    (Bdl1_Times(i).Distance * Bdl1_Trips(i).Trips(Mode, 2, j))

                'Sum of uncongested time Multiplied by Number of Trips ' Stored in the first row of the matrix
                'Peak
                Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) + _
                    (Bdl1_Times(i).UncongestedTime(j) * Bdl1_Trips(i).Trips(Mode, 1, j))
                'OffPeak
                Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) + _
                    (Bdl1_Times(i).UncongestedTime(j) * Bdl1_Trips(i).Trips(Mode, 1, j))


            Next Mode
            k = k + OffPeakIncrement
        Next j
    Next i

    Erase PasteTimes

End Sub

Sub ImportBdl2_Time()
    Dim c As Integer
    Dim txt As String
    Dim char As String * 1  'Read the line one Char at a time
    Dim data
    Dim i, Column As Integer
    Dim PasteTimes() As Double
    ReDim PasteTimes(1 To ODPairs, 1 To MaxCol)
    Dim MaxModes As Integer
    Dim k As Integer
    Dim OffPeakIncrement As Integer
    Dim FileName As String
    Dim Fy As Integer
    Dim FileNum As Integer

    FileNum = FreeFile
    Fy = Sheet27.Range("A18") + 1


    FileName = Sheet31.TextBox6.value

    MaxModes = Sheet9.Cells(12, 3)
    OffPeakIncrement = 24 - Sheet9.Cells(15, 15)

    Open FileName For Input As #FileNum

    'Read File
    r = 1
    c = 0
    txt = ""
    Do Until EOF(FileNum)
        Line Input #FileNum, data 'Read a line
        txt = ""
```

```
    For i = 1 To Len(data) + 1

        char = Mid(data, i, 1)
        If char = "," Or i = Len(data) + 1 Then 'comma
            'Store Data
            c = c + 1
            PasteTimes(r, c) = txt
            txt = ""
            Column = c
        ElseIf i = Len(data) Then 'End of File
            If char <> Chr(34) Then txt = txt & char

        ElseIf char <> Chr(34) Then
            txt = txt & char
        End If

    Next i
    c = 0
    r = r + 1
    Mode = 0
Loop
Close #FileNum

'ReDim Bdl1_Times(1 To r - 1)

For i = 1 To r - 1
    Bdl1_Times(i).Origin = PasteTimes(i, 1)
    Bdl1_Times(i).Destination = PasteTimes(i, 2)
    Bdl1_Times(i).Distance = PasteTimes(i, 3)
    k = 4
    For j = 1 To Fy
        Bdl1_Times(i).UncongestedTime(j) = PasteTimes(i, k)
        k = k + 1
        For Mode = 1 To MaxModes
            If Sheet9.Cells(14 + Mode, 12) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
            End If
            If Sheet9.Cells(14 + Mode, 13) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = PasteTimes(i, k)
```

```
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
    Else
        Bdl1_Times(i).AccessTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
    Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
End If
If Sheet9.Cells(14 + Mode, 14) = 1 Then
    If PasteTimes(i, k) >= 9999 Then
        Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
    Else
        Bdl1_Times(i).WaitTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
    Else
        Bdl1_Times(i).WaitTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
    Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
End If

'Sum of Trip time multiplied by number of trips 'Stored in the row if the matrix
Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).InVehTime(Mode, 1, j) * Bdl2_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).InVehTime(Mode, 2, j) * Bdl2_Trips(i).Trips(Mode, 2, j))

Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).AccessTime(Mode, 1, j) * Bdl2_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).AccessTime(Mode, 2, j) * Bdl2_Trips(i).Trips(Mode, 2, j))

Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).WaitTime(Mode, 1, j) * Bdl2_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).WaitTime(Mode, 2, j) * Bdl2_Trips(i).Trips(Mode, 2, j))

'Sum of distance Multiplied by Number of trips ' Stored in the first row of the matrix
Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).Distance * Bdl2_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).Distance * Bdl2_Trips(i).Trips(Mode, 2, j))

'Sum of uncongested time Multiplied by Number of Trips ' Stored in the first row of the matrix
'Peak
Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).UncongestedTime(j) * Bdl2_Trips(i).Trips(Mode, 1, j))
'OffPeak
```

```
                Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).UncongestedTime(j) * Bdl2_Trips(i).Trips(Mode, 1, j))


            Next Mode
            k = k + OffPeakIncrement
        Next j
    Next i

    Erase PasteTimes

End Sub

Sub ImportBdl3_Time()
    Dim c As Integer
    Dim txt As String
    Dim char As String * 1  'Read the line one Char at a time
    Dim data
    Dim i, Column As Integer
    Dim PasteTimes() As Double
    ReDim PasteTimes(1 To ODPairs, 1 To MaxCol)
    Dim MaxModes As Integer
    Dim k As Integer
    Dim OffPeakIncrement As Integer
    Dim FileName As String
    Dim Fy As Integer
    Dim FileNum As Integer

    FileNum = FreeFile
    Fy = Sheet27.Range("A18") + 1


    FileName = Sheet31.TextBox8.value

    MaxModes = Sheet9.Cells(12, 3)
    OffPeakIncrement = 24 - Sheet9.Cells(15, 15)

    Open FileName For Input As #FileNum

    'Read File
    r = 1
    c = 0
    txt = ""
    Do Until EOF(FileNum)
        Line Input #FileNum, data 'Read a line
        txt = ""
        For i = 1 To Len(data) + 1

            char = Mid(data, i, 1)
            If char = "," Or i = Len(data) + 1 Then 'comma
                'Store Data
                c = c + 1
                PasteTimes(r, c) = txt
                txt = ""
                Column = c
```

```
        ElseIf i = Len(data) Then 'End of File
            If char <> Chr(34) Then txt = txt & char

        ElseIf char <> Chr(34) Then
            txt = txt & char
        End If

    Next i
    c = 0
    r = r + 1
    Mode = 0
Loop
Close #FileNum

'ReDim Bdl1_Times(1 To r - 1)

For i = 1 To r - 1
    Bdl1_Times(i).Origin = PasteTimes(i, 1)
    Bdl1_Times(i).Destination = PasteTimes(i, 2)
    Bdl1_Times(i).Distance = PasteTimes(i, 3)
    k = 4
    For j = 1 To Fy
        Bdl1_Times(i).UncongestedTime(j) = PasteTimes(i, k)
        k = k + 1
        For Mode = 1 To MaxModes
            If Sheet9.Cells(14 + Mode, 12) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
            End If
            If Sheet9.Cells(14 + Mode, 13) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).AccessTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
```

```
                Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
            End If
            If Sheet9.Cells(14 + Mode, 14) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).WaitTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).WaitTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
                Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
            End If

            'Sum of Trip time multiplied by number of trips 'Stored in the row if the matrix
            Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).InVehTime(Mode, 1, j) * Bdl3_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).InVehTime(Mode, 2, j) * Bdl3_Trips(i).Trips(Mode, 2, j))

            Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).AccessTime(Mode, 1, j) * Bdl3_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).AccessTime(Mode, 2, j) * Bdl3_Trips(i).Trips(Mode, 2, j))

            Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).WaitTime(Mode, 1, j) * Bdl3_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).WaitTime(Mode, 2, j) * Bdl3_Trips(i).Trips(Mode, 2, j))

            'Sum of distance Multiplied by Number of trips ' Stored in the first row of the matrix
            Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).Distance * Bdl3_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).Distance * Bdl3_Trips(i).Trips(Mode, 2, j))

            'Sum of uncongested time Multiplied by Number of Trips ' Stored in the first row of the matrix
            'Peak
            Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).UncongestedTime(j) * Bdl3_Trips(i).Trips(Mode, 1, j))
            'OffPeak
            Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).UncongestedTime(j) * Bdl3_Trips(i).Trips(Mode, 1, j))


        Next Mode
        k = k + OffPeakIncrement
    Next j
Next i
```

```
    Erase PasteTimes

End Sub

Sub ImportBdl4_Time()
    Dim c As Integer
    Dim txt As String
    Dim char As String * 1  'Read the line one Char at a time
    Dim data
    Dim i, Column As Integer
    Dim PasteTimes() As Double
    ReDim PasteTimes(1 To ODPairs, 1 To MaxCol)
    Dim MaxModes As Integer
    Dim k As Integer
    Dim OffPeakIncrement As Integer
    Dim FileName As String
    Dim Fy As Integer
    Dim FileNum As Integer

    FileNum = FreeFile
    Fy = Sheet27.Range("A18") + 1


    FileName = Sheet31.TextBox10.value

    MaxModes = Sheet9.Cells(12, 3)
    OffPeakIncrement = 24 - Sheet9.Cells(15, 15)

    Open FileName For Input As #FileNum

    'Read File
    r = 1
    c = 0
    txt = ""
    Do Until EOF(FileNum)
        Line Input #FileNum, data 'Read a line
        txt = ""
        For i = 1 To Len(data) + 1

            char = Mid(data, i, 1)
            If char = "," Or i = Len(data) + 1 Then 'comma
                'Store Data
                c = c + 1
                PasteTimes(r, c) = txt
                txt = ""
                Column = c
            ElseIf i = Len(data) Then 'End of File
                If char <> Chr(34) Then txt = txt & char

            ElseIf char <> Chr(34) Then
                txt = txt & char
            End If

        Next i
        c = 0
```

```
    r = r + 1
    Mode = 0
Loop
Close #FileNum

'ReDim Bdl1_Times(1 To r - 1)

For i = 1 To r - 1
    Bdl1_Times(i).Origin = PasteTimes(i, 1)
    Bdl1_Times(i).Destination = PasteTimes(i, 2)
    Bdl1_Times(i).Distance = PasteTimes(i, 3)
    k = 4
    For j = 1 To Fy
        Bdl1_Times(i).UncongestedTime(j) = PasteTimes(i, k)
        k = k + 1
        For Mode = 1 To MaxModes
            If Sheet9.Cells(14 + Mode, 12) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
            End If
            If Sheet9.Cells(14 + Mode, 13) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).AccessTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
                Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
            End If
            If Sheet9.Cells(14 + Mode, 14) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).WaitTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
```

```
                    Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).WaitTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
                Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
            End If

            'Sum of Trip time multiplied by number of trips 'Stored in the row if the matrix
            Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).InVehTime(Mode, 1, j) * Bdl4_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).InVehTime(Mode, 2, j) * Bdl4_Trips(i).Trips(Mode, 2, j))

            Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).AccessTime(Mode, 1, j) * Bdl4_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).AccessTime(Mode, 2, j) * Bdl4_Trips(i).Trips(Mode, 2, j))

            Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).WaitTime(Mode, 1, j) * Bdl4_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).WaitTime(Mode, 2, j) * Bdl4_Trips(i).Trips(Mode, 2, j))

            'Sum of distance Multiplied by Number of trips ' Stored in the first row of the matrix
            Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).Distance * Bdl4_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).Distance * Bdl4_Trips(i).Trips(Mode, 2, j))

            'Sum of uncongested time Multiplied by Number of Trips ' Stored in the first row of the matrix
            'Peak
            Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).UncongestedTime(j) * Bdl4_Trips(i).Trips(Mode, 1, j))
            'OffPeak
            Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).UncongestedTime(j) * Bdl4_Trips(i).Trips(Mode, 1, j))


        Next Mode
        k = k + OffPeakIncrement
    Next j
Next i

Erase PasteTimes

End Sub

Sub ImportBdl5_Time()
    Dim c As Integer
    Dim txt As String
    Dim char As String * 1  'Read the line one Char at a time
    Dim data
```

```
Dim i, Column As Integer
Dim PasteTimes() As Double
ReDim PasteTimes(1 To ODPairs, 1 To MaxCol)
Dim MaxModes As Integer
Dim k As Integer
Dim OffPeakIncrement As Integer
Dim FileName As String
Dim Fy As Integer
Dim FileNum As Integer

FileNum = FreeFile
Fy = Sheet27.Range("A18") + 1


FileName = Sheet31.TextBox12.value

MaxModes = Sheet9.Cells(12, 3)
OffPeakIncrement = 24 - Sheet9.Cells(15, 15)

Open FileName For Input As #FileNum

'Read File
r = 1
c = 0
txt = ""
Do Until EOF(FileNum)
    Line Input #FileNum, data 'Read a line
    txt = ""
    For i = 1 To Len(data) + 1

        char = Mid(data, i, 1)
        If char = "," Or i = Len(data) + 1 Then 'comma
            'Store Data
            c = c + 1
            PasteTimes(r, c) = txt
            txt = ""
            Column = c
        ElseIf i = Len(data) Then 'End of File
            If char <> Chr(34) Then txt = txt & char

        ElseIf char <> Chr(34) Then
            txt = txt & char
        End If

    Next i
    c = 0
    r = r + 1
    Mode = 0
Loop
Close #FileNum

'ReDim Bdl1_Times(1 To r - 1)

For i = 1 To r - 1
    Bdl1_Times(i).Origin = PasteTimes(i, 1)
```

```
Bdl1_Times(i).Destination = PasteTimes(i, 2)
Bdl1_Times(i).Distance = PasteTimes(i, 3)
k = 4
For j = 1 To Fy
    Bdl1_Times(i).UncongestedTime(j) = PasteTimes(i, k)
    k = k + 1
    For Mode = 1 To MaxModes
        If Sheet9.Cells(14 + Mode, 12) = 1 Then
            If PasteTimes(i, k) >= 9999 Then
                Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
            Else
                Bdl1_Times(i).InVehTime(Mode, 1, j) = PasteTimes(i, k)
            End If
            If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
            Else
                Bdl1_Times(i).InVehTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
            End If
            k = k + 1
        Else
            Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
            Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
        End If
        If Sheet9.Cells(14 + Mode, 13) = 1 Then
            If PasteTimes(i, k) >= 9999 Then
                Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
            Else
                Bdl1_Times(i).AccessTime(Mode, 1, j) = PasteTimes(i, k)
            End If
            If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
            Else
                Bdl1_Times(i).AccessTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
            End If
            k = k + 1
        Else
            Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
            Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
        End If
        If Sheet9.Cells(14 + Mode, 14) = 1 Then
            If PasteTimes(i, k) >= 9999 Then
                Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
            Else
                Bdl1_Times(i).WaitTime(Mode, 1, j) = PasteTimes(i, k)
            End If
            If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
            Else
                Bdl1_Times(i).WaitTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
            End If
            k = k + 1
        Else
            Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
            Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
        End If
```

```
                'Sum of Trip time multiplied by number of trips 'Stored in the row if the matrix
                Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).InVehTime(Mode, 1, j) * Bdl5_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).InVehTime(Mode, 2, j) * Bdl5_Trips(i).Trips(Mode, 2, j))

                Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).AccessTime(Mode, 1, j) * Bdl5_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).AccessTime(Mode, 2, j) * Bdl5_Trips(i).Trips(Mode, 2, j))

                Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).WaitTime(Mode, 1, j) * Bdl5_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).WaitTime(Mode, 2, j) * Bdl5_Trips(i).Trips(Mode, 2, j))

                'Sum of distance Multiplied by Number of trips ' Stored in the first row of the matrix
                Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).Distance * Bdl5_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).Distance * Bdl5_Trips(i).Trips(Mode, 2, j))

                'Sum of uncongested time Multiplied by Number of Trips ' Stored in the first row of the matrix
                'Peak
                Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).UncongestedTime(j) * Bdl5_Trips(i).Trips(Mode, 1, j))
                'OffPeak
                Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).UncongestedTime(j) * Bdl5_Trips(i).Trips(Mode, 1, j))


            Next Mode
            k = k + OffPeakIncrement
        Next j
    Next i

    Erase PasteTimes

End Sub

Sub ImportBdl6_Time()
    Dim c As Integer
    Dim txt As String
    Dim char As String * 1  'Read the line one Char at a time
    Dim data
    Dim i, Column As Integer
    Dim PasteTimes() As Double
    ReDim PasteTimes(1 To ODPairs, 1 To MaxCol)
    Dim MaxModes As Integer
    Dim k As Integer
    Dim OffPeakIncrement As Integer
    Dim FileName As String
    Dim Fy As Integer
    Dim FileNum As Integer
```

```
FileNum = FreeFile
Fy = Sheet27.Range("A18") + 1


FileName = Sheet31.TextBox14.value

MaxModes = Sheet9.Cells(12, 3)
OffPeakIncrement = 24 - Sheet9.Cells(15, 15)

Open FileName For Input As #FileNum

'Read File
r = 1
c = 0
txt = ""
Do Until EOF(FileNum)
    Line Input #FileNum, data 'Read a line
    txt = ""
    For i = 1 To Len(data) + 1

        char = Mid(data, i, 1)
        If char = "," Or i = Len(data) + 1 Then 'comma
            'Store Data
            c = c + 1
            PasteTimes(r, c) = txt
            txt = ""
            Column = c
        ElseIf i = Len(data) Then 'End of File
            If char <> Chr(34) Then txt = txt & char

        ElseIf char <> Chr(34) Then
            txt = txt & char
        End If

    Next i
    c = 0
    r = r + 1
    Mode = 0
Loop
Close #FileNum


For i = 1 To r - 1
    Bdl1_Times(i).Origin = PasteTimes(i, 1)
    Bdl1_Times(i).Destination = PasteTimes(i, 2)
    Bdl1_Times(i).Distance = PasteTimes(i, 3)
    k = 4
    For j = 1 To Fy
        Bdl1_Times(i).UncongestedTime(j) = PasteTimes(i, k)
        k = k + 1
        For Mode = 1 To MaxModes
            If Sheet9.Cells(14 + Mode, 12) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
```

```
    Else
        Bdl1_Times(i).InVehTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
    Else
        Bdl1_Times(i).InVehTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
    Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
End If
If Sheet9.Cells(14 + Mode, 13) = 1 Then
    If PasteTimes(i, k) >= 9999 Then
        Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
    Else
        Bdl1_Times(i).AccessTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
    Else
        Bdl1_Times(i).AccessTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
    Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
End If
If Sheet9.Cells(14 + Mode, 14) = 1 Then
    If PasteTimes(i, k) >= 9999 Then
        Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
    Else
        Bdl1_Times(i).WaitTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
    Else
        Bdl1_Times(i).WaitTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
    Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
End If

'Sum of Trip time multiplied by number of trips 'Stored in the row if the matrix
Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).InVehTime(Mode, 1, j) * Bdl6_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).InVehTime(Mode, 2, j) * Bdl6_Trips(i).Trips(Mode, 2, j))

Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).AccessTime(Mode, 1, j) * Bdl6_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) + _
```

```
                            (Bdl1_Times(i).AccessTime(Mode, 2, j) * Bdl6_Trips(i).Trips(Mode, 2, j))

                Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).WaitTime(Mode, 1, j) * Bdl6_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).WaitTime(Mode, 2, j) * Bdl6_Trips(i).Trips(Mode, 2, j))

                'Sum of distance Multiplied by Number of trips ' Stored in the first row of the matrix
                Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).Distance * Bdl6_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).Distance * Bdl6_Trips(i).Trips(Mode, 2, j))

                'Sum of uncongested time Multiplied by Number of Trips ' Stored in the first row of the matrix
                'Peak
                Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).UncongestedTime(j) * Bdl6_Trips(i).Trips(Mode, 1, j))
                'OffPeak
                Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).UncongestedTime(j) * Bdl6_Trips(i).Trips(Mode, 1, j))


            Next Mode
            k = k + OffPeakIncrement
        Next j
    Next i

    Erase PasteTimes

End Sub


Sub ImportBdl7_Time()
    Dim c As Integer
    Dim txt As String
    Dim char As String * 1  'Read the line one Char at a time
    Dim data
    Dim i, Column As Integer
    Dim PasteTimes() As Double
    ReDim PasteTimes(1 To ODPairs, 1 To MaxCol)
    Dim MaxModes As Integer
    Dim k As Integer
    Dim OffPeakIncrement As Integer
    Dim FileName As String
    Dim Fy As Integer
    Dim FileNum As Integer

    FileNum = FreeFile
    Fy = Sheet27.Range("A18") + 1


    FileName = Sheet31.TextBox16.value

    MaxModes = Sheet9.Cells(12, 3)
    OffPeakIncrement = 24 - Sheet9.Cells(15, 15)
```

```
Open FileName For Input As #FileNum

'Read File
r = 1
c = 0
txt = ""
Do Until EOF(FileNum)
    Line Input #FileNum, data 'Read a line
    txt = ""
    For i = 1 To Len(data) + 1

        char = Mid(data, i, 1)
        If char = "," Or i = Len(data) + 1 Then 'comma
            'Store Data
            c = c + 1
            PasteTimes(r, c) = txt
            txt = ""
            Column = c
        ElseIf i = Len(data) Then 'End of File
            If char <> Chr(34) Then txt = txt & char

        ElseIf char <> Chr(34) Then
            txt = txt & char
        End If

    Next i
    c = 0
    r = r + 1
    Mode = 0
Loop
Close #FileNum


For i = 1 To r - 1
    Bdl1_Times(i).Origin = PasteTimes(i, 1)
    Bdl1_Times(i).Destination = PasteTimes(i, 2)
    Bdl1_Times(i).Distance = PasteTimes(i, 3)
    k = 4
    For j = 1 To Fy
        Bdl1_Times(i).UncongestedTime(j) = PasteTimes(i, k)
        k = k + 1
        For Mode = 1 To MaxModes
            If Sheet9.Cells(14 + Mode, 12) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
```

```
Else
    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
    Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
End If
If Sheet9.Cells(14 + Mode, 13) = 1 Then
    If PasteTimes(i, k) >= 9999 Then
        Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
    Else
        Bdl1_Times(i).AccessTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
    Else
        Bdl1_Times(i).AccessTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
    Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
End If
If Sheet9.Cells(14 + Mode, 14) = 1 Then
    If PasteTimes(i, k) >= 9999 Then
        Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
    Else
        Bdl1_Times(i).WaitTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
    Else
        Bdl1_Times(i).WaitTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
    Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
End If

'Sum of Trip time multiplied by number of trips 'Stored in the row if the matrix
Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).InVehTime(Mode, 1, j) * Bdl7_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).InVehTime(Mode, 2, j) * Bdl7_Trips(i).Trips(Mode, 2, j))

Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).AccessTime(Mode, 1, j) * Bdl7_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).AccessTime(Mode, 2, j) * Bdl7_Trips(i).Trips(Mode, 2, j))

Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).WaitTime(Mode, 1, j) * Bdl7_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).WaitTime(Mode, 2, j) * Bdl7_Trips(i).Trips(Mode, 2, j))

'Sum of distance Multiplied by Number of trips ' Stored in the first row of the matrix
Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) + _
```

```
                    (Bdl1_Times(i).Distance * Bdl7_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) + _
                    (Bdl1_Times(i).Distance * Bdl7_Trips(i).Trips(Mode, 2, j))

                'Sum of uncongested time Multiplied by Number of Trips ' Stored in the first row of the matrix
                'Peak
                Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) + _
                    (Bdl1_Times(i).UncongestedTime(j) * Bdl7_Trips(i).Trips(Mode, 1, j))
                'OffPeak
                Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) + _
                    (Bdl1_Times(i).UncongestedTime(j) * Bdl7_Trips(i).Trips(Mode, 1, j))


            Next Mode
            k = k + OffPeakIncrement
        Next j
    Next i

    Erase PasteTimes

End Sub

Sub Importbdl8_Time()
    Dim c As Integer
    Dim txt As String
    Dim char As String * 1  'Read the line one Char at a time
    Dim data
    Dim i, Column As Integer
    Dim PasteTimes() As Double
    ReDim PasteTimes(1 To ODPairs, 1 To MaxCol)
    Dim MaxModes As Integer
    Dim k As Integer
    Dim OffPeakIncrement As Integer
    Dim FileName As String
    Dim Fy As Integer
    Dim FileNum As Integer

    FileNum = FreeFile
    Fy = Sheet27.Range("A18") + 1


    FileName = Sheet31.TextBox18.value

    MaxModes = Sheet9.Cells(12, 3)
    OffPeakIncrement = 24 - Sheet9.Cells(15, 15)

    Open FileName For Input As #FileNum

    'Read File
    r = 1
    c = 0
    txt = ""
    Do Until EOF(FileNum)
        Line Input #FileNum, data 'Read a line
        txt = ""
```

```
    For i = 1 To Len(data) + 1

        char = Mid(data, i, 1)
        If char = "," Or i = Len(data) + 1 Then 'comma
            'Store Data
            c = c + 1
            PasteTimes(r, c) = txt
            txt = ""
            Column = c
        ElseIf i = Len(data) Then 'End of File
            If char <> Chr(34) Then txt = txt & char

        ElseIf char <> Chr(34) Then
            txt = txt & char
        End If

    Next i
    c = 0
    r = r + 1
    Mode = 0
Loop
Close #FileNum

For i = 1 To r - 1
    Bdl1_Times(i).Origin = PasteTimes(i, 1)
    Bdl1_Times(i).Destination = PasteTimes(i, 2)
    Bdl1_Times(i).Distance = PasteTimes(i, 3)
    k = 4
    For j = 1 To Fy
        Bdl1_Times(i).UncongestedTime(j) = PasteTimes(i, k)
        k = k + 1
        For Mode = 1 To MaxModes
            If Sheet9.Cells(14 + Mode, 12) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
            End If
            If Sheet9.Cells(14 + Mode, 13) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
```

```
        Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
    Else
        Bdl1_Times(i).AccessTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
    Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
End If
If Sheet9.Cells(14 + Mode, 14) = 1 Then
    If PasteTimes(i, k) >= 9999 Then
        Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
    Else
        Bdl1_Times(i).WaitTime(Mode, 1, j) = PasteTimes(i, k)
    End If
    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
        Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
    Else
        Bdl1_Times(i).WaitTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
    End If
    k = k + 1
Else
    Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
    Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
End If

'Sum of Trip time multiplied by number of trips 'Stored in the row if the matrix
Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).InVehTime(Mode, 1, j) * Bdl8_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).InVehTime(Mode, 2, j) * Bdl8_Trips(i).Trips(Mode, 2, j))

Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).AccessTime(Mode, 1, j) * Bdl8_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).AccessTime(Mode, 2, j) * Bdl8_Trips(i).Trips(Mode, 2, j))

Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).WaitTime(Mode, 1, j) * Bdl8_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).WaitTime(Mode, 2, j) * Bdl8_Trips(i).Trips(Mode, 2, j))

'Sum of distance Multiplied by Number of trips ' Stored in the first row of the matrix
Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).Distance * Bdl8_Trips(i).Trips(Mode, 1, j))
Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).Distance * Bdl8_Trips(i).Trips(Mode, 2, j))

'Sum of uncongested time Multiplied by Number of Trips ' Stored in the first row of the matrix
'Peak
Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) + _
            (Bdl1_Times(i).UncongestedTime(j) * Bdl8_Trips(i).Trips(Mode, 1, j))
'OffPeak
Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) + _
            (Bdl1_Times(i).UncongestedTime(j) * Bdl8_Trips(i).Trips(Mode, 1, j))
```

```
            Next Mode
            k = k + OffPeakIncrement
        Next j
    Next i

    Erase PasteTimes

End Sub

Sub Importbdl9_Time()
    Dim c As Integer
    Dim txt As String
    Dim char As String * 1  'Read the line one Char at a time
    Dim data
    Dim i, Column As Integer
    Dim PasteTimes() As Double
    ReDim PasteTimes(1 To ODPairs, 1 To MaxCol)
    Dim MaxModes As Integer
    Dim k As Integer
    Dim OffPeakIncrement As Integer
    Dim FileName As String
    Dim Fy As Integer
    Dim FileNum As Integer

    FileNum = FreeFile
    Fy = Sheet27.Range("A18") + 1


    FileName = Sheet31.TextBox20.value

    MaxModes = Sheet9.Cells(12, 3)
    OffPeakIncrement = 24 - Sheet9.Cells(15, 15)

    Open FileName For Input As #FileNum

    'Read File
    r = 1
    c = 0
    txt = ""
    Do Until EOF(FileNum)
        Line Input #FileNum, data 'Read a line
        txt = ""
        For i = 1 To Len(data) + 1

            char = Mid(data, i, 1)
            If char = "," Or i = Len(data) + 1 Then 'comma
                'Store Data
                c = c + 1
                PasteTimes(r, c) = txt
                txt = ""
                Column = c
            ElseIf i = Len(data) Then 'End of File
                If char <> Chr(34) Then txt = txt & char
```

```
        ElseIf char <> Chr(34) Then
            txt = txt & char
        End If

    Next i
    c = 0
    r = r + 1
    Mode = 0
Loop
Close #FileNum


For i = 1 To r - 1
    Bdl1_Times(i).Origin = PasteTimes(i, 1)
    Bdl1_Times(i).Destination = PasteTimes(i, 2)
    Bdl1_Times(i).Distance = PasteTimes(i, 3)
    k = 4
    For j = 1 To Fy
        Bdl1_Times(i).UncongestedTime(j) = PasteTimes(i, k)
        k = k + 1
        For Mode = 1 To MaxModes
            If Sheet9.Cells(14 + Mode, 12) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
            End If
            If Sheet9.Cells(14 + Mode, 13) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).AccessTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
                Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
            End If
            If Sheet9.Cells(14 + Mode, 14) = 1 Then
```

```
                    If PasteTimes(i, k) >= 9999 Then
                        Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
                    Else
                        Bdl1_Times(i).WaitTime(Mode, 1, j) = PasteTimes(i, k)
                    End If
                    If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                        Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
                    Else
                        Bdl1_Times(i).WaitTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                    End If
                    k = k + 1
                Else
                    Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
                    Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
                End If

                'Sum of Trip time multiplied by number of trips 'Stored in the row if the matrix
                Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).InVehTime(Mode, 1, j) * Bdl9_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).InVehTime(Mode, 2, j) * Bdl9_Trips(i).Trips(Mode, 2, j))

                Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).AccessTime(Mode, 1, j) * Bdl9_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).AccessTime(Mode, 2, j) * Bdl9_Trips(i).Trips(Mode, 2, j))

                Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).WaitTime(Mode, 1, j) * Bdl9_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).WaitTime(Mode, 2, j) * Bdl9_Trips(i).Trips(Mode, 2, j))

                'Sum of distance Multiplied by Number of trips ' Stored in the first row of the matrix
                Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).Distance * Bdl9_Trips(i).Trips(Mode, 1, j))
                Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).Distance * Bdl9_Trips(i).Trips(Mode, 2, j))

                'Sum of uncongested time Multiplied by Number of Trips ' Stored in the first row of the matrix
                'Peak
                Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) + _
                            (Bdl1_Times(i).UncongestedTime(j) * Bdl9_Trips(i).Trips(Mode, 1, j))
                'OffPeak
                Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) + _
                            (Bdl1_Times(i).UncongestedTime(j) * Bdl9_Trips(i).Trips(Mode, 1, j))


            Next Mode
            k = k + OffPeakIncrement
        Next j
    Next i

    Erase PasteTimes

End Sub
```

```
Sub Importbdl10_Time()
    Dim c As Integer
    Dim txt As String
    Dim char As String * 1  'Read the line one Char at a time
    Dim data
    Dim i, Column As Integer
    Dim PasteTimes() As Double
    ReDim PasteTimes(1 To ODPairs, 1 To MaxCol)
    Dim MaxModes As Integer
    Dim k As Integer
    Dim OffPeakIncrement As Integer
    Dim FileName As String
    Dim Fy As Integer
    Dim FileNum As Integer

    FileNum = FreeFile
    Fy = Sheet27.Range("A18") + 1


    FileName = Sheet31.TextBox22.value

    MaxModes = Sheet9.Cells(12, 3)
    OffPeakIncrement = 24 - Sheet9.Cells(15, 15)

    Open FileName For Input As #FileNum

    'Read File
    r = 1
    c = 0
    txt = ""
    Do Until EOF(FileNum)
        Line Input #FileNum, data 'Read a line
        txt = ""
        For i = 1 To Len(data) + 1

            char = Mid(data, i, 1)
            If char = "," Or i = Len(data) + 1 Then 'comma
                'Store Data
                c = c + 1
                PasteTimes(r, c) = txt
                txt = ""
                Column = c
            ElseIf i = Len(data) Then 'End of File
                If char <> Chr(34) Then txt = txt & char

            ElseIf char <> Chr(34) Then
                txt = txt & char
            End If

        Next i
        c = 0
        r = r + 1
        Mode = 0
    Loop
```

```
Close #FileNum


For i = 1 To r - 1
    Bdl1_Times(i).Origin = PasteTimes(i, 1)
    Bdl1_Times(i).Destination = PasteTimes(i, 2)
    Bdl1_Times(i).Distance = PasteTimes(i, 3)
    k = 4
    For j = 1 To Fy
        Bdl1_Times(i).UncongestedTime(j) = PasteTimes(i, k)
        k = k + 1
        For Mode = 1 To MaxModes
            If Sheet9.Cells(14 + Mode, 12) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).InVehTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).InVehTime(Mode, 1, j) = 0
                Bdl1_Times(i).InVehTime(Mode, 2, j) = 0
            End If
            If Sheet9.Cells(14 + Mode, 13) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).AccessTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).AccessTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
                k = k + 1
            Else
                Bdl1_Times(i).AccessTime(Mode, 1, j) = 0
                Bdl1_Times(i).AccessTime(Mode, 2, j) = 0
            End If
            If Sheet9.Cells(14 + Mode, 14) = 1 Then
                If PasteTimes(i, k) >= 9999 Then
                    Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
                Else
                    Bdl1_Times(i).WaitTime(Mode, 1, j) = PasteTimes(i, k)
                End If
                If PasteTimes(i, k + OffPeakIncrement) >= 9999 Then
                    Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
                Else
                    Bdl1_Times(i).WaitTime(Mode, 2, j) = PasteTimes(i, k + OffPeakIncrement)
                End If
```

```
                k = k + 1
            Else
                Bdl1_Times(i).WaitTime(Mode, 1, j) = 0
                Bdl1_Times(i).WaitTime(Mode, 2, j) = 0
            End If

            'Sum of Trip time multiplied by number of trips 'Stored in the row if the matrix
            Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).InVehTime(Mode, 1, j) * Bdl10_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) = Bdl1_Times(1).InVehTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).InVehTime(Mode, 2, j) * Bdl10_Trips(i).Trips(Mode, 2, j))

            Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).AccessTime(Mode, 1, j) * Bdl10_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) = Bdl1_Times(1).AccessTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).AccessTime(Mode, 2, j) * Bdl10_Trips(i).Trips(Mode, 2, j))

            Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).WaitTime(Mode, 1, j) * Bdl10_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) = Bdl1_Times(1).WaitTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).WaitTime(Mode, 2, j) * Bdl10_Trips(i).Trips(Mode, 2, j))

            'Sum of distance Multiplied by Number of trips ' Stored in the first row of the matrix
            Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).Distance * Bdl10_Trips(i).Trips(Mode, 1, j))
            Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) = Bdl1_Times(1).DistanceTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).Distance * Bdl10_Trips(i).Trips(Mode, 2, j))

            'Sum of uncongested time Multiplied by Number of Trips ' Stored in the first row of the matrix
            'Peak
            Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 1, j) + _
                        (Bdl1_Times(i).UncongestedTime(j) * Bdl10_Trips(i).Trips(Mode, 1, j))
            'OffPeak
            Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) = Bdl1_Times(1).UncongestedTimesTrips(Mode, 2, j) + _
                        (Bdl1_Times(i).UncongestedTime(j) * Bdl10_Trips(i).Trips(Mode, 1, j))


        Next Mode
        k = k + OffPeakIncrement
    Next j
Next i

Erase PasteTimes

End Sub

Public Function TotalTripsByMode(TripMtx() As Trips_Mtx2, ODs As Long, Years As Integer) As Double()
Dim Modes, Mj, Fy As Integer
Dim TempMtx() As Double
Modes = Sheet9.Cells(12, 3)
ReDim TempMtx(1 To Modes, 1 To 2, 1 To 4)

For i = 1 To ODs
    For Mj = 1 To Modes
        For Fy = 1 To Years
```

```
            'Peak
            TempMtx(Mj, 1, Fy) = TempMtx(Mj, 1, Fy) + TripMtx(i).Trips(Mj, 1, Fy)
            TempMtx(Mj, 2, Fy) = TempMtx(Mj, 2, Fy) + TripMtx(i).Trips(Mj, 2, Fy)

        Next Fy
    Next Mj
Next i
TotalTripsByMode = TempMtx()

End Function
Public Function AvgUncongestedTime(totaltrips() As Double, timeMtx() As Time_MTX2, Modes As Integer, _
                ForecastYears As Integer) As Double()
Dim Fy As Integer
Dim TempArr() As Double

ReDim TempArr(1 To 1, 1 To 2, 1 To ForecastYears)

For Fy = 1 To ForecastYears
    'Peak
    If totaltrips(1, 1, Fy) = 0 Then
        TempArr(1, 1, Fy) = 0
    Else
        TempArr(1, 1, Fy) = timeMtx(1).UncongestedTimesTrips(1, 1, Fy) / totaltrips(1, 1, Fy)
    End If
    'Off Peak
    If totaltrips(1, 2, Fy) = 0 Then
        TempArr(1, 2, Fy) = 0
    Else
        TempArr(1, 2, Fy) = timeMtx(1).UncongestedTimesTrips(1, 2, Fy) / totaltrips(1, 2, Fy)
    End If
Next Fy
AvgUncongestedTime = TempArr()

End Function

Public Function AvgTripTimeByMode(totaltrips() As Double, timeMtx() As Time_MTX2, Modes As Integer, ForecastYears As Integer) As Double()
Dim TempArr() As Double
ReDim TempArr(1 To Modes, 1 To 3, 1 To 2, 1 To ForecastYears)

For Fy = 1 To ForecastYears
    For Mj = 1 To Modes
        'Peak
        If totaltrips(Mj, 1, Fy) = 0 Then
            'InVehicle
            TempArr(Mj, 1, 1, Fy) = 0
            'Access Time
            TempArr(Mj, 2, 1, Fy) = 0
            'Wait Time
            TempArr(Mj, 3, 1, Fy) = 0
        Else
            'InVehicle
            TempArr(Mj, 1, 1, Fy) = timeMtx(1).InVehTimesTrips(Mj, 1, Fy) / totaltrips(Mj, 1, Fy)
        'Access Time
            TempArr(Mj, 2, 1, Fy) = timeMtx(1).AccessTimesTrips(Mj, 1, Fy) / totaltrips(Mj, 1, Fy)
        'Wait Time
```

```
            TempArr(Mj, 3, 1, Fy) = timeMtx(1).WaitTimesTrips(Mj, 1, Fy) / totaltrips(Mj, 1, Fy)
         End If
         'Off Peak
         If totaltrips(Mj, 2, Fy) = 0 Then
            'InVehicle
            TempArr(Mj, 1, 2, Fy) = 0
            'Access Time
            TempArr(Mj, 2, 2, Fy) = 0
            'Wait Time
            TempArr(Mj, 3, 2, Fy) = 0
        Else
            'InVehicle
            TempArr(Mj, 1, 2, Fy) = timeMtx(1).InVehTimesTrips(Mj, 2, Fy) / totaltrips(Mj, 2, Fy)
        'Access Time
            TempArr(Mj, 2, 2, Fy) = timeMtx(1).AccessTimesTrips(Mj, 2, Fy) / totaltrips(Mj, 2, Fy)
        'Wait Time
            TempArr(Mj, 3, 2, Fy) = timeMtx(1).WaitTimesTrips(Mj, 2, Fy) / totaltrips(Mj, 2, Fy)
         End If
    Next Mj
Next Fy

AvgTripTimeByMode = TempArr()

End Function

Public Function AvgDistanceTraveledByMode(totaltrips() As Double, timeMtx() As Time_MTX2, Modes As Integer, ForecastYears As Integer) As
Double()
Dim TempArr() As Double
ReDim TempArr(1 To Modes, 1 To 2, 1 To ForecastYears)

For Fy = 1 To ForecastYears
    For Mj = 1 To Modes
        'Peak
        If totaltrips(Mj, 1, Fy) = 0 Then
            TempArr(Mj, 1, Fy) = 0
        Else
            TempArr(Mj, 1, Fy) = timeMtx(1).DistanceTimesTrips(Mj, 1, Fy) / totaltrips(Mj, 1, Fy)
        End If

        'Off Peak
        If totaltrips(Mj, 2, Fy) = 0 Then
            TempArr(Mj, 2, Fy) = 0
        Else
            TempArr(Mj, 2, Fy) = timeMtx(1).DistanceTimesTrips(Mj, 2, Fy) / totaltrips(Mj, 2, Fy)
        End If
    Next Mj
Next Fy

AvgDistanceTraveledByMode = TempArr()

End Function

Public Function TimeBenefits(Base_Trips() As Trips_Mtx2, Bundle_Trips() As Trips_Mtx2, Base_Time() As Time_MTX2, _
                             Bundle_Time() As Time_MTX2, ODPair As Long, Modes As Integer, ForecastYears As Integer) As Double()
```

```
Dim TempArr() As Double
ReDim TempArr(1 To Modes, 1 To 3, 1 To 2, 1 To ForecastYears)
Dim ExistingTrips, NewTrips As Double

For i = 1 To ODPair
    For Mj = 1 To Modes
        For Fy = 1 To ForecastYears
            For Period = 1 To 2
        'Existing and New Trips
                If Base_Trips(i).Trips(Mj, Period, Fy + 1) < Bundle_Trips(i).Trips(Mj, Period, Fy) Then
                    'TimeBenefits
                    'In Vehicle Time
                    TempArr(Mj, 1, Period, Fy) = TempArr(Mj, 1, Period, Fy) + _
                        (Base_Time(i).InVehTime(Mj, Period, Fy + 1) - Bundle_Time(i).InVehTime(Mj, Period, Fy)) * _
                        (Base_Trips(i).Trips(Mj, Period, Fy + 1) + Bundle_Trips(i).Trips(Mj, Period, Fy)) * 0.5
                    'Access Time
                    TempArr(Mj, 2, Period, Fy) = TempArr(Mj, 2, Period, Fy) + _
                        (Base_Time(i).AccessTime(Mj, Period, Fy + 1) - Bundle_Time(i).AccessTime(Mj, Period, Fy)) * _
                        (Base_Trips(i).Trips(Mj, Period, Fy + 1) + Bundle_Trips(i).Trips(Mj, Period, Fy)) * 0.5
                    'Wait Time
                    TempArr(Mj, 3, Period, Fy) = TempArr(Mj, 3, Period, Fy) + _
                        (Base_Time(i).WaitTime(Mj, Period, Fy + 1) - Bundle_Time(i).WaitTime(Mj, Period, Fy)) * _
                        (Base_Trips(i).Trips(Mj, Period, Fy + 1) + Bundle_Trips(i).Trips(Mj, Period, Fy)) * 0.5
                Else
                    'In Vehicle Time
                    TempArr(Mj, 1, Period, Fy) = TempArr(Mj, 1, Period, Fy) + _
                    (Base_Time(i).InVehTime(Mj, Period, Fy + 1) - Bundle_Time(i).InVehTime(Mj, Period, Fy)) * _
                    Bundle_Trips(i).Trips(Mj, Period, Fy)
                    'Access Time
                    TempArr(Mj, 2, Period, Fy) = TempArr(Mj, 2, Period, Fy) + _
                    (Base_Time(i).AccessTime(Mj, Period, Fy + 1) - Bundle_Time(i).AccessTime(Mj, Period, Fy)) * _
                    Bundle_Trips(i).Trips(Mj, Period, Fy)
                    'Wait Time
                     TempArr(Mj, 3, Period, Fy) = TempArr(Mj, 3, Period, Fy) + _
                    (Base_Time(i).WaitTime(Mj, Period, Fy + 1) - Bundle_Time(i).WaitTime(Mj, Period, Fy)) * _
                    Bundle_Trips(i).Trips(Mj, Period, Fy)
                End If
            Next Period
        Next Fy
    Next Mj
Next i

TimeBenefits = TempArr()

End Function
```

Module Lock_Hide

```
Sub MOSAIC_ProtectModel()
    Dim Sh As Worksheet

    For Each Sh In Worksheets
        Sh.Protect UserInterfaceOnly:=True
    Next Sh
```

```
    Sheet2.Range("F21") = "Locked for Editing"
    Sheet34.Range("D3") = "Locked for Editing"

    Range("A1").Select
End Sub

Sub MOSAIC_UnprotectModel()
    Dim Sh As Worksheet
    For Each Sh In Worksheets
        Sh.Unprotect
    Next Sh
    Sheet2.Range("F21") = "Fully Unlocked"
    Sheet34.Range("D3") = "Fully Unlocked"
    Range("A1").Select
End Sub

Sub Mosaic_Lock_Unlock()
    Application.ScreenUpdating = False
    If Sheet2.Range("F21") = "Locked for Editing" Then
        MOSAIC_UnprotectModel
    Else
        MOSAIC_ProtectModel
    End If
   Application.ScreenUpdating = True
End Sub

Sub UnhideSheets()
    Dim Sh As Worksheets
    Dim Name As String
    Dim i, MaxNumber As Integer
    Dim CurrentCondition As String
    Application.ScreenUpdating = False
    MaxNumber = 39
    CurrentCondition = Sheet2.Range("F21")
    'Umprotect Workbook
    MOSAIC_UnprotectModel
    For i = 1 To MaxNumber
        If Sheet34.Cells(4 + i, 4) = 1 Then
            Name = Sheet34.Cells(4 + i, 2)
            On Error Resume Next
            Worksheets(Name).Visible = True
            If Err Then
                Sheet34.Cells(4 + i, 2) = Worksheets(i).Name
                Name = Sheet34.Cells(4 + i, 2)
                Cells(4 + i, 2).Hyperlinks(1).SubAddress = "'" & Name & "'!A1"
                Cells(4 + i, 2).Hyperlinks(1).ScreenTip = "Go To " & Name & " worksheet."
                Worksheets(Name).Visible = True
            End If
            On Error GoTo 0
            Sheet34.Range(Cells(4 + i, 4), Cells(4 + i, 4)).EntireRow.Hidden = False
        End If
    Next i
    'Sheet"WEIGHT INDICATORS" Unhide columns J-Y
    Sheet26.Range("J1:Z1").EntireColumn.Hidden = False
```

```
    Sheet34.Range("D4") = "Advanced Use"
    Sheet2.Range("F22") = "Advanced Use"

'    'Protect or Unprotect
'    If CurrentCondition = "Fully Unlocked" Then
'        MOSAIC_UnprotectModel
'    Else
'        MOSAIC_ProtectModel
'    End If
    Sheet2.Select
    Sheet2.Range("A1").Select
    Application.ScreenUpdating = True

End Sub

Sub HideSheets()
    Dim Sh As Worksheets
    Dim Name As String
    Dim i, MaxNumber As Integer
    Dim CurrentCondition As String

    Application.ScreenUpdating = False
    MaxNumber = 39
    CurrentCondition = Sheet2.Range("F21")

    'Unprotect Workbook
    MOSAIC_UnprotectModel
    For i = 1 To MaxNumber
        If Sheet34.Cells(4 + i, 4) = 1 Then
            Name = Sheet34.Cells(4 + i, 2)
            On Error Resume Next
            Worksheets(Name).Visible = False
            If Err Then
                Sheet34.Cells(4 + i, 2) = Worksheets(i).Name
                Name = Sheet34.Cells(4 + i, 2)
                Cells(4 + i, 2).Hyperlinks(1).SubAddress = "'" & Name & "'!A1"
                Cells(4 + i, 2).Hyperlinks(1).ScreenTip = "Go To " & Name & " worksheet."
                Worksheets(Name).Visible = False
            End If
            On Error GoTo 0

            Sheet34.Range(Cells(4 + i, 4), Cells(4 + i, 4)).EntireRow.Hidden = True
        End If
    Next i

    'Hide columns J-Y in worksheet "WEIGHT INDICATORS"
    Sheet26.Range("J1:Z1").EntireColumn.Hidden = True

    Sheet34.Range("D4") = "Standard Use"
    Sheet2.Range("F22") = "Standard Use"
    Range("A1").Select

    'Protect or Unprotect
    If CurrentCondition = "Fully Unlocked" Then
        MOSAIC_UnprotectModel
```

```
    Else
        MOSAIC_ProtectModel
    End If

    Sheet2.Select
    Sheet2.Range("A1").Select

    Application.ScreenUpdating = True
End Sub

Sub MOSAIC_ADVANCED()
        Application.ScreenUpdating = False
       Sheet34.Select
        If Sheet34.Range("D4") = "Advanced Use" Then
            HideSheets
        Else
            UnhideSheets
        End If
        Range("A1").Select
        Application.ScreenUpdating = True
End Sub
```

Module MODASensitivity

```
'VARIABILITY PARAMETERS

Private UserInput As Double 'increment percent change

'ARRAYS
Private NumArray() As String 'array with all the numeric data
Private NameArray() As String 'array with Variable Names

Private Const maxParamNum = 7 'maximum number of parameters in the numeric array
Private Const maxParamName = 9 'maximum number of parameters in the string array

'WORKSHEETS
Private shWeight As Worksheet 'Weight Indicators
Private shSensitivity As Worksheet 'Sensitivity worksheet
Private shModaMatrix As Worksheet 'ModaMatrix worksheet

'COLUMN CONSTANTS
Private Const colW_UseInMosaic = 7 'column G in WEIGHT INDICATORS worksheet where "UseInMosaic" valuse are. Used to see if there is a value in
that row.
Private Const colW_SpecificInd = 6 'column F in WEIGHT INDICATORS worksheet where "General Indicators" valuse are
Private Const colW_Weight = 8 'column H in WEIGTH INDICATORS worksheet that we need to be changing
Private Const colW_CheckScores = 14 'CHECK SCORES column

'RANGES
Private rngOutput As Range 'output range

'OTHER
Private Const NumberOfBundles = 10 'Number of bundles

Public Sub Main_ModaSensitivity()
    Dim CurrentStatus As String
```

```
    CurrentStatus = Sheet2.Range("F21")
    Application.ScreenUpdating = False
    'Unprotect Model
    MOSAIC_UnprotectModel

    Call AssignInputValues
    Call WeightTable
    Call CollectData
    Application.DisplayAlerts = False
    shModaMatrix.Delete
    Application.DisplayAlerts = True

    Call PutDataToWorksheet
    Erase NumArray
    Erase NameArray
    Calculate
    If CurrentStatus = "Locked for Editing" Then
        'Lock Model
        Sheet2.Range("F21") = "locked for Editing"
        MOSAIC_ProtectModel

    End If

    Application.ScreenUpdating = True
End Sub

Private Sub ClearWorksheet()
    shSensitivity.Select
    shSensitivity.Columns("FA:GX").Clear '*** change this to a calculation
End Sub

Private Sub AssignInputValues()
    ReDim Preserve NumArray(1 To maxParamNum, 0 To 0)
    ReDim Preserve NameArray(1 To maxParamName, 0 To 0)

    Set shSensitivity = Sheet41
    Set shWeight = Sheet26 'weight indicators

    UserInput = shSensitivity.Range("sensitivityUserInput")
End Sub

Private Sub WeightTable()

    myCalculation = Application.Calculation
    Application.Calculation = xlCalculationManual
    Set shModaMatrix = ThisWorkbook.Worksheets.Add
    'shModaMatrix.Cells.Clear
    Set rngOutput = shWeight.Range("AggScore")

    Dim iRow As Long 'current row (in the loop)
    Const FirstRow = 6 'row where variables start in MODEL PARAMETERS
    Dim LastRow 'last row in MODEL PARAMETERS
    LastRow = shWeight.UsedRange.Row + shWeight.UsedRange.Rows.Count - 1

    Dim OriginalValue As Double 'original value of the parameter (from MODEL PARAMETERS)
```

```
    Dim MinValue As Double 'Minimum value of the parameter, based on the variability defined in AssignInputValues Sub
    Dim MaxValue As Double 'Maximum value of the parameter, based on the variability defined in AssignInputValues Sub
    Dim Increment As Double 'Increment value (not in percentage terms)
    Dim iValue As Double 'Current Value in percentage terms (in the loop)
    Dim iValueRound As Double 'Current Value in percentage terms, rounded to 5th decimal
    Dim VariableIndex As Integer 'Variable Index (order in the Model Parameters worksheet)
    Dim VariableName As String 'Variable Name

        'put original values
    Range(shModaMatrix.Cells(FirstRow, 9), shModaMatrix.Cells(LastRow, 9)).Formula = "=ROW()"
    Range(shModaMatrix.Cells(FirstRow, 10), shModaMatrix.Cells(LastRow, 10)).value = Range(shWeight.Cells(FirstRow, colW_Weight),
shWeight.Cells(LastRow, colW_Weight)).value
    'Range(shModaMatrix.Cells(FirstRow, 11), shModaMatrix.Cells(LastRow, 11)).value = "=VLOOKUP(RC[-1],"

    SumWeights = Range("SumWeights")

    myRow = 0
    For iRow = FirstRow To LastRow
        If shWeight.Cells(iRow, colW_SpecificInd) <> "" Then 'only if there is a specific indicator value in this row

            If IsNumeric(shWeight.Cells(iRow, colW_Weight)) Then
                OriginalValue = shWeight.Cells(iRow, colW_Weight) 'keep the original value (for calculations below and to put it back when the
loop is over)
            Else
                Exit For
            End If

            VariableIndex = VariableIndex + 1
            ReDim Preserve NameArray(1 To maxParamName, 0 To VariableIndex)
            VariableName = shWeight.Cells(iRow, colW_SpecificInd)

            Application.StatusBar = "Processing Weight for Indicator " & VariableIndex & ": " & VariableName

            NameArray(1, VariableIndex) = VariableIndex 'index
            NameArray(2, VariableIndex) = VariableName 'variable name
            NameArray(3, VariableIndex) = OriginalValue 'original value
            If shWeight.Cells(iRow, colW_CheckScores) = 1 Then
                myRow = myRow + 1
                With shModaMatrix
                    .Cells(myRow, 2) = iRow
                    .Cells(myRow, 3) = OriginalValue
                    .Cells(myRow, 4).FormulaR1C1 = "=if(RC2=R2C1,0,RC3)"
                    .Cells(myRow, 5).FormulaR1C1 = "=RC[-1]/R1C1"
                    .Cells(myRow, 6).FormulaR1C1 = "=if(RC2=R2C1,R3C1,(" & SumWeights & "-R3C1)*RC5)"
                End With
            End If
        End If
    Next iRow

    shModaMatrix.Cells(1, 1).FormulaR1C1 = "=SUM(R1C4:R" & myRow & "C4)"
    Application.StatusBar = "Done"
    Range(shModaMatrix.Cells(FirstRow, 11), shModaMatrix.Cells(LastRow, 11)).value = "=IFERROR(VLOOKUP(RC[-2],R1C2:R1000C6,5,FALSE),RC[-1])"

    shModaMatrix.Calculate
    Application.Calculation = myCalculation
```

```
End Sub


Private Sub CollectData()
    'Set rngNPV = shNPVCalc.Range("D36:D45")
    'Set rngBC = shNPVCalc.Range("D12:D21")

    'Set rngOutput = shWeight.Range("P59:Y59")

    Dim iRow As Long 'current row (in the loop)
    Const FirstRow = 6 'row where variables start in MODEL PARAMETERS
    Dim LastRow 'last row in MODEL PARAMETERS
    LastRow = shWeight.UsedRange.Row + shWeight.UsedRange.Rows.Count - 1

    Dim OriginalValue As Double 'original value of the parameter (from MODEL PARAMETERS)
    Dim MinValue As Double 'Minimum value of the parameter, based on the variability defined in AssignInputValues Sub
    Dim MaxValue As Double 'Maximum value of the parameter, based on the variability defined in AssignInputValues Sub
    Dim Increment As Double 'Increment value (not in percentage terms)
    Dim iValue As Double 'Current Value in percentage terms (in the loop)
    Dim iValueRound As Double 'Current Value in percentage terms, rounded to 5th decimal
    Dim VariableIndex As Integer 'Variable Index (order in the Model Parameters worksheet)
    Dim VariableName As String 'Variable Name


    For iRow = FirstRow To LastRow
        If shWeight.Cells(iRow, colW_SpecificInd) <> "" And shWeight.Cells(iRow, colW_CheckScores) = 1 Then 'only if there is a specific
indicator value in this row

            If IsNumeric(shWeight.Cells(iRow, colW_Weight)) Then
                OriginalValue = shWeight.Cells(iRow, colW_Weight) 'keep the original value (for calculations below and to put it back when the
loop is over)
            Else
                Exit For
            End If

            VariableIndex = VariableIndex + 1
            ReDim Preserve NameArray(1 To maxParamName, 0 To VariableIndex)
            VariableName = shWeight.Cells(iRow, colW_SpecificInd)

            Application.StatusBar = "Processing Weight for Indicator " & VariableIndex & ": " & VariableName

            NameArray(1, VariableIndex) = VariableIndex 'index
            NameArray(2, VariableIndex) = VariableName 'variable name
            NameArray(3, VariableIndex) = OriginalValue 'original value

            iValue = OriginalValue + UserInput
            If iValue > 100 Then iValue = 100 'check if this exceeds 100%

                    iValueRound = Round(iValue, 5)
                    shModaMatrix.Range("A2") = iRow
                    shModaMatrix.Range("A3") = iValue

                    Calculate

                    'shModaMatrix.Select
```

```
                    'Range(shModaMatrix.Cells(FirstRow, 11), shModaMatrix.Cells(FirstRow, 10).End(xlDown).Offset(0, 1)).Select

                    'shWeight.Select
                    'Range(shWeight.Cells(FirstRow, colW_Weight), shWeight.Cells(FirstRow, colW_Weight).End(xlDown)).Select

                Range(shWeight.Cells(FirstRow, colW_Weight), shWeight.Cells(FirstRow + Range("TotalIndicatorsSpecif") - 1, colW_Weight)).value
= _
                        Range(shModaMatrix.Cells(FirstRow, 11), shModaMatrix.Cells(LastRow, 11)).value
                    Call AddRecord(VariableIndex, iValueRound)

                Range(shWeight.Cells(FirstRow, colW_Weight), shWeight.Cells(FirstRow + Range("TotalIndicatorsSpecif") - 1, colW_Weight)).value
= _
                        Range(shModaMatrix.Cells(FirstRow, 10), shModaMatrix.Cells(LastRow, 10)).value 'restoring the original value
            End If
            'shModPar.Cells(iRow, colMP_InUse) = OriginalValue 'restoring the original value

    Next iRow
    Application.StatusBar = "Done"
End Sub

Private Sub AddRecord(ByVal VariableIndex As Integer, iValue As Double)
    Dim iMatrixCol As Long
    Dim iMatrixColMax As Long 'Upper bound of the new array
    Dim iBundle As Integer

    iMatrixCol = UBound(NumArray, 2)
    iMatrixColMax = iMatrixCol + NumberOfBundles
    ReDim Preserve NumArray(1 To UBound(NumArray, 1), 0 To iMatrixColMax) 'increase the size of the array by one column

    For iBundle = 1 To NumberOfBundles
        iMatrixCol = iMatrixCol + 1
        NumArray(1, iMatrixCol) = VariableIndex 'put the variable index value
        NumArray(2, iMatrixCol) = 2 'this is where the VLOOKUP will be used to get the actual variable name
        NumArray(3, iMatrixCol) = iBundle 'put the bundle number into the array
        NumArray(4, iMatrixCol) = 4 'this is where the VLOOKUP will be used to get the actual bundle name
        NumArray(5, iMatrixCol) = iValue 'put the variation parameter into the array
        NumArray(6, iMatrixCol) = rngOutput.Cells(iBundle) 'Val(rngNPV.Cells(iBundle)) 'put NPV values into the array
        NumArray(7, iMatrixCol) = 0 'rngBC.Cells(iBundle) 'Val(rngBC.Cells(iBundle)) 'put B/C ratios into the array
    Next iBundle
End Sub

Private Sub PutDataToWorksheet()
    Dim rngStart As Range 'first cell of the table

Application.Calculation = xlCalculationManual
    For i = 1 To 10
        shSensitivity.Range("sensBundlesScore").Cells(i).Formula = "=IFERROR(ROUND(" & rngOutput.Cells(i) & ",3)," & Chr(34) & Chr(34) & ")"
    Next i

    Set rngStart = shSensitivity.Range("sensBundlesScore").Cells(1).Offset(1, -1)

    For i = 1 To UBound(NameArray, 2)

        rngStart.Offset(i, 0) = NameArray(2, i)
        For j = 1 To NumberOfBundles
```

```
            rngStart.Offset(i, j) = IIf(NumArray(6, (i - 1) * NumberOfBundles + j) = "n/a", "", "=ROUND(" & NumArray(6, (i - 1) *
NumberOfBundles + j) & ",3)")
        Next j
    Next i

Application.Calculation = xlCalculationAutomatic

End Sub
```

Module Restore_Default

```
Option Explicit

Sub MOSAIC_Restore_Default_Values()

If Not continueprocedure() Then Exit Sub

Dim Sh As Worksheet
Set Sh = Sheet6

'Annualization factor
    Sh.Range("E8").Formula = "=G8"
'Unit for display of benefits
    Sh.Range("E10").Formula = "=G10"
'Min and Max score
    Sh.Range("E12").Formula = "=G12"
    Sh.Range("E13").Formula = "=G13"
'Real discount rate
    Sh.Range("E15").Formula = "=G15"
'Real discount rate, carbon emission
    Sh.Range("E19").Formula = "=G19"
'Adjustment to capital costs
    Sh.Range("E23").Formula = "=G23"
'Adjustment to O&M costs
    Sh.Range("E25").Formula = "=G25"

'Value of time personal trips - local
    Sh.Range("E28").Formula = "=G28"
'Value of time business trips - local
    Sh.Range("E32").Formula = "=G32"
'Value of time personal trips - intercity
    Sh.Range("E36").Formula = "=G36"
'Value of time business trips - local
    Sh.Range("E40").Formula = "=G40"
'Value of time Personal trips - high speed rail & air
    Sh.Range("E44").Formula = "=G44"
'Value of time business trips - high speed rail & air
    Sh.Range("E48").Formula = "=G48"
'Value of time truck drivers
    Sh.Range("E52").Formula = "=G52"
'Expected Growth in labor productivity
    Sh.Range("E56").Formula = "=G56"
'Vehicle Operating Costs per Mile, Autos
    Sh.Range("E60").Formula = "=G60"
'Vehicle Operating Costs per Mile, Trucks
```

```
    Sh.Range("E64").Formula = "=G64"

'Social cost of carbon monoxide
    Sh.Range("E69").Formula = "=G69"
'Social cost of VOC
    Sh.Range("E73").Formula = "=G73"
'Social cost of NOX
    Sh.Range("E77").Formula = "=G77"
'Social cost of PM
    Sh.Range("E81").Formula = "=G81"
'Social cost of SO2
    Sh.Range("E85").Formula = "=G85"
'Social cost of O3
    Sh.Range("E89").Formula = "=G89"
'Social cost of Pb
    Sh.Range("E93").Formula = "=G93"
'Social cost of CO2
    Sh.Range("E97").Formula = "=G97"
'Annual growth of social cost of CO2 - 2010-2020
    Sh.Range("E101").Formula = "=G101"
'Annual growth of social cost of CO2 - 2020-2030
    Sh.Range("E105").Formula = "=G105"
'Annual growth of social cost of CO2 - 2030-2040
    Sh.Range("E109").Formula = "=G109"
'Annual growth of social cost of CO2 - 2040-2050
    Sh.Range("E113").Formula = "=G113"

'Value of life
    Sh.Range("E118").Formula = "=G118"
'Elasticity of willingeness-to-pay to avoid death with respect to real income
    Sh.Range("E122").Formula = "=G122"
'Value of Preventing a Category A Injury (incapacitating)
    Sh.Range("E126").Formula = "=G126"
'Value of Preventing a Category B Injury (non-incapacitating)
    Sh.Range("E130").Formula = "=G130"
'PDO Crash Costs
    Sh.Range("E134").Formula = "=G134"

'Lifetime cost of illness - COLORECTAL CANCER
    Sh.Range("E139").Formula = "=G139"
'Annual cost of illness - DIABETES
    Sh.Range("E143").Formula = "=G143"
'Lifetime cost of illness - ALL CARDIO-VASCULAR DISEASES
    Sh.Range("E147").Formula = "=G147"
'Lifetime cost of illness - BREAST CANCER
    Sh.Range("E151").Formula = "=G151"
'Lifetime cost of illness - DEMENTIA
    Sh.Range("E155").Formula = "=G155"
'Annual cost of illness - DEPRESSION
    Sh.Range("E159").Formula = "=G159"

'Pedestrian Environment - STREET LIGHTING
    Sh.Range("E163").Formula = "=G163"
'Pedestrian Environment - CURB LEVEL
    Sh.Range("E167").Formula = "=G167"
```

```
'Pedestrian Environment - INFORMATION PANEL
    Sh.Range("E171").Formula = "=G171"
'Pedestrian Environment - PAVEMENT EVENNESS
    Sh.Range("E175").Formula = "=G175"
'Pedestrian Environment - DIRECTIONAL SIGNAGE
    Sh.Range("E179").Formula = "=G179"
'Pedestrian Environment - BENCHES
    Sh.Range("E183").Formula = "=G183"

'Bicycle User Environment - OFF-ROAD SEGREGATED CYCLE TRACK
    Sh.Range("E187").Formula = "=G187"
'Bicycle User Environment - ON-ROAD SEGREGATED CYCLE LANE
    Sh.Range("E191").Formula = "=G191"
'Bicycle User Environment - ON-ROAD NON-SEGREGATED CYCLE LANE
    Sh.Range("E195").Formula = "=G195"
'Bicycle User Environment - WIDER LANE
    Sh.Range("E199").Formula = "=G199"
'Bicycle User Environment - SHARED BUS LANE
    Sh.Range("E203").Formula = "=G203"

'Marginal External Costs for Noise, Autos, All Highways
    Sh.Range("E207").Formula = "=G207"
'Marginal External Costs for Noise, Trucks, All Highways
    Sh.Range("E211").Formula = "=G211"

End Sub

Function continueprocedure() As Boolean
Dim Config As Integer
Dim Ans As Integer

Config = vbYesNo + vbQuestion + vbDefaultButton2
Ans = MsgBox("Are you sure you want to restore all default values? This cannot be undone.", Config)
If Ans = vbYes Then
    continueprocedure = True
Else
    continueprocedure = False

End If

End Function
```

Module Send_Email

```
Sub EmailSheet()
'Step 1: Declare our variables
    Dim OLApp As Outlook.Application

    Dim OLMail As Object

    Set OutApp = CreateObject("Outlook.Application")
    Set OutMail = OutApp.CreateItem(0)

    On Error Resume Next
```

```
'Step 3: Open Outlook start a new mail item
Set OLApp = New Outlook.Application
Set OLMail = OLApp.CreateItem(0)
OLApp.Session.Logon

'Step 4: Build our mail item and send
With OLMail
    .Display
    .Attachments.Add ActiveWorkbook.FullName
End With
'Step 6: Memory cleanup
Set OLMail = Nothing
Set OLApp = Nothing

End Sub
```

Module Sensitivity

```
'VARIABILITY PARAMETERS
Private UserInput As Double

'ARRAYS
Private NumArray() As String 'array with all the numeric data
Private NameArray() As String 'array with Variable Names

Private Const maxParamNum = 7 'maximum number of parameters in the numeric array
Private Const maxParamName = 9 'maximum number of parameters in the string array

'WORKSHEETS
Private shModPar As Worksheet 'Model Parameters worksheet
Private shNPVCalc As Worksheet 'NPV Calc worksheet
Private shSensitivity As Worksheet 'Sensitivity worksheet

'COLUMN CONSTANTS
Private Const colMP_Category = 2 'column in MODEL PARAMETERS worksheet where "Category" valuse are. Used to see if there is a value in that
row.
Private Const colMP_VarName = 3 'column in MODEL PARAMETERS worksheet where "Variable Name" valuse are
Private Const colMP_InUse = 5 'column in MODEL PARAMETERS worksheet where "In Use" valuse are
Private Const colMP_DistLabel = 6 'column for distribution label
Private Const colMP_Dist = 7 'column for distribution value

'RANGES
Private rngNPV As Range 'NPV range in NPV Calc worksheet
Private rngBC As Range 'B/C range in NPV Calc worksheet

'OTHER
Private Const NumberOfBundles = 10 'Number of bundles

Public Sub Main_Sensitivity()
    Application.ScreenUpdating = False
    Call AssignInputValues
    Call ClearWorksheet
    Call CollectData
    Call PutDataToWorksheet
    Erase NumArray
```

```
    Erase NameArray
    Application.ScreenUpdating = True
End Sub

Private Sub ClearWorksheet()
    shSensitivity.Select
End Sub

Private Sub AssignInputValues()
    ReDim Preserve NumArray(1 To maxParamNum, 0 To 0)
    ReDim Preserve NameArray(1 To maxParamName, 0 To 0)

    Set shModPar = Sheet6
    Set shNPVCalc = Sheet32
    Set shSensitivity = Sheet40

    UserInput = shSensitivity.Range("sensitivityUserInput")

End Sub

Private Sub CollectData()
    Set rngNPV = shNPVCalc.Range("D24:D33")
    Set rngBC = shNPVCalc.Range("D12:D21")

    Dim iRow As Long 'current row (in the loop)
    Const FirstRow = 14 'row where test variables start in MODEL PARAMETERS
    Dim LastRow 'last row in MODEL PARAMETERS
    LastRow = shModPar.UsedRange.Row + shModPar.UsedRange.Rows.Count - 1

    Dim OriginalValue As Double 'original value of the parameter (from MODEL PARAMETERS)
    Dim MinValue As Double 'Minimum value of the parameter, based on the variability defined in AssignInputValues Sub
    Dim MaxValue As Double 'Maximum value of the parameter, based on the variability defined in AssignInputValues Sub
    Dim Increment As Double 'Increment value (not in percentage terms)
    Dim iValue As Double 'Current Value in percentage terms (in the loop)
    Dim iValueRound As Double 'Current Value in percentage terms, rounded to 5th decimal
    Dim VariableIndex As Integer 'Variable Index (order in the Model Parameters worksheet)
    Dim VariableName As String 'Variable Name

    For iRow = FirstRow To LastRow
        If Trim(shModPar.Cells(iRow, colMP_Category)) <> "" And shModPar.Cells(iRow, colMP_Category).Font.Color = vbBlack Then 'only if there
is a category value in this row


            If IsNumeric(shModPar.Cells(iRow, colMP_InUse)) Then
                OriginalValue = shModPar.Cells(iRow, colMP_InUse) 'keep the original value (for calculations below and to put it back when the
loop is over)
            Else
                Exit For
            End If

            VariableIndex = VariableIndex + 1
            ReDim Preserve NameArray(1 To maxParamName, 0 To VariableIndex)
            VariableName = shModPar.Cells(iRow, colMP_VarName)

            Application.StatusBar = "Processing Variable " & VariableIndex & ": " & VariableName
```

```
            NameArray(1, VariableIndex) = VariableIndex 'index
            NameArray(2, VariableIndex) = VariableName 'variable name
            NameArray(3, VariableIndex) = OriginalValue 'original value
            'put distributions
            'LOW
            If IsNumeric(shModPar.Cells(iRow, colMP_Dist)) Then
                NameArray(4, VariableIndex) = shModPar.Cells(iRow, colMP_DistLabel)
                NameArray(5, VariableIndex) = shModPar.Cells(iRow, colMP_Dist)
                If NameArray(5, VariableIndex) = "" Then GoTo Skip
            End If
            'MID
            If IsNumeric(shModPar.Cells(iRow + 1, colMP_Dist)) Then
                NameArray(6, VariableIndex) = shModPar.Cells(iRow + 1, colMP_DistLabel)
                NameArray(7, VariableIndex) = shModPar.Cells(iRow + 1, colMP_Dist)
                If NameArray(7, VariableIndex) = "" Then GoTo Skip
            End If
            'HIGH
            If IsNumeric(shModPar.Cells(iRow + 2, colMP_Dist)) Then
                NameArray(8, VariableIndex) = shModPar.Cells(iRow + 2, colMP_DistLabel)
                NameArray(9, VariableIndex) = shModPar.Cells(iRow + 2, colMP_Dist)
                '***if "" then
            End If

Skip:


                    iValue = OriginalValue * (1 + UserInput)
                    iValueRound = Round(iValue, 5)
                    shModPar.Cells(iRow, colMP_InUse) = iValue 'OriginalValue * (1 + iValue)
                    'Calculate 'in case we need to calculate manually
                    Call AddRecord(VariableIndex, iValueRound)
            shModPar.Cells(iRow, colMP_InUse) = OriginalValue 'restoring the original value
        End If
    Next iRow
    Application.StatusBar = "Done"


End Sub

Private Sub AddRecord(ByVal VariableIndex As Integer, iValue As Double)
    Dim iMatrixCol As Long
    Dim iMatrixColMax As Long 'Upper bound of the new array
    Dim iBundle As Integer

    iMatrixCol = UBound(NumArray, 2)
    iMatrixColMax = iMatrixCol + NumberOfBundles
    ReDim Preserve NumArray(1 To UBound(NumArray, 1), 0 To iMatrixColMax) 'increase the size of the array by one column

    For iBundle = 1 To NumberOfBundles
        iMatrixCol = iMatrixCol + 1
        NumArray(1, iMatrixCol) = VariableIndex 'put the variable index value
        NumArray(2, iMatrixCol) = 2 'this is where the VLOOKUP will be used to get the actual variable name
        NumArray(3, iMatrixCol) = iBundle 'put the bundle number into the array
        NumArray(4, iMatrixCol) = 4 'this is where the VLOOKUP will be used to get the actual bundle name
```

```
        NumArray(5, iMatrixCol) = iValue 'put the variation parameter into the array
        NumArray(6, iMatrixCol) = rngNPV.Cells(iBundle) 'Val(rngNPV.Cells(iBundle)) 'put NPV values into the array
        NumArray(7, iMatrixCol) = rngBC.Cells(iBundle) 'Val(rngBC.Cells(iBundle)) 'put B/C ratios into the array
    Next iBundle
End Sub

Private Sub PutDataToWorksheet()
    Dim rngStart As Range 'first cell of the table
    Dim rngBCStart As Range
    Dim rngNPVStart As Range

Application.Calculation = xlCalculationManual
    shSensitivity.Range("sensBundlesBC").value = Application.WorksheetFunction.Transpose(rngBC)
'Application.WorksheetFunction.Transpose(NumArray)
    shSensitivity.Range("sensBundlesNPV").value = Application.WorksheetFunction.Transpose(rngNPV)
'Application.WorksheetFunction.Transpose(NumArray)

    Set rngBCStart = shSensitivity.Range("sensBundlesBC").Cells(1).Offset(1, -1)
    Set rngNPVStart = shSensitivity.Range("sensBundlesNPV").Cells(1).Offset(1, -1)

    For i = 1 To UBound(NameArray, 2)
        rngBCStart.Offset(i, 0) = NameArray(2, i)
        For j = 1 To NumberOfBundles
            rngBCStart.Offset(i, j) = NumArray(7, (i - 1) * NumberOfBundles + j)
        Next j
    Next i

    For i = 1 To UBound(NameArray, 2)
        rngNPVStart.Offset(i, 0) = NameArray(2, i)
        For j = 1 To NumberOfBundles
            rngNPVStart.Offset(i, j) = NumArray(6, (i - 1) * NumberOfBundles + j)
        Next j
    Next i
Application.Calculation = xlCalculationAutomatic

End Sub
```