VMT Computation Procedures (DRAFT)

Notice: SACOG is inviting the public to review the Outside-the-Region VMT Estimation Methodology and revised SACOG SB 743 regional screening maps. Public comment period is open from October 1st - 23rd. Please send comments to sacsim@sacog.org. Final documentation and updated SB743 screening maps will be posted on the SACOG website by October 30th, 2020.

Background and Introduction

SACSIM19 simulates several key types of travel by residents (as opposed to commercial trips like freight, deliveries, etc.):

- Trips by SACOG residents to destinations within the SACOG region. These are known as internal-internal, or II trips. These trips are modeled by the DAYSIM submodel.
- Trips by SACOG residents to destinations outside the SACOG region, known as internal-external, or IX trips.
- Trips by non-SACOG residents to destinations in the SACOG region, known as external-internal, or XI trips.
- IX and XI trips are both modeled by the IXXI submodel.

All the above trip types generate VMT. Important to note is that the model does not currently predict VMT generated outside the region, e.g., for IX trips, the model only counts the VMT generated from the trip origin in the region until the region's border. Methodology for estimating VMT that is outside the region is documented seperately and avaliable on the SACOG modeling webpage https://www.sacog.org/analysis-tools.

This document describes the technical procedures to compute household generated or residential VMT (Residential VMT is used in below paragraphs) and Work related VMT in the following order:

- Compute internal-internal, or II, VMT based on trip table from DAYSIM
- Compute internal-external and external-internal, or IXXI, VMT based on vehicle trip matrix from IXXI sub-model
- Tally residential VMT to parcels
    - Tally internal-internal residential VMT to parcels
    - Tally internal-external residential VMT to parcels
    - Compute household generated VMT per capita
- Tally internal-internal work related VMT to job location parcels
    - Compute Work-related VMT per job

Cube Voyager programming scripts used are included as appendices at the end for references. Other software and programs are also needed to perform calculations and joining tables.

Step 1 Compute internal-internal VMT based on trip table from DAYSIM

When the SACSIM model run completes, it produces the _trips.tsv file, which is a table of all internal-internal trips. However, because the trip distance in the original table is estimated based on the congested speed prior to the last global iteration, the user must run a Cube Voyage script, provided in Appendix A, to estimate the distance based on the final iteration network congestion. This script attaches trip distances to each trip by creating a temporary O-D distance skim based on the final loaded model network, then tagging the skim distance values to each trip based on the trip's origin and destination TAZs.

The output of this supplementary Cube script is a CSV file, "_trip_1_1.csv", which has the same table as _trips.tsv but with the following attributes added to each trip:

- timeau - updated travel time by auto
- distau - updated trip distance by auto
- distcong - congested distance, and

Table 1 describes the fields in the "_trips_1_1.csv" output CSV file. Below is the equation to compute each trip's VMT. Factors are applied to the trip distance based on trip MODE.

VMT = distau if MODE = 3 (Drive Alone)
distau*0.5 if MODE = 4 (Shared Drive 2)
distau*0.3 if MODE = 5 (Shared Drive 3 or more)

Table 1 – Person Trip Table Output (_trip_1_1.csv) Variables and Descriptions

	Variable	Description and potential values (if applicable)
Variables in the trip table direct from DAYSIM	id	Unique Trip ID
	tour_id	Unique Tour ID
	hhno	Unique Household ID
	pno	Person ID in the household
	day	
	tour	Tour ID for the person
	half	1-first half tour,2-secon half
	tseg	Trip sequence number in the half tour
	tsvid	

	opurp	0 - none/home 1 - work 2 - school 3 - escort 4 - pers.bus 5 - shop 6 - meal 7 - social 8 - recreational (combined with social) 9 - medical (combined with pers.bus.) 10 - change mode inserted purpose
	dpurp	0 - none/home 1 - work 2 - school 3 - escort 4 - pers.bus 5 - shop 6 - meal 7 - social 8 - recreational (combined with social) 9 - medical (combined with pers.bus.) 10 - change mode inserted purpose
	oadtyp	1 - Home 2 - Usual workplace 3 - Usual School 4 - Other 5 - Missing 6 - Change mode inserted location
	dadtyp	1 - Home 2 - Usual workplace 3 - Usual School 4 - Other 5 - Missing 6 - Change mode inserted location
	opcl	Origin Parcel
	otaz	Origin TAZ
	dpcl	Destination Parcel
	dtaz	Destination TAZ

	mode	1 - walk 2 - bike 3 - sov 4 - hov2 5 - hov3+ 6 - transit 8 - school bus 9 - other
	pathtype	0 - none 1 - full network 2 - no-toll network 3 - localbus 4 - light rail 5 - premium bus 6 - commuter rail 7 - ferry
	dorp	1 - Driver 2 - Passenger 3 - N/A 9 - Missing
	deptm	Departure Time - minutes from middle night
	arrtm	Arrival Time - minutes from middle night
	endacttm	Activity End Time - minutes from middle night
	travtime	Travel Time – minutes
	travcost	Travel Cost – minutes
	travdist	Travel Distance – mile
	vot	Value of Time
	trexpfac	trip expansion factor
	timeau	updated travel time by auto
	distau	updated travel distance by auto
	distcong	congested distance
	VMT	IF(mode=3,distau,IF(mode=4,distau*0.5,IF(mode=5,distau*0.3)))

Step 2 Compute internal-external VMT based on vehicle trip matrix from IXXI sub-model

After running the Cube script in Appendix A to add travel times and distances to the output trip table, a Cube Voyager script, shown in Appendix B, is run to compute VMT and other variables for both IXXI and commercial trips. Please note that the script in Appendix A must be executed prior the script in Appendix B because the temporary skims created from the Appendix A script

are used. Table 2 shows the fields for the output created by the IXXI VMT Cube script in Appendix B.

Table 2 - IXXI VMT and other variables by TAZ

Variable	Description	Values
I	Traffic Analysis Zone (TAZ)	Gateway zones = 1-30, internal zones = 31-1533
IX_VT_I	Vehicle trips originating at zone I	
IX_VT_J	Vehicle trips ending at zone I	
IX_VHT_I	Vechicle Hours originating at zone I	
IX_VHT_J	Vechicle Hours ending at zone I	
IX_VMT_I	VMT originating at zone I	
IX_VMT_J	VMT ending at zone I	
IX_CVMT_I	Commercial VMT originating at zone I	
IX_CVMT_J	Commercial VMT ending at zone I	
HHS	Households in zone I	
EMPTOT	Jobs in zone I	
FOOD	Jobs in Food sector in zone I	
RET	Retail jobs in zone I	
SVC	Service Jobs in zone I	
IX_VMT_RES (internal-external VMT made by SACOG residents)	= (IX_VMT_I+IX_VMT_J) *(HHS / (1 + HHS + 1.1*(EMPTOT – FOOD - RET -0.25* SVC)))	

Step 3 Tally residential VMT to parcels

This section will cover how to:

- Tally internal-internal residential VMT to parcels
- Tally internal-external residential VMT to parcels
- Compute residential VMT per capita

3.1 Tally Internal-Internal VMT by Residence Parcels

A. Join **_household.tsv** (direct output from DAYSIM) to **_trip_1_1.csv** using the HHNO field. The resulting table will be for individual trips and have two key variables:
   - Vmt – the trip's VMT, from _trip_1_1.csv and calculated in Step 1
   - Hhparcel – the parcel on which the household is located. This value comes from the _household.tsv table.

B. From the joined table, calculate the sum of total VMT for each parcel.

C. Summarize VMT by HHPARCEL in _household.tsv. Variable HHPARCEL is the unique identifier for the traveler's residence parcel. The ID for this household HHNO is 17. The parcel ID HHPARCEL is 101013667 as Table 2 shows. The total VMT from this household on parcel 101013667 is 3.04 miles. The total residential VMT on this parcel should also include the VMT made by all residents on this parcel regardless the destinations.

Table 3 An Example of Household Table **_household.tsv**

Variable	Sample Household	Definition
Hhno	17	Household id
fraction_with_jobs_outside	0.028	Residence zone worker IX fraction
Hhsize	1	Household size
Hhvehs	1	Vehicles available
Hhwkrs	1	Household workers
Hhftw	1	HH full time workers (type 1)
Hhptw	0	HH part time workers (type 2)
Hhret	0	HH retired adults (type 3)
Hhoad	0	HH other adults (type 4)
Hhuni	0	HH college students (type 5)
Hhhsc	0	HH high school students (type 6)
hh515	0	HH kids age 5-15 (type 7)
hhcu5	0	HH kids age 0-4 (type 8)
Hhincome	11722	Household income ($)
Hownrent	3	Household own or rent
Hrestype	3	Household residence type
Hhparcel	101013667	Residence parcel id
zone_id	1241	Internal id based on parcel id
Hhtaz	1242	Based on parcel id
Hhexpfac	1	HH expansion factor
Samptype	1	Sample type

3.2 Tally internal-external residential VMT by parcels

A. Estimate the shares of IXXI (combined IX + XI) VMT made by residents in SACOG region at **TAZ** level using variables in Table 2 with the following equation:

$$\text{IX\_VMT\_RES} = (\text{IX\_VMT\_I} + \text{IX\_VMT\_J}) * \left(\frac{HHS}{1 + HHS + 1.1 * (EMPTOT - FOOD - RET - 0.25 * SVC)}\right)$$

B. Compute IXXI residential VMT per capita rate by RAD

- Join the IX VMT by TAZ table (Table 2) to the table in "tazrad07.txt" using TAZ values as the join key. The resulting table will tag each TAZ with the Regional Analysis District (RAD) that it falls within.
- Join resulting table to "raw_household.txt" using HHTAZ as the join key in the household table and "TAZ" in the join key in the TAZ/RAD table, which will result in a table with each record being a household with TAZ and RAD tagged to it.

In the resulting table, sum residential VMT (IX_VMT_RES) by RAD and total population (HHSIZE) by RAD.

- Compute the IX_VMT_RES per capita at RAD level by dividing each RAD's total IX_VMT_RES by its total population. This should result in a table, at the RAD level, listing each RAD and its per-capita VMT, or RES_VMT_RATE

C. Compute IXXI residential VMT at parcel level
   - Using the household-level table and per-capita VMT by RAD table created in step B:
     i. Join the household-level table to the per-capita VMT by RAD table using RAD as the join key
     ii. Calculate each household's VMT by multiplying its HHSIZE * RES_VMT_RATE of its corresponding RAD, the resulting product will be HHVMT
     iii. Sum total HHVMT by HHPARCEL values to get total residential VMT for each parcel.

3.3 Compute household generated VMT per Capita by any Areas

A. If you completed the steps above in 3.2, you will have a table that has, at a minimum, the following data for each parcel: total population, total residential VMT, and residential VMT per capita. There are several approaches to getting VMT per capita for larger geographies.
B. To calculate for certain, model-derived geographies like TAZ, RAD, or census geographies:
   a. Join the parcel-level VMT per capita table (from step 3.2) to the "parcel.txt" table using PARCELID as the join key. The parcel.txt table contains detailed data on each parcel including several geography keys like RAD, TAZ, and several census geographies.
   b. After joining, you can get aggregate VMT per capita values, e.g. VMT/capita for each TAZ by doing the following:
      i. Get total population and total VMT within each larger geographic area
      ii. Divide each TAZ's VMT by its population, which will return the VMT per capita for each area of that geography type (e.g. TAZ, RAD, etc)
C. To calculate for custom geographies that are not listed in the parcel.txt table (e.g. if you want VMT/capita for each city in the SACOG region, or VMT/capita for special districts), do the following:
   a. Acquire necessary GIS shapefiles:
      i. Parcel points

ii. Polygons of the geographies whose VMT/capita you want to calculate

b. In GIS, spatially join the polygon data on to the parcel points based on which polygon each parcel point lies within. E.g., if you have a polygon file of council districts, the spatial join operation will return a new version of the parcel point file that, in addition to the original parcel-level data, will now indicate which district each parcel point falls within.
c. Using PARCELID as the join key, join the parcel point table, now with the district information, to the parcel-level VMT per capita table you created in Step 3.2. The resulting table will have PARCELID, district ID, and VMT/capita for each parcel.
d. With this resulting table, you can now get VMT/capita for each district using the same process described in step 3.3.B

Step 4 Tally Internal-Internal Work-related VMT to Job Locations

A. Join "_trip_1_1.csv" to "_tour.tsv", which will result in the table with the following columns that are necessary for computing work location-based VMT:

Table 4-1: Key variables used in calculating work-related VMT to job location

Field	Source table	Description
PDPURP	_tour.tsv	Primary destination purpose for tour
PARENT	_tour.tsv	Parent tour, 1 = tour is a work-based subtour
TDPCL	_tour.tsv	Tour's primary destination parcel
TOPCL	_tour.tsv	If tour is a subtour, this is the primary destination parcel of the parent tour
VMT	_trip_1_1.csv	Calculated from distau and mode columns in trip table

B. Filter Table 4-1 to only have trips on work tours, i.e., where PDPURP = 1
C. Compute sum of VMT for each tour destination parcel (TDPCL). This is the VMT of primary work tours that have the parcel as their destination. But it does not yet have the VMT generated by subtours that start at the parcel.
D. Starting with Table 4-1 unfiltered, filter Table 4-1 so that PARENT > 0, so that you only have work-based subtours that start and end at the workplace, regardless of trip purpose.
E. Compute sum of VMT for each TOPCL, which is the origin parcel of the subtour. This VMT is that which is generated by subtours originating at each parcel.
F. For each parcel, sum its primary tour VMT (from step C) with its subtour VMT (from step E). By now you should be able to make a table that lists each parcel and the total

internal-internal trip VMT generated by SACOG region residents that is associated with that parcel.

G. To compute Work related VMT per job at job site, it is suggested to deduct the jobs taking by external workers, since the total VMT calculated in steps A-F only consider SACOG residents. To do so:
    - Open the table "worker_ixxifractions.dat" and join it to the parcel level table from steps A-F using TAZ as the join key. The worker_ixxifractions has 3 variables - TAZ, the share of TAZ residents going outside the region for work, and the share of workers who live outside the region who come into the region to work in that TAZ.
    - Estimate the number of external workers for each parcel by multiplying its fraction of external workers (from the worker_ixxifractions.dat table) by the total number of jobs on the parcel (from the parcel table).
    - Subtract the number of external workers from the total workers to get the approximate number of SACOG resident workers whose job location is at the parcel.
    - Divide that parcel's work-end VMT (calculated in steps A-F) by its estimated total resident workers to get the parcel's average VMT per job.

Table 4 An Example of Tour Table **_tour.tsv**

Variable	Example	Definition
id	171	internal daysim record ID
person_id	17	internal daysim record ID
person_day_id	17	internal daysim record ID
hhno	17	Household id
pno	1	person seq no on file
day	1	Diary / simulation day ID
tour	1	tour id
jtindex	0	hh joint tour index
parent	0	parent tour id if >0 meaning it is a sub-tour
subtrs	0	number of subtours
pdpurp	1	Tour destination purpose 1- work 2- school 3-
tlvorig	513	time leave tour origin
tardest	567	time larrive tour dest
tlvdest	1129	time leave tour dest
tarorig	1135	time arrive tour origin
toadtyp	1	tour origin address type
tdadtyp	2	tour destination address type

topcl	101013667	tour origin parcel
totaz	1242	tour origin TAZ
tdpcl	101013930	tour dest parcel
tdtaz	1243	tour destination TAZ
tmodetp	5	tour main mode type
tpathtp	1	tour main mode path type
tautotime	1.157218359	tour 1-way auto time
tautocost	0.061806588	tour 1-way auto distance
tautodist	0.490617922	tour 1-way auto cost
tripsh1	3	1st half tour # of trips
tripsh2	2	2nd half tour # of trips
phtindx1	0	1st half-partial joint half tour index
phtindx2	0	2nd half-partial joint half tour index
fhtindx1	0	1s half- fully joint half tour index
fhtindx2	0	2nd half- fully joint half tour index
toexpfac	1	trip expansion factor

Appendix A Cube Voyager Script – Attach Skims to Trips

```
;-------------------------------------------------------------------------------

; Notes:

;

;-------------------------------------------------------------------------------

loop p=1,9

    if (p=01) per='h07'

    if (p=02) per='h08'

    if (p=03) per='h09'

    if (p=04) per='md5'

    if (p=05) per='h15'

    if (p=06) per='h16'

    if (p=07) per='h17'

    if (p=08) per='ev2'

    if (p=09) per='n11'



  votcl=1

  tnt=1

  ivot=60/12.51 ;need to update to match tolling ivots

  tntst = 'nt'


;-----------------------------------------------------------------

;step 8

IF (p=1,2,3,5,6,7) ; am,pm end loop at 739

  RUN PGM=HIGHWAY  MSG='step 8 Highway skims'

  ; Highway skims for all occupancies, at one period, VOT, tolling class at a time
```

```
 NETI=vo.@per@.net                ; input network
MATO=tempsk@per@.mat, MO=1-9,dec=9*2,    ; output skim matrices
name=da_time, da_dist,da_cdist, s2_time,s2_dist,s2_cdist, s3_time, s3_dist,s3_cdist

;IFCLUSTER:
DistributeIntraStep processid='sacsimsub', ProcessList=1-3, mingroupsize=400

PHASE=LINKREAD                   ;define link groups

  if (li.vc_1>=1.0)
    comp lw.vc10dist=li.distance
  else
    lw.vc10dist=0.0
  ENDIF

; Settings for network path choice based on configuration settings
CostPerMile = 0.17
HOV2Divisor = 1.00;  was 1.66 - No need to divide by occupancy since DaySim places shared ride trips in higher VOT class
HOV3Divisor = 1.00; was 2.23

            IF (li.USECLASS == 0) ADDTOGROUP=1       ;GENERAL PURPOSE
 IF (li.USECLASS == 2) ADDTOGROUP=2       ;HOV2
 IF (li.USECLASS == 3) ADDTOGROUP=3       ;HOV3+
```

```
lw.AOCost = li.distance * CostPerMile
if (@tnt@ <= 0)     ;if No-Toll class...
   tollivot = 150   ;severe perception factor for tolls, for the no-toll class
else
   tollivot = @ivot@
endif

lw.imped_da = li.time_1 + (li.tollda*tollivot + lw.AOCost*@ivot@)
lw.imped_s2 = li.time_1 + (li.tolls2*tollivot + lw.AOCost*@ivot@) / HOV2Divisor
lw.imped_s3 = li.time_1 + (li.tolls3*tollivot + lw.AOCost*@ivot@) / HOV3Divisor
endphase


        PHASE=ILOOP
        ;Skim SOV paths without HOV links
                PATHLOAD PATH=lw.imped_da,EXCLUDEGRP=2,3,
                  mw[1]=pathtrace(li.time_1), noaccess=0,
                  mw[2]=pathtrace(li.distance), noaccess=0,
                  mw[3]=pathtrace(lw.vc10dist), noaccess=0

        ; Skim SR2 paths with HOV links
                PATHLOAD PATH=lw.imped_s2,EXCLUDEGRP=3,
                  mw[4]=pathtrace(li.time_1), noaccess=0,
                  mw[5]=pathtrace(li.distance), noaccess=0,
                  mw[6]=pathtrace(lw.vc10dist), noaccess=0

        ; Skim SR3 paths with HOV links
```

```
            PATHLOAD PATH=lw.imped_s3,
                mw[7]=pathtrace(li.time_1), noaccess=0,
                mw[8]=pathtrace(li.distance), noaccess=0,
                mw[9]=pathtrace(lw.vc10dist), noaccess=0



    ;Intrazonals
      ;Intrazonals
                iz1=lowest(1,1,0.005,10000)*0.5
                iz2=lowest(2,1,0.005,10000)*0.5
                iz4=lowest(4,1,0.005,10000)*0.5
                iz5=lowest(5,1,0.005,10000)*0.5
                iz7=lowest(7,1,0.005,10000)*0.5
                iz8=lowest(8,1,0.005,10000)*0.5
                jloop j=i
                    mw[1]=iz1
                    mw[2]=iz2
                    mw[4]=iz4
                    mw[5]=iz5
                    mw[7]=iz7
                    mw[8]=iz8
                endjloop


    ENDPHASE
ENDRUN
 ;======md,ev,ni=========================
```

```
ELSE


RUN PGM=HIGHWAY  MSG='step 8 Highway skims'
; Highway skims for all occupancies, at one period, VOT, tolling class at a time


NETI=vo.@per@.net                    ; input network
MATO=tempsk@per@.mat, MO=1-9,dec=9*2,    ; output skim matrices
name=da_time, da_dist,da_cdist, s2_time,s2_dist,s2_cdist, s3_time, s3_dist,s3_cdist


;IFCLUSTER:
DistributeIntraStep processid='sacsimsub', ProcessList=1-3, mingroupsize=400


PHASE=LINKREAD                    ;define link groups


if (li.vc_1>=1.0)
  comp lw.vc10dist=li.distance
else
  lw.vc10dist=0.0
ENDIF



; Settings for network path choice based on configuration settings
CostPerMile = 0.17
HOV2Divisor = 1.00;  was 1.66 - No need to divide by occupancy since DaySim places shared ride trips in higher VOT class
HOV3Divisor = 1.00; was 2.23
```

```
            IF (li.USECLASS == 0) ADDTOGROUP=1       ;GENERAL PURPOSE
      IF (li.USECLASS == 2) ADDTOGROUP=2       ;HOV2
      IF (li.USECLASS == 3) ADDTOGROUP=3       ;HOV3+
            IF (li.USECLASS == 4) ADDTOGROUP=4       ;3+ axle commercial (for off peak)

       lw.AOCost = li.distance * CostPerMile
       if (@tnt@ <= 0)     ;if No-Toll class...
          tollivot = 150   ;severe perception factor for tolls, for the no-toll class
       else
          tollivot = @ivot@
       endif

       lw.imped_da = li.time_1 + (li.tollda*tollivot + lw.AOCost*@ivot@)
       lw.imped_s2 = li.time_1 + (li.tolls2*tollivot + lw.AOCost*@ivot@) / HOV2Divisor
       lw.imped_s3 = li.time_1 + (li.tolls3*tollivot + lw.AOCost*@ivot@) / HOV3Divisor
     endphase


      PHASE=ILOOP

          ;Skim SOV paths without HOV links
       PATHLOAD PATH=lw.imped_da,EXCLUDEGRP=2,3,
         mw[1]=pathtrace(li.time_1), noaccess=0,
         mw[2]=pathtrace(li.distance), noaccess=0,
         mw[3]=pathtrace(lw.vc10dist), noaccess=0

    ; Skim SR2 paths with HOV links
```

```
PATHLOAD PATH=lw.imped_s2,EXCLUDEGRP=3,
  mw[4]=pathtrace(li.time_1), noaccess=0,
  mw[5]=pathtrace(li.distance), noaccess=0,
  mw[6]=pathtrace(lw.vc10dist), noaccess=0

; Skim SR3 paths with HOV links
  PATHLOAD PATH=lw.imped_s3,
  mw[7]=pathtrace(li.time_1), noaccess=0,
  mw[8]=pathtrace(li.distance), noaccess=0,
  mw[9]=pathtrace(lw.vc10dist), noaccess=0

  ;Intrazonals
            iz1=lowest(1,1,0.005,10000)*0.5
            iz2=lowest(2,1,0.005,10000)*0.5
            iz4=lowest(4,1,0.005,10000)*0.5
            iz5=lowest(5,1,0.005,10000)*0.5
            iz7=lowest(7,1,0.005,10000)*0.5
            iz8=lowest(8,1,0.005,10000)*0.5
            jloop j=i
                  mw[1]=iz1
                  mw[2]=iz2
                  mw[4]=iz4
                  mw[5]=iz5
                  mw[7]=iz7
                  mw[8]=iz8
            endjloop
      endphase
```

```
      ENDRUN


    ; 9 auto time periods

    endif

 ENDLOOP

;-------------------------------------------------------------------------------

run pgm=matrix

; Attach skim data to trip records


;dbi[1]=.\_trip_1.tsv, delimiter[1]=',t', fields=1


FILEI RECI = _trip.tsv,
id=1,tour_id=2,hhno=3,pno=4,day=5,tour=6,half=7,tseg=8,tsvid=9,opurp=10,dpurp=11,oadtyp=12,dadtyp=13,opcl=14,otaz=15,dpcl=16,dtaz=17,

mode=18,pathtype=19,dorp=20,deptm=21,arrtm=22,endacttm=23,travtime=24,travcost=25,travdist=26,vot=27,trexpfac=28,

delimiter[1]=' ,t', SORT=otaz


MATI[1]=tempskh07.mat ;7am skim file

MATI[2]=tempskh08.mat

MATI[3]=tempskh09.mat

MATI[4]=tempskmd5.mat

MATI[5]=tempskh15.mat

MATI[6]=tempskh16.mat

MATI[7]=tempskh17.mat

MATI[8]=tempskev2.mat

MATI[9]=tempskn11.mat
```

```
if (reci.recno=1)
   loop f=1, reci.numfields
      if (reci.cfield[f]='id')      f_id      = f
      if (reci.cfield[f]='tour_id')  f_tour_id  = f
      if (reci.cfield[f]='hhno')    f_hhno     = f
      if (reci.cfield[f]='pno')     f_pno      = f
      if (reci.cfield[f]='day')     f_day      = f
      if (reci.cfield[f]='tour')    f_tour     = f
      if (reci.cfield[f]='half')    f_half     = f
      if (reci.cfield[f]='tseg')    f_tseg     = f
      if (reci.cfield[f]='tsvid')   f_tsvid    = f
      if (reci.cfield[f]='opurp')   f_opurp    = f
      if (reci.cfield[f]='dpurp')   f_dpurp    = f
      if (reci.cfield[f]='oadtyp')  f_oadtyp   = f
      if (reci.cfield[f]='dadtyp')  f_dadtyp   = f
      if (reci.cfield[f]='opcl')    f_opcl     = f
      if (reci.cfield[f]='otaz')    f_otaz     = f
      if (reci.cfield[f]='dpcl')    f_dpcl     = f
      if (reci.cfield[f]='dtaz')    f_dtaz     = f
      if (reci.cfield[f]='mode')    f_mode     = f
      if (reci.cfield[f]='pathtype') f_pathtype = f
      if (reci.cfield[f]='dorp')    f_dorp     = f
      if (reci.cfield[f]='deptm')   f_deptm    = f
      if (reci.cfield[f]='arrtm')   f_arrtm    = f
      if (reci.cfield[f]='endacttm')  f_endacttm  = f
      if (reci.cfield[f]='travtime')  f_travtime  = f
```

```
      if (reci.cfield[f]='travcost')    f_travcost  = f
      if (reci.cfield[f]='travdist')    f_travdist  = f
      if (reci.cfield[f]='vot')     f_vot     = f
      if (reci.cfield[f]='trexpfac') f_trexpfac = f
   endloop
  print file=.\_trip_1_1.csv, list=
'id,tour_id,hhno,pno,day,tour,half,tseg,tsvid,opurp,dpurp,oadtyp,dadtyp,opcl,otaz,dpcl,dtaz,
mode,pathtype,dorp,deptm,arrtm,endacttm,travtime,travcost,travdist,vot,trexpfac,timeau,di
stau,distcong'
else
     trip_id = val(reci.cfield[f_id])
     tour_id = val(reci.cfield[f_tour_id])
     hhno  = val(reci.cfield[f_hhno])
     pno  = val(reci.cfield[f_pno])
     day  = val(reci.cfield[f_day])
     tour  = val(reci.cfield[f_tour])
     half   = val(reci.cfield[f_half])
     tseg  = val(reci.cfield[f_tseg])
     tsvid = val(reci.cfield[f_tsvid])
     opurp  = val(reci.cfield[f_opurp])
     dpurp  = val(reci.cfield[f_dpurp])
     oadtyp  = val(reci.cfield[f_oadtyp])
     dadtyp  = val(reci.cfield[f_dadtyp])
     opcl  = val(reci.cfield[f_opcl])
     otaz  = val(reci.cfield[f_otaz])
     dpcl  = val(reci.cfield[f_dpcl])
     dtaz  = val(reci.cfield[f_dtaz])
     mode  = val(reci.cfield[f_mode])
```

```
pathtype  = val(reci.cfield[f_pathtype])
dorp  = val(reci.cfield[f_dorp])
deptm  = val(reci.cfield[f_deptm])
arrtm = val(reci.cfield[f_arrtm])
endacttm  = val(reci.cfield[f_endacttm])
travtime  = val(reci.cfield[f_travtime])
travcost  = val(reci.cfield[f_travcost])
travdist  = val(reci.cfield[f_travdist])
vot  = val(reci.cfield[f_vot])
trexpfac  = val(reci.cfield[f_trexpfac])

; Segment the trip time
 array seghr=9, tripseg=9
 seghr[ 1]= 7
 seghr[ 2]= 8
 seghr[ 3]= 9
 seghr[ 4]= 10
 seghr[ 5]= 15
 seghr[ 6]= 16
 seghr[ 7]= 17
 seghr[ 8]= 18
 seghr[ 9]= 20

   arrhr = ri.arrtm/60 ;arrival hour
   dephr = ri.deptm/60 ;departure hour
   durhr = (arrhr - dephr) ;trip duration, in hours
   if (durhr = 0)
```

```
        arrhr = arrhr + 0.0001 ;ensure there are no trips with zero duration
        durhr = 0.0001
      elseif (durhr < 0)
        durhr = durhr + 24
      endif

    ; Separate the after-midnight portion only if trip straddles midnight
    ;     into trip1: dephr to 24 (alias arrhr),
    ;     and trip2: 0 to arrhr (alias arr2)
      if (arrhr < dephr) ;if departure hour is from previous day (e.g., from 11pm-1am), then split trip into two, with first ending at midnight and second starting at midnight
        arr2 = arrhr
        arrhr = 24
      else
        arr2 = 0
      endif

    ; Fraction of trip within each period
    ; E.g. if trip started at 8:30 and ended 9:30, for loop s = 2 it'd be (9am - 8:30am) + 0, or 30mins, or 50% of trip in time period one, then 1-50%, or 0.5, in period 2
    ; does this work for trips that straddle 3 or more time periods? Yes, it does. Tested.
      ni = 1 ;ni = network in?
      loop s=1,8 ;don't loop through s[9] because each loop goes to s[n+1]
        hrbeg = seghr[s] ;returns hour of period s
        hrend = seghr[s+1] ;returns hour of period (s+1)
        t_part = (max(0, min(hrend, arrhr) - max(hrbeg, dephr)) + ;(duration of trip in period)/total duration of trip.
                 max(0, min(hrend, arr2 ) - hrbeg)) / durhr
```

```
            tripseg[s] = t_part
        ;       ni9 = ni9 - tseg
            ni = ni -t_part
        endloop
        ni = max(0, ni)

    ; To fields
        a1 = tripseg[1] ;portion of trip happening in time segment 1
        a2 = tripseg[2]
        a3 = tripseg[3]
        md = tripseg[4]
        p1 = tripseg[5]
        p2 = tripseg[6]
        p3 = tripseg[7]
        ev = tripseg[8]
        ni = ni


; Select skims


IF (mode = 5)   ; S3
    a1timeau   = a1*matval(1, 7, otaz, dtaz)    ;MatVal( filenumber, tablenumber, i, j, failvalue)
    a1distau   = a1*matval(1, 8, otaz, dtaz) ;(portion of trip in time period 1) * (period 1 skim distance from trip's origin to trip's destination)
    a1distcong   = a1*matval(1, 9, otaz, dtaz) ;matrix value for i-j combo in the congested distance tab (tab 9) of the first skim file (7am period)
    a2timeau   = a2*matval(2, 7, otaz, dtaz)
    a2distau   = a2*matval(2, 8, otaz, dtaz)
```

a2distcong = a2*matval(2, 9, otaz, dtaz)

a3timeau = a3*matval(3, 7, otaz, dtaz)

a3distau = a3*matval(3, 8, otaz, dtaz)

a3distcong = a3*matval(3, 9, otaz, dtaz)

mdtimeau = MD*matval(4, 7, otaz, dtaz)

mddistau = MD*matval(4, 8, otaz, dtaz)

mddistcong = MD*matval(4, 9, otaz, dtaz)

p1timeau = P1*matval(5, 7, otaz, dtaz)

p1distau = P1*matval(5, 8, otaz, dtaz)

p1distcong = P1*matval(5, 9, otaz, dtaz)

p2timeau = P2*matval(6, 7, otaz, dtaz)

p2distau = P2*matval(6, 8, otaz, dtaz)

p2distcong = P2*matval(6, 9, otaz, dtaz)

p3timeau = P3*matval(7, 7, otaz, dtaz)

p3distau = P3*matval(7, 8, otaz, dtaz)

p3distcong = P3*matval(7, 9, otaz, dtaz)

evtimeau = EV*matval(8, 7, otaz, dtaz)

evdistau = EV*matval(8, 8, otaz, dtaz)

evdistcong = EV*matval(8, 9, otaz, dtaz)

nitimeau = ni*matval(9, 7, otaz, dtaz)

nidistau = ni*matval(9, 8, otaz, dtaz)

nidistcong = ni*matval(9, 9, otaz, dtaz)

ELSEIF (mode = 4) ; S2

a1timeau = a1*matval(1, 4, otaz, dtaz)

a1distau = a1*matval(1, 5, otaz, dtaz)

a1distcong = a1*matval(1, 6, otaz, dtaz)

a2timeau = a2*matval(2, 4, otaz, dtaz)

a2distau = a2*matval(2, 5, otaz, dtaz)

a2distcong = a2*matval(2, 6, otaz, dtaz)

a3timeau = a3*matval(3, 4, otaz, dtaz)

a3distau = a3*matval(3, 5, otaz, dtaz)

a3distcong = a3*matval(3, 6, otaz, dtaz)

mdtimeau = MD*matval(4, 4, otaz, dtaz)

mddistau = MD*matval(4, 5, otaz, dtaz)

mddistcong = MD*matval(4, 6, otaz, dtaz)

p1timeau = P1*matval(5, 4, otaz, dtaz)

p1distau = P1*matval(5, 5, otaz, dtaz)

p1distcong = P1*matval(5, 6, otaz, dtaz)

p2timeau = P2*matval(6, 4, otaz, dtaz)

p2distau = P2*matval(6, 5, otaz, dtaz)

p2distcong = P2*matval(6, 6, otaz, dtaz)

p3timeau = P3*matval(7, 4, otaz, dtaz)

p3distau = P3*matval(7, 5, otaz, dtaz)

p3distcong = P3*matval(7, 6, otaz, dtaz)

evtimeau = EV*matval(8, 4, otaz, dtaz)

evdistau = EV*matval(8, 5, otaz, dtaz)

evdistcong = EV*matval(8, 6, otaz, dtaz)

nitimeau = ni*matval(9, 4, otaz, dtaz)

nidistau = ni*matval(9, 5, otaz, dtaz)

nidistcong = ni*matval(9, 6, otaz, dtaz)

ELSEIF (mode=3) ; Drive Alone

a1timeau = a1*matval(1, 1, otaz, dtaz)

a1distau = a1*matval(1, 2, otaz, dtaz)

a1distcong = a1*matval(1, 3, otaz, dtaz)

a2timeau = a2*matval(2, 1, otaz, dtaz)

a2distau = a2*matval(2, 2, otaz, dtaz)

a2distcong = a2*matval(2, 3, otaz, dtaz)

a3timeau = a3*matval(3, 1, otaz, dtaz)

a3distau = a3*matval(3, 2, otaz, dtaz)

a3distcong = a3*matval(3, 3, otaz, dtaz)

mdtimeau = MD*matval(4, 1, otaz, dtaz)

mddistau = MD*matval(4, 2, otaz, dtaz)

mddistcong = MD*matval(4, 3, otaz, dtaz)

p1timeau = P1*matval(5, 1, otaz, dtaz)

p1distau = P1*matval(5, 2, otaz, dtaz)

p1distcong = P1*matval(5, 3, otaz, dtaz)

p2timeau = P2*matval(6, 1, otaz, dtaz)

p2distau = P2*matval(6, 2, otaz, dtaz)

p2distcong = P2*matval(6, 3, otaz, dtaz)

p3timeau = P3*matval(7, 1, otaz, dtaz)

p3distau = P3*matval(7, 2, otaz, dtaz)

p3distcong = P3*matval(7, 3, otaz, dtaz)

evtimeau = EV*matval(8, 1, otaz, dtaz)

evdistau = EV*matval(8, 2, otaz, dtaz)

evdistcong = EV*matval(8, 3, otaz, dtaz)

nitimeau = ni*matval(9, 1, otaz, dtaz)

nidistau = ni*matval(9, 2, otaz, dtaz)

nidistcong = ni*matval(9, 3, otaz, dtaz)

ELSEIF (mode=9) ; TNC (4/30/2018 - using drive-alone skims for now, but this should be changed in future to reflect shared TNCs)

a1timeau = a1*matval(1, 1, otaz, dtaz)

a1distau = a1*matval(1, 2, otaz, dtaz)

a1distcong = a1*matval(1, 3, otaz, dtaz)

a2timeau = a2*matval(2, 1, otaz, dtaz)

a2distau = a2*matval(2, 2, otaz, dtaz)

a2distcong = a2*matval(2, 3, otaz, dtaz)

a3timeau = a3*matval(3, 1, otaz, dtaz)

a3distau = a3*matval(3, 2, otaz, dtaz)

a3distcong = a3*matval(3, 3, otaz, dtaz)

mdtimeau = MD*matval(4, 1, otaz, dtaz)

mddistau = MD*matval(4, 2, otaz, dtaz)

mddistcong = MD*matval(4, 3, otaz, dtaz)

p1timeau = P1*matval(5, 1, otaz, dtaz)

```
   p1distau   = P1*matval(5, 2, otaz, dtaz)
   p1distcong   = P1*matval(5, 3, otaz, dtaz)
    p2timeau   = P2*matval(6, 1, otaz, dtaz)
   p2distau   = P2*matval(6, 2, otaz, dtaz)
   p2distcong   = P2*matval(6, 3, otaz, dtaz)
    p3timeau   = P3*matval(7, 1, otaz, dtaz)
   p3distau   = P3*matval(7, 2, otaz, dtaz)
   p3distcong   = P3*matval(7, 3, otaz, dtaz)

   evtimeau   = EV*matval(8, 1, otaz, dtaz)
   evdistau   = EV*matval(8, 2, otaz, dtaz)
   evdistcong   = EV*matval(8, 3, otaz, dtaz)

   nitimeau   = ni*matval(9, 1, otaz, dtaz)
   nidistau   = ni*matval(9, 2, otaz, dtaz)
   nidistcong   = ni*matval(9, 3, otaz, dtaz)
ENDIF
IF (mode=1,2,6,7,8)   ;if mode is not a car mode, then auto time and congested distance all = 0
   timeau   = 0
   distau   = 0
   distcong = 0
   print file=.\_trip_1_1.csv, list=
trip_id(20.0),',',tour_id(12.0),',',hhno(10.0),',',pno(2.0),',',day(2.0),',',tour(2.0),',',half(2.0),',',ts
eg(2.0),',',tsvid(2.0),',',opurp(2.0),',',dpurp(2.0),',',oadtyp(2.0),',',dadtyp(2.0),',',opcl(10.0),',',ot
az(10.0),',',dpcl(10.0),',',dtaz(10.0),',',mode(2.0),',',pathtype(2.0),',',dorp(2.0),',',deptm(10.0),',
',arrtm(10.0),',',endacttm(10.0),',',travtime(10.2),',',travcost(10.2),',',travdist(10.2),',',vot(8.2),
',',trexpfac(2.0),',',timeau(10.2),',',distau(10.2),',',distcong(10.2)
```

```
ELSEIF (mode=3,4,5,9) ;if mode is a car mode, then get total times, distance, congested distance as follows:


timeau=a1timeau+a2timeau+a3timeau+mdtimeau+p1timeau+p2timeau+p3timeau+evtimeau+nitimeau ;sum the skim-based times


distau=a1distau+a2distau+a3distau+mddistau+p1distau+p2distau+p3distau+evdistau+nidistau ;sum the skim-based total distances


distcong=a1distcong+a2distcong+a3distcong+mddistcong+p1distcong+p2distcong+p3distcong+evdistcong+nidistcong ;sum the skim-based congested distances

  print file=.\_trip_1_1.csv, list=
trip_id(20.0),',',tour_id(12.0),',',hhno(10.0),',',pno(2.0),',',day(2.0),',',tour(2.0),',',half(2.0),',',tseg(2.0),',',tsvid(2.0),',',opurp(2.0),',',dpurp(2.0),',',oadtyp(2.0),',',dadtyp(2.0),',',opcl(10.0),',',otaz(10.0),',',dpcl(10.0),',',dtaz(10.0),',',mode(2.0),',',pathtype(2.0),',',dorp(2.0),',',deptm(10.0),',',arrtm(10.0),',',endacttm(10.0),',',travtime(10.2),',',travcost(10.2),',',travdist(10.2),',',vot(8.2),',',trexpfac(2.0),',',timeau(10.2),',',distau(10.2),',',distcong(10.2)


ENDIF


ENDIF ; reco


ENDRUN


;-------------------------------------------------------------------------------

;-------------------------------------------------------------------------------
```

Appendix B Cube Voyager Script – IXXI trips

```
;-----------------------------------------------------------------------------------------

RUN PGM=MATRIX  MSG='convert parcel txt into dbf'
```

```
FILEI RECI = ?_raw_parcel.txt,delimiter[1]=','


FILEO reco[1] = ?_raw_parcel.dbf,

fields=parcelid(12.0),taz(5.0),hh_p(10.5),empfoo_p(10.5),empret_p(10.5),empsvc_p(10.5),em
ptot_p(10.5)


if (reci.recno=1)
   loop f=1, reci.numfields
      if (reci.cfield[f]='parcelid')      f_parcelid      = f
      if (reci.cfield[f]='taz_p')  f_taz_p  = f
      if (reci.cfield[f]='hh_p')  f_hh_p  = f
      if (reci.cfield[f]='empfoo_p')    f_empfoo_p      = f
      if (reci.cfield[f]='empret_p')     f_empret_p      = f
      if (reci.cfield[f]='empsvc_p')     f_empsvc_P     = f
      if (reci.cfield[f]='emptot_p')     f_emptot_p    = f
   endloop


ELSE
      ro.parcelid  = val(reci.cfield[f_parcelid])
      ro.taz  = val(reci.cfield[f_taz_p])
      ro.hh_p  = val(reci.cfield[f_hh_p])
      ro.empfoo_p  = val(reci.cfield[f_empfoo_p])
      ro.empret_p  = val(reci.cfield[f_empret_p])
      ro.empsvc_p   = val(reci.cfield[f_empsvc_p])
      ro.emptot_p  = val(reci.cfield[f_emptot_P])


   WRITE reco=1
```

```
ENDIF

ENDRUN


;=====================================================================

run pgm=matrix  msg='compile ix+xi person and vehicle trips, miles, c-miles'
; compile ix+xi (basically, any trip from the external matrix for regional indicators
;
; Input files:
filei mati[1]="trips.external.mat"
     mati[2]="tempskh07.mat"
     mati[3]="tempskmd5.mat"
     mati[4]="tempskh16.mat"
     mati[5]="tempskev2.mat"
; main output is a matrix w/ all variables combined for calculations...
mato[1]="ixxi_temp.mat", mo=5-13 name=x_vt,x_vht,x_vmt,x_cvmt,hhs,emptot,food,ret,svc
; initialize p-a person trips by purpose
;
zdati[2] = ?_raw_parcel.dbf,
sum=hh_p,
   EMPFOO_P,
   EMPRET_P,
   EMPSVC_P,
   EMPTOT_P
```

```
;
jloop
;
mw[1]=mi.1.xwk
mw[2]=mi.1.xpb
mw[3]=mi.1.xsh
mw[4]=mi.1.xsr
; transposes
mw[28] = mi.1.xwk.t
mw[23] = mi.1.xpb.t
mw[24] = mi.1.xsh.t
mw[25] = mi.1.xsr.t



; A3
; first drive alone
;       share       tot_pa       tot_ap
x_da_a3 = 0.890*(mi.1.xwk*0.295 + mw[28]*0.018)
x_da_a3 = 0.540*(mi.1.xpb*0.088 + mw[23]*0.037) + x_da_a3
x_da_a3 = 0.450*(mi.1.xsh*0.028 + mw[24]*0.027) + x_da_a3
x_da_a3 = 0.290*(mi.1.xsr*0.060 + mw[25]*0.029) + x_da_a3
;
; next shared ride--s2 and s3+ combined...
;       share       tot_pa       tot_ap  vocc
x_sr_a3 = 0.110*(mi.1.xwk*0.295 + mw[28]*0.018)/2.34
x_sr_a3 = 0.460*(mi.1.xpb*0.088 + mw[23]*0.037)/2.55 + x_sr_a3
x_sr_a3 = 0.550*(mi.1.xsh*0.028 + mw[24]*0.027)/2.41 + x_sr_a3
```

```
x_sr_a3 = 0.710*(mi.1.xsr*0.060 + mw[25]*0.029)/2.85 + x_sr_a3
;
; MD
; first drive alone
;       share       tot_pa      tot_ap
x_da_md = 0.890*(mi.1.xwk*0.101 + mw[28]*0.098)
x_da_md = 0.540*(mi.1.xpb*0.264 + mw[23]*0.226) + x_da_md
x_da_md = 0.450*(mi.1.xsh*0.231 + mw[24]*0.217) + x_da_md
x_da_md = 0.290*(mi.1.xsr*0.173 + mw[25]*0.149) + x_da_md
;
; next shared ride--s2 and s3+ combined...
;       share       tot_pa      tot_ap  vocc
x_sr_md = 0.110*(mi.1.xwk*0.101 + mw[28]*0.098)/2.34
x_sr_md = 0.460*(mi.1.xpb*0.264 + mw[23]*0.226)/2.55 + x_sr_md
x_sr_md = 0.550*(mi.1.xsh*0.231 + mw[24]*0.217)/2.41 + x_sr_md
x_sr_md = 0.710*(mi.1.xsr*0.173 + mw[25]*0.149)/2.85 + x_sr_md
;
; P3
; first drive alone
;       share       tot_pa      tot_ap
x_da_p3 = 0.890*(mi.1.xwk*0.035 + mw[28]*0.300)
x_da_p3 = 0.540*(mi.1.xpb*0.112 + mw[23]*0.165) + x_da_p3
x_da_p3 = 0.450*(mi.1.xsh*0.181 + mw[24]*0.178) + x_da_p3
x_da_p3 = 0.290*(mi.1.xsr*0.147 + mw[25]*0.117) + x_da_p3
;
; next shared ride--s2 and s3+ combined...
;       share       tot_pa      tot_ap  vocc
```

```
x_sr_p3 = 0.110*(mi.1.xwk*0.035 + mw[28]*0.300)/2.34
x_sr_p3 = 0.460*(mi.1.xpb*0.112 + mw[23]*0.165)/2.55 + x_sr_p3
x_sr_p3 = 0.550*(mi.1.xsh*0.181 + mw[24]*0.178)/2.41 + x_sr_p3
x_sr_p3 = 0.710*(mi.1.xsr*0.147 + mw[25]*0.117)/2.85 + x_sr_p3
;
; EV
; first drive alone
;       share       tot_pa      tot_ap
x_da_ev = 0.890*(mi.1.xwk*0.069 + mw[28]*0.084)
x_da_ev = 0.540*(mi.1.xpb*0.036 + mw[23]*0.072) + x_da_ev
x_da_ev = 0.450*(mi.1.xsh*0.060 + mw[24]*0.078) + x_da_ev
x_da_ev = 0.290*(mi.1.xsr*0.120 + mw[25]*0.205) + x_da_ev
;
; next shared ride--s2 and s3+ combined...
;       share       tot_pa      tot_ap  vocc
x_sr_ev = 0.110*(mi.1.xwk*0.069 + mw[28]*0.084)/2.34
x_sr_ev = 0.460*(mi.1.xpb*0.036 + mw[23]*0.072)/2.55
x_sr_ev = 0.550*(mi.1.xsh*0.060 + mw[24]*0.078)/2.41
x_sr_ev = 0.710*(mi.1.xsr*0.120 + mw[25]*0.205)/2.85

;order of calcs as follows:  vts; vht; vmt; cvmt

mw[5]=x_da_a3+x_da_md+x_da_p3+x_da_ev+x_sr_a3+x_sr_md+x_sr_p3+x_sr_ev
mw[6]=(x_da_a3*mi.2.da_time+x_da_md*mi.3.da_time+x_da_p3*mi.4.da_time+x_da_ev*mi
.5.da_time+x_sr_a3*(mi.2.s2_time+mi.2.s3_time)*0.5+x_sr_md*mi.3.da_time+x_sr_p3*(mi.4
.s2_time+mi.4.s3_time)*0.5+x_sr_ev*mi.5.da_time)/60.0
```

```
mw[7]=x_da_a3*mi.2.da_dist+x_da_md*mi.3.da_dist+x_da_p3*mi.4.da_dist+x_da_ev*mi.5.da_dist+x_sr_a3*(mi.2.s2_dist+mi.2.s3_dist)*0.5+x_sr_md*mi.3.da_dist+x_sr_p3*(mi.4.s2_dist+mi.4.s3_dist)*0.5+x_sr_ev*mi.5.da_dist

mw[8]=x_da_a3*mi.2.da_cdist+x_da_md*mi.3.da_cdist+x_da_p3*mi.4.da_cdist+x_da_ev*mi.5.da_cdist+x_sr_a3*(mi.2.s2_cdist+mi.2.s3_cdist)*0.5+x_sr_md*mi.3.da_cdist+x_sr_p3*(mi.4.s2_cdist+mi.4.s3_cdist)*0.5+x_sr_ev*mi.5.da_cdist

;

;mw[9]  = zi.2.housesp / (1+zi.2.housesp + 1.1*(zi.2.emptot_p - zi.2.empfoodp - zi.2.empret_p - 0.25*zi.2.empsvc_p)) ;res share of ixxi

mw[9] = zi.2.hh_p

mw[10] = zi.2.emptot_p

mw[11] = zi.2.empfoo_p

mw[12] = zi.2.empret_p

mw[13] = zi.2.empsvc_p


endjloop

;

endrun


run pgm=matrix

filei mati[1]=trips.cv.mat

    mati[2]="tempskh07.mat"

    mati[3]="tempskmd5.mat"

    mati[4]="tempskh16.mat"

    mati[5]="tempskev2.mat"

FILEO MATO=cv_temp.mat, MO=1-4

;

jloop

;
```

mw[1]=mi.2.da_dist

mw[2]=mi.3.da_dist

mw[3]=mi.4.da_dist

mw[4]=mi.5.da_dist

mw[5]=mi.2.da_time/60.0

mw[6]=mi.3.da_time/60.0

mw[7]=mi.4.da_time/60.0

mw[8]=mi.5.da_time/60.0

mw[9]=mi.2.da_cdist

mw[10]=mi.3.da_cdist

mw[11]=mi.4.da_cdist

mw[12]=mi.5.da_cdist

mw[13]=mi.1.cv2x

mw[14]=mi.1.cv3x

;

;A3

; calcs 1 and 2=c2vmt,c3vmt; 3 and 4=c2vht, c3vht; 5 and 6 = c2cvmt, c3cvmt

mw[15]=mw[13]*0.224*mw[5] + mw[13]*0.407*mw[6] + mw[13]*0.136*mw[7] + mw[13]*0.208*mw[8]

mw[16]=mw[14]*0.287*mw[5] + mw[14]*0.319*mw[6] + mw[14]*0.089*mw[7] + mw[14]*0.279*mw[8]

;

mw[17]=mw[13]*0.224*mw[1] + mw[13]*0.407*mw[2] + mw[13]*0.136*mw[3] + mw[13]*0.208*mw[4]

mw[18]=mw[14]*0.287*mw[1] + mw[14]*0.319*mw[2] + mw[14]*0.089*mw[3] + mw[14]*0.279*mw[4]

;

mw[19]=mw[13]*0.224*mw[9] + mw[13]*0.407*mw[10] + mw[13]*0.136*mw[11] + mw[13]*0.208*mw[12]

```
mw[20]=mw[14]*0.287*mw[9] + mw[14]*0.319*mw[10] + mw[14]*0.089*mw[11] +
mw[14]*0.279*mw[12]

;

endjloop

;

fileo reco[1]=cveh_taz.dbf,
fields=i,c2_vt_i,c3_vt_i,c2_vht_i,c3_vht_i,c2_vmt_i,c3_vmt_i,c2_cvmt_i,c3_cvmt_i

ro.i=i

c2_vt_i=rowsum(13)

c3_vt_i=rowsum(14)

c2_vht_i=rowsum(15)

c3_vht_i=rowsum(16)

c2_vmt_i=rowsum(17)

c3_vmt_i=rowsum(18)

c2_cvmt_i=rowsum(19)

c3_cvmt_i=rowsum(20)

write reco=1

;

endrun

;



;run pgm=matrix

;filei mati[1]=temp.mat

;mw[1]=mi.1.1

;mw[2]=mi.1.2

;mw[3]=mi.1.3

;mw[4]=mi.1.4
```

```
;endrun



run pgm=matrix  msg='compile ix+xi person and vehicle trips, miles, c-miles'

; compile ix+xi (basically, any trip from the external matrix for regional indicators

; Input files:

filei mati[1]="ixxi_temp.mat"


; also output a rowsum file in dbf

jloop

mw[1]=mi.1.x_vt

mw[2]=mi.1.x_vht

mw[3]=mi.1.x_vmt

mw[4]=mi.1.x_cvmt

mw[5]=mi.1.x_vt.t

mw[6]=mi.1.x_vht.t

mw[7]=mi.1.x_vmt.t

mw[8]=mi.1.x_cvmt.t

mw[9]=mi.1.hhs

mw[10]=mi.1.emptot

mw[11]=mi.1.food

mw[12]=mi.1.ret

mw[13]=mi.1.svc

;

endjloop

;
```

```
fileo reco[1]=ixxi_taz.dbf,
fields=i,ix_vt_i,ix_vt_j,ix_vht_i,ix_vht_j,ix_vmt_i,ix_vmt_j,ix_cvmt_i,ix_cvmt_j,hhs,emptot,food,ret,svc

ro.i=i

ix_vt_i=rowsum(1)

ix_vt_j=rowsum(5)

ix_vht_i=rowsum(2)

ix_vht_j=rowsum(6)

ix_vmt_i=rowsum(3)

ix_vmt_j=rowsum(7)

ix_cvmt_i=rowsum(4)

ix_cvmt_j=rowsum(8)

hhs=rowave(9)

emptot=rowave(10)

food=rowave(11)

ret=rowave(12)

svc=rowave(13)

write reco=1

;

endrun

------------------------------------------------------------
```