Lab 2: Trip Production

Hayden Atchley, Tristan Parker, Steven Goodsell, Austin Nichalson

9/27/2021

Trip Production Rates

Many of the original trip production rates provided in the Cube scenario are set at 1 across the board, regardless of purpose, number of vehicles, etc. This is obviously not realistic, so we need to use different data. We used the data from the NHTS 2017 report, which we worked with in HW #2 as well. The data manipulation required to get these values is very similar to that from HW #2, so we present just the results here:

HBW

wrkcount	0 vehicles	1 vehicles	2 vehicles	3 vehicles
0	0.007	0.013	0.058	0.032
1	0.500	0.996	1.224	1.040
2	3.786	2.068	2.299	2.588

HBSHOP

hhsize	0 vehicles	1 vehicles	2 vehicles	3 vehicles
1	0.601	0.874	0.826	0.887
2	2.101	1.748	1.781	1.771
3	1.986	1.678	1.303	1.838
4	1.580	1.820	1.840	1.657

HBO

hhsize	0 vehicles	1 vehicles	2 vehicles	3 vehicles
1	0.414	0.558	0.540	0.339
2	0.420	1.549	1.153	1.123
3	3.980	1.895	2.039	1.931
4	3.318	5.282	4.306	4.195

We also calculated the 95% confidence intervals for each trip type, in case we need to adjust them later:

```
#For HBW
tripprod %>%
  group_by(wrkcount, hhvehcnt) %>%
  summarize(
    n = n(),
    HBWsd = wtd.sd(HBW, wthhfin)
   ) %>%
  mutate(
    conf = 1.96 * HBWsd / n^2,
    conf_sci = ifelse(conf < 0.001, scientific(conf, digits = 3), round(conf, digits = 3)),
    hhvehcnt = paste(hhvehcnt, "vehicles")
    ) %>%
  pivot_wider(id_cols = wrkcount, names_from = hhvehcnt, values_from = conf_sci) %>%
  my_flextable() %>% #for formatting
  add_header_lines("HBW Confidence Intervals")
```

HBW Confidence Intervals

wrkcount	0 vehicles	1 vehicles	2 vehicles	3 vehicles
0	1.54e-06	9.64e-08	4.49e-07	1.82e-06
1	2.55e-04	9.28e-07	1.19e-06	3.19e-06
2	0.062	1.35e-04	1.62e-06	1.92e-06

(The code is virtually the same for the other purposes)

HBSHOP Confidence Intervals

hhsize	0 vehicles	1 vehicles	2 vehicles	3 vehicles
1	1.75e-05	4.11e-07	7.32e-06	5.72e-05
2	0.001	6.78e-06	6.19e-07	2.17e-06
3	0.006	1.58e-04	1.84e-05	1.19e-05
4	0.039	3.55e-04	1.69e-05	1.32e-05

HBO Confidence Intervals

hhsize	0 vehicles	1 vehicles	2 vehicles	3 vehicles
1	1.44e-05	3.71e-07	6.30e-06	3.32e-05
2	5.99e-04	5.65e-06	5.22e-07	1.85e-06
3	0.007	1.59e-04	2.51e-05	1.25e-05
4	0.036	7.40e-04	3.04e-05	2.13e-05

Note that there are some data points that seem to not make sense, for example a 3-person, 0-vehicle household makes more HBO and HBSHOP trips than a 4-person, 0-vehicle household, which should not be the case. We had to adjust some of these trip rates to account for that intuition. Our new tables are presented below:

HBW

wrkcount	0 vehicles	1 vehicles	2 vehicles	3 vehicles
0	0.000	0.000	0.000	0.000
1	0.500	0.996	1.224	1.040
2	3.786	2.068	2.299	2.588

HBSHOP

hhsize	0 vehicles	1 vehicles	2 vehicles	3 vehicles
1	0.601	0.874	0.874	0.874
2	2.000	1.748	1.781	1.781
3	2.000	1.800	1.800	1.838
4	2.000	1.820	1.840	1.840

HBO

hhsize	0 vehicles	1 vehicles	2 vehicles	3 vehicles
1	0.414	0.558	0.558	0.558
2	0.420	1.549	1.153	1.123
3	3.000	1.895	2.039	1.931
4	3.318	5.282	4.306	4.195

We then wrote these tables to their respective .dbf files and ran the Trip Generation submodel in Cube. Ideally, the total output trips would be somewhat close to the NHTS reported total weighted trips for the area:

Purpose	**Weighted Survey Trips**
HBW	118,653
HBO	267,987
HBSHOP	129,614

After our first run, these are the results we got:

```
read.dbf("dbf/HH_PROD1.dbf") %>% round(digits = 2) %>% head()
```

```
##   TAZ    HBWP  NHBWP    HBOP  HBSCP   HBSHP   NHBOP
## 1   1  901.97 421.88  979.64 179.57 1126.08 1497.29
## 2   2  173.25  80.48  287.14  85.19  245.85  338.41
## 3   3  388.73 180.28  499.14 110.82  521.62  672.83
## 4   4 1604.19 740.39 1800.62 498.94 1652.08 2312.72
## 5   5  784.69 360.54 1055.01 369.49  772.97 1148.25
## 6   6  441.61 203.07  516.47 155.99  440.94  626.11
```

Adding up each trip by purpose gives us:

```
read.dbf("dbf/HH_PROD1.dbf")[,c(2,4,6)] %>% colSums() %>% round(digits = 0)
```

```
##   HBWP   HBOP  HBSHP
## 135792 176210 170254
```

These are obviously not very close to the targets, so we need to do some adjusting. We began by adjusting the rates with the largest confidence intervals first, and these are the results we got: # `{r hhprod2} # read.dbf("dbf/HH_PROD2.dbf") %>% round(digits = 2) %>% head() # read.dbf("dbf/HH_PROD2.dbf")[,-1] %>% colSums() %>% round(digits = 0) #`