|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |
|  | | |

|  |  |
| --- | --- |
| 180 Battery Street, Suite 350 Burlington, VT 05401 802.383.0118 **www.rsginc.com** | **prepared for:** |
| Town of Williston | CCRPC |
| **Submitted By:** |
| rsg |
|  |
|  |

[1.0 Introduction 1](#_Toc140350856)

[1.1 Background 1](#_Toc140350857)

[1.2 How To Use This Guide 1](#_Toc140350858)

[2.0 Using SPA 3](#_Toc140350859)

[2.1 Background Software 3](#_Toc140350860)

[2.2 Download the SPA code 3](#_Toc140350861)

[2.3 Input Files 4](#_Toc140350862)

[Generators 4](#_Toc140350863)

[Parking 5](#_Toc140350864)

[Parking Demand and Adjustments3.xlsx 5](#_Toc140350865)

[Restrict List 6](#_Toc140350866)

[Configuration.YAML 6](#_Toc140350867)

[2.4 Installation of the SPA Tool 6](#_Toc140350868)

[2.5 Running the Tool 8](#_Toc140350869)

[3.0 Output Data and Results 9](#_Toc140350880)

[3.1 CSV outputs 9](#_Toc140350901)

[3.2 Post-processing 9](#_Toc140350902)

[3.3 post-processing output 12](#_Toc140350903)

[4.0 Calibration and Refining Input Data 16](#_Toc140350904)

[5.0 Creating Input Files 17](#_Toc140350906)

[5.1 Create Parking Lot Preference List 17](#_Toc140350907)

[5.2 Create Generator and Parking Lots Input Files 19](#_Toc140350908)

[Generator File 19](#_Toc140350909)

[Parking Lots File 20](#_Toc140350910)

[5.3 Create Adjustment Factors File 21](#_Toc140350911)

**List of Figures**

[Figure 1: Anaconda Prompt 7](#_Toc140350912)

[Figure 2: Conda Environment Install 7](#_Toc140350913)

[Figure 3: Conda Python Environment (after) 8](#_Toc140350914)

[Figure 19: Employment and Parking lot data displayed with only the Home Depot and Walmart locations and associated parking lots highlighted 17](#_Toc140350915)

[Figure 20: Open the Generate Parking Preference Tool 18](#_Toc140350916)

[Figure 21: Use the pulldown menu when SELECTING data 18](#_Toc140350917)

[Figure 22: Original data grouped as "Original Data" 18](#_Toc140350918)

[Figure 23: Generator and parking data created by the Generate Parking Preference tool 19](#_Toc140350919)

[Figure 24: Parking lot preferences by generator 19](#_Toc140350920)

# Introduction

## Background

This user guide provides instructions for the Shared Parking Analysis (SPA) tool developed by the Chittenden County Regional Planning Commission (CCRPC) with support from RSG. The SPA tool models shared parking demand for specific geographic areas based on land use and adjustment factors by month, day of week, and time of day.

The SPA uses parking demand factors from Shared Parking, Second Edition by the Urban Land Institute*[[1]](#footnote-2)* (ULI) are incorporated by default, although the tool is flexible to accept user specific inputs along with other parking demand factors.

The SPA uses open-source software in both Python and R software languages complemented by any GIS program that can generate a Shapefile with latitude and longitude location data for both the source of parking demand (i.e., land uses) and the parking lots (parking supply).

The SPA is a unique and powerful tool that expands the methodology developed by the ULI. Rather than the typical ULI Shared Parking analysis as a many demand to one lot analysis, The SPA tool evaluates multiple parking demands across multiple possible parking lots.

First it calculates parking demand for each parking generator for all time combinations. The combination dimensions from Shared Parkingare:

* Time of Day: 6 AM – Midnight
* Day: Weekday or Weekend
* Time of Year: all 12 months and December after Christmas (termed “Late December”)

The user assigns each generator a list of parking lots in order of preference, and then SPA allocates each generator’s parking demand. If a generator’s first choice lot is already full, its demand is moved to its second choice. The model continues to allocate demand until all demands are met or no parking spots are available.

The tool creates a csv file of parking lot utilization for all factor combinations and a report noting the time of highest parking use and any times where there was insufficient parking for a particular generator.

## How To Use This Guide

This guide sets out the process for developing the inputs for the SPA, installing and running the SPA tool, and using the post process visualization summary in Excel.

To run the SPA tool:

* Install the tool
* Create the *Parking Lots* input file
* Create the *Generators* input file
* Create a *Restrict* lot list
* Create shared parking demand and adjustment factors, or use the default data supplied with the tool
* Run the tool
* Check the outputs for reasonableness

This guide provides chapters on:

* Preparing SPA inputs
* Running the SPA tool
* Analyzing the results
* Testing the effect of adding a new generator to an existing population of parking lots and generators
* Refining the input data with real world observations

File names are in *italics*.

Tab names in an Excel file are in **bold**.

# Using SPA

This chapter sets out the process for installing the necessary software to run the SPA tool.

## Background Software

Several pieces of software are necessary to develop the inputs and run the SPA tool. These include the following:

* **conda via**[**Anaconda**](https://www.anaconda.com/products/individual)**or [Miniconda](https://docs.conda.io/en/latest/miniconda.html)**: Conda is an open-source package management and environment management system that simplifies the installation and management of software packages and dependencies across multiple programming languages. It allows users to create isolated environments to run different projects with specific package versions and configurations.
* **R**. R is a free and open-source programming language and software environment primarily used for statistical computing and graphics. It provides a wide range of statistical and graphical techniques and has a large community of users and developers contributing to its extensive collection of packages.
* **R Studio**. RStudio is an integrated development environment (IDE) for the R programming language. It provides a user-friendly interface and various tools to enhance productivity, making it easier to write, run, and debug R code while offering features like code editing, data visualization, and package management.
* **GIS (qGIS, ArcGIS, ArcMap)**. GIS software (Geographic Information System) is a type of software designed for capturing, analyzing, managing, and presenting geographic data. It enables users to visualize spatial information, perform spatial analysis, create maps, and make informed decisions based on geographical data. QGIS is an open source GIS program while ArcGIS and ArcMap are proprietary ESRI products.

## Download the SPA code

The SPA Source code is stored in a GitHub repository. In order to use the SPA tool the user is required to download the code by cloning the repository or downloading the zip file of the repository from GitHub.

[ pictures of the GitHub repo screen shot and code ]

The SPA ‘Shared\_Parking’ folder should be moved to the location on the local machine where it will be called. Typically, a location near to the root C: drive or in a My Documents location is used.

## Input Files

The next chapter sets out the process for setting up the inputs. The model input files are comprised of three components:

* Generators: the land uses that generate the demand for parking
* Parking: the supply of parking spaces
* Demand and Adjustment file: this is the ULI rates of parking demand by land use.

### Generators

The generator data are shapefiles for any scenarios to be tested in the SPA tool. Any generator shapefile has the following data fields:

* Name – Name of the parking generator. This column is not used in the tool and is to help the user identify the generator.
* Address – Physical address of the land use
* Gen\_ID – Unique ID associated with each generator. **This is a mandatory field.**
* Long & Lat – are the longitude and latitude of the land use.
* NAICS – code for distinguishing the land use that generates parking demand
* NAICS Desc – the description for the NAICS code
* Land Use – general description of the land use, either zoning, or some other local definition.
* LUC – Land Use Code of a particular generator’s land use. This number must match a land use code in the *Land Use Demand* file and *Adjustment Factors* file. **This is a mandatory field.**
* Type – Description of the type of land use. Similar, if not identical to, the Land Use field.
* Unit – The unit in which size is measured. The unit type must match the Land Use’s unit type in the *Land Use Demand* file.
* Size – The size of the land use in the units specified.
* DUType – SF single family & MF multifamily
* DUCount – the number of households living in the structure.

### Parking

| **WinCity\_Aug21\_parking** | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **LOT\_NAME** | **SPACE\_TOT** | **LOT\_UID** | **LOT\_LON** | **LOT\_LAT** | **RESTRICT** | **LOT\_GEN\_ID** | **Shape\_Leng** | **Shape\_Area** | **NOTES** | **Category** | **Street** |
| Lapointe Street | 21 | 1 | -73.1746875862 | 44.4972150788 | 0 | 0 | 407.050748091 | 611.072401597 |  | On Street | Lapointe St |

The generator data are shapefiles for any scenarios to be tested in the SPA tool. Any generator shapefile has the following data fields:

* Name – Name of the parking location.
* Space\_Tot – the number of parking spaces in the parking lot
* Lot\_UID - Unique ID associated with each generator. **This is a mandatory field.**
* Long & Lat – are the longitude and latitude of the parking supply.
* Restrict – this code specifies who can park in the parking spaces.
* Lot\_Gen\_ID: if there is a specific land use that it is connected with.
* Shape\_length – information on the parking polygon
* Shape\_area – information on the parking polygon
* Category – on-street, off-street, residential, etc..
* Street – the name of the street it is most closely associated with.

### Parking Demand and Adjustments3.xlsx

This Excel file contains parking demand ratios for the land uses in the *Generators* folder. These ratios and their corresponding adjustment factors can come from any source. Currently, the SPA tool uses demand factors and adjustments from the second edition of Shared Parking, but different data can be used as long as it maintains the same format.

* LUC – Land Use Codes. These numbers can be arbitrary, but they must match the land use codes in the *Generators* file (LUC column).
* Land Use – Description of the type of land use. This column is not used in the tool.
* User – Either “Visitor/Customer” or “Employee.” These two types of users have different parking demand ratios. In the case of housing, residents’ parking demand appears under “Employee.”
* Weekday – Weekday demand ratio, the number of parking spaces per unit required at peak weekday times.
* Weekend – Weekend demand ratio, the number of parking spaces per unit required at peak weekend times.
* Unit – The units which correspond to the demand ratios.

### Restrict List

The restrict list is a file designed to limit the land uses who are allowed to park in specific parking areas.

### Configuration.YAML

Now that all the inputs have been entered there is one more step prior to running the tool. The ‘Configuration.yaml’ file includes the necessary link between the inputs and outputs as well as a handful of variables. The file includes information on months to analyze, days of the week, hours of the day, etc.

* data\_dir: data (this is the folder where the data inputs are stored)
* output\_dir: baseoutput (this is the name of the folder where the output will go)
* factors\_file: **'Parking Demand and Adjustments3.xlsx'**

The file also needs to be filled in with the file names for the Generator shapefile and the Parking shapefile.

* demand\_shapefile: **WinCity\_Calibratedbased.shp**
* supply\_shapefile: **WinCity\_BaseCalibration\_Supply.shp**

Other data in the file shouldn’t need to be altered, but the YAML does provide flexibility on other key model parameters and inputs.

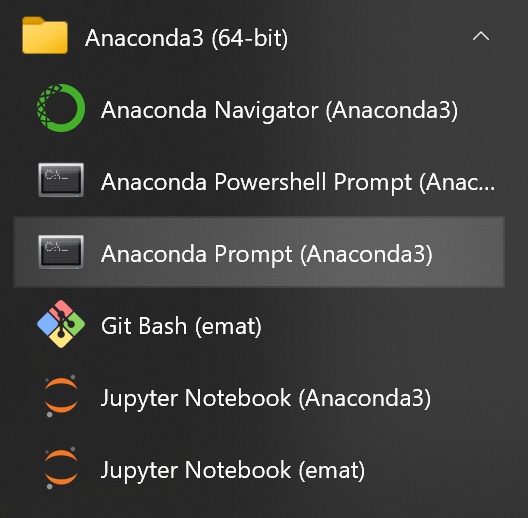
Choose the Output Folder in the “Output File Folder” field and change the Output File Name if desired.

## Installation of the SPA Tool

The installation of the SPA tool requires the use of setting up a python environment using the Anaconda python interface.

Use the Start menu to find the Anaconda Prompt (see Figure 1). A command window opens. Using commands you need to navigate to the shared\_parking folder.

Figure : Anaconda Prompt



For an example stored on the C drive in a GitHub folder

* cd c:\GitHub\shared\_parking

Once the command window is setup to point to the shared\_parking folder, install the Shared\_Parking environment.

* conda env create -f environment.yml

Figure : Conda Environment Install

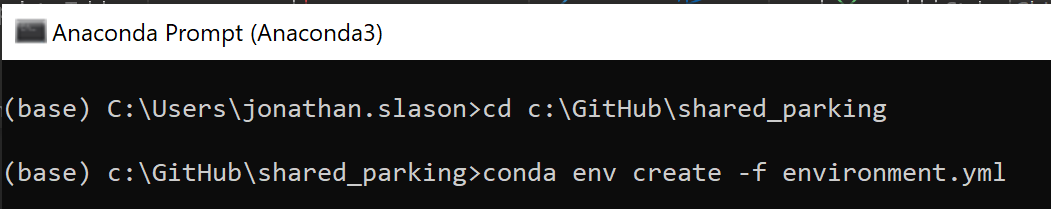
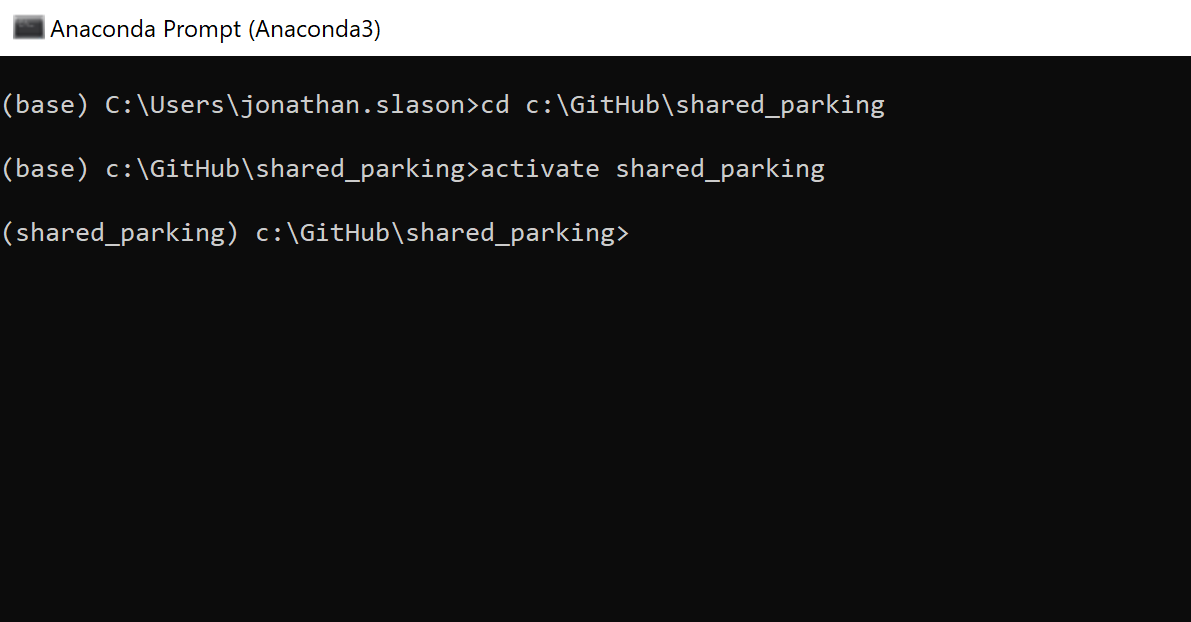


Figure : Conda Python Environment (after)



## Running the Tool

Once the conda environment has been activated then the tool is ready for running. Enter the following command to run the tool:

* Run python run\_model.py -c tests/winooski\_example/configuration.yaml to use the example configuration. This will generate output files for use in the post-processing R scripts.

# Output Data and Results

## CSV outputs

The Shared Parking Analysis Tool’s Post-Processing R Scripts are designed to extract data from the CSV output files are created after running the Python script. These include: This section describes the format of the output files generated by the SPA:

* Demand: Demand for individual lots.
* Factors: Hourly adjustment factors for weekdays and weekends.
* Gen\_lots: Parking generators with generator and lot IDs.
* Preference: Generates information for parking preferences.
* Timeseries: A timeseries showing utilization by lot.

## Post-processing

Using these scrips requires coordination of directories following running the SPA tool, and includes the following R script files:

* Post Process.R: Master file to run all scripts and generate outputs.
* read\_shapefiles.R: Open’s shapefiles of generated parking lots.
* specific\_gens\_lots.R: Gets demand information for one or more generators or parking lots using generator and lot ID variables.
* constraint check.R: Generates summary space constraints on parking lots and demand.
* counts\_analysis.R: Generates parking counts for on and off-street parking during weekdays and weekends.
* demand\_check.R: Estimates total parking unconstrained demand versus demand in the parking model.

Developing the outputs from the post processing will require only running the Post Process file, while all other R scripts should be in a subfolder titled “Source” in the R project directory. Required libraries will be called in the R script but may require installation before running the script.

Text

Description automatically generated

Figure 4: post process step 1 - load packages

Running the post process will require defining the project type and editing directory selections within the script to correctly call and store files. The projects available for analysis include “Winooski\_city” and “Winooski\_ave.” The directory called from for “model\_dir” should contain the output from Python scripts used in the SPA. The directory called from “dir” should contain a folder titled “Outputs” for storing post processing results.

Text

Description automatically generated

figure 5: post process step 2 - define project & loading inputs

Text

Description automatically generated

figure 6: post process step 2 - define project, setting directories

Step 3 reads data outputs from the Python model and shapefile inputs used for the SPA model run using the “read\_shapefiles.R” script. The folder “Source” must be in the same directory as the R Project in order these scripts.

Text

Description automatically generated

figure 6: post process step 3 – read data

Step 4 analyzes the parking data to project capacity versus demand stored in usable output files. Run the constraint check section to plot demand vs capacity for both the model and unconstrained demand given the SPA outputs.

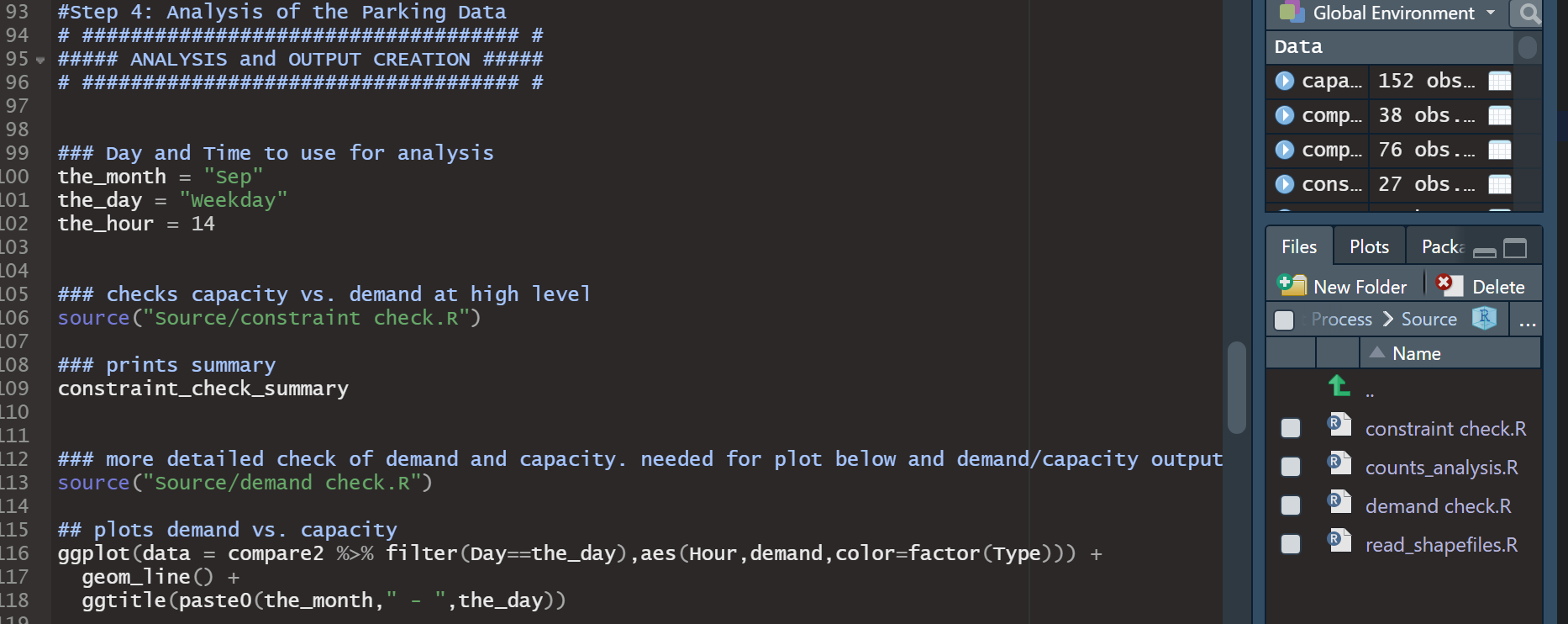


figure 7: post process step 4 – constraint check

Running the script file “counts\_analysis.R” will develop counts for weekday and weekend to project demand by hour, day, and month. Land use codes can be adjusted for on-street shared parking as well.

Text

Description automatically generated

figure 8: post process step 4 – counts analysis

Running the last step will export an Excel output to the “Outputs” folder under the output directory.

Text

Description automatically generated

figure 9: post process step 5 – write output

## post-processing output

Post-processing will store an Excel file titled “visualization” to the chosen directory. The file contains the following tabs:

* “Overall summaries” contains pivot tables showing high-level summaries with changeable filters.
* “OnStreet 3 period” shows on-street parking utilization for hours 8:00, 13:00, and 18:00 by street.
* “Single Street Pivot” shows parking demand and utilization parking over a 1-day period for a given street.
* “parking\_formatted” shows geographic information of parking lots.
* “timeseries” contains raw data for parking utilization by street, time, and land use.

The first table in the “Overall summaries” worksheet shows the following data points for a for a given month, day (weekday vs weekend), and hour:

* The total number of spaces (“Sum of SPACE\_TOT”).
* Total utilized spaces by land use type (“Sum of demand”).

Graphical user interface, table

Description automatically generated

figure 10: overall summaries output 1

In contrast, the second table, “OnStreet 3 Period,” can also be filtered by land use category and is analyzed at a lot-specific level. This table includes the same analyzed variables and percent utilization for each lot (“Sum of PctFull”).

Table

Description automatically generated

figure 11: overall summaries output 2

The table in this worksheet shows utilization rates for each parking lot (“Lot\_UID”) at hours 8:00, 13:00, and 16:00, organized by street. This also shows a grand total utilization rate for a 24-hour period. These can be filtered by month, day, and land use category.

Table

Description automatically generated

figure 12: onstreet 3 period output

This worksheet shows aggregated parking utilization for a given street by hours 0:00 and 6:00-23:00. The table includes:

* Total parking spaces (“Sum of SPACE\_TOT”)
* Utilized spaces/demand (“Sum of demand”)
* Percent utilization (“Sum of PctFull”)

These observations can be filtered by month, day, and street.

Graphical user interface, table

Description automatically generated

figure 13: single street pivot output

The user should check these results for reasonableness. Do they generally agree with the user’s expectation? Do they agree with anecdotal data on this location? If not, the user should double check the inputs and consider if there are special cases in this area that do not conform well with the default demand and adjustment factors.

# Calibration and Refining Input Data

The SPA tool uses a generic dataset of national data from Shared Parking. It should be used as a planning tool to understand the effects of shared parking, both where excess capacity may exist and where a new generator may require more parking than is currently available. Like all planning data, the demand and adjustment factors used here are not perfect, and the user should be careful when demand is shown to be close to supply.

There are a variety of reasons a user may want to change the demand and adjustments factors. A user may decide to use local data for time adjustment factors or use a higher generation rate for a particularly popular generator. The available land use codes may not cover a desired land use type. Shared Parking explains its methods for data collection and how to collect local data.

A good first step is to perform field counts at the times the SPA tool indicates peak demand occurs. It may also be helpful to compare anecdotal data for particular times with what the tool’s output indicates. These observations may show that the tool is generally accurate, or over- or under-estimating peak demand. It is also possible that some stores are not open when the default factors are showing they have demand, e.g. restaurants that are not open after midnight.

If the user determines that the demand and adjustment factors need to be refined, the user should perform parking lot counts in accordance with the Shared Parking methodology. It may be possible to perform counts at only the times of highest demand and adjust the factors accordingly and thus avoid counting all 26 days of factors. Changing factors to reflect store hour hours will also help calibrate a particular area.

# Creating Input Files

The Shared Parking Analysis Tool is designed to work with the Excel input files described in Section 2.2 Input Files. This section describes creating the input files from the Williston Employment and Parking shapefiles that come with the tool, but the user can also create the input files a different way or from different source data as long as the final input files have the columns described in Section 2.2. The Employment shapefile used in this example contains the parking generators to be studied, but the user could also use non-employment-related sources of parking generation, such as housing.

## Create Parking Lot Preference List

Open ArcMap

Load the Employment point shapefile.

Load the Parking Lot polygon shapefile.

Select the employment points and parking lot polygons to include in the analysis. Figure 23 shows only the point locations Home Depot and Walmart and their associated parking lots highlighted. Only these points and polygons will be included in the analysis.

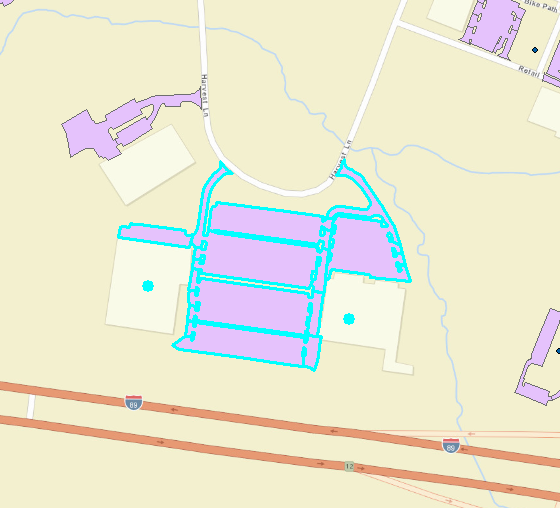


Figure : Employment and Parking lot data displayed with only the Home Depot and Walmart locations and associated parking lots highlighted

Double-click Generate Parking Preference tool in Parking Analysis Tools toolbox to open the tool (Figure 24).

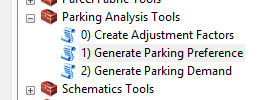


Figure : Open the Generate Parking Preference Tool

Choose Employment and Parking layers from pulldown menus in the tool for the Generator Point File and Parking Polygon File, respectively

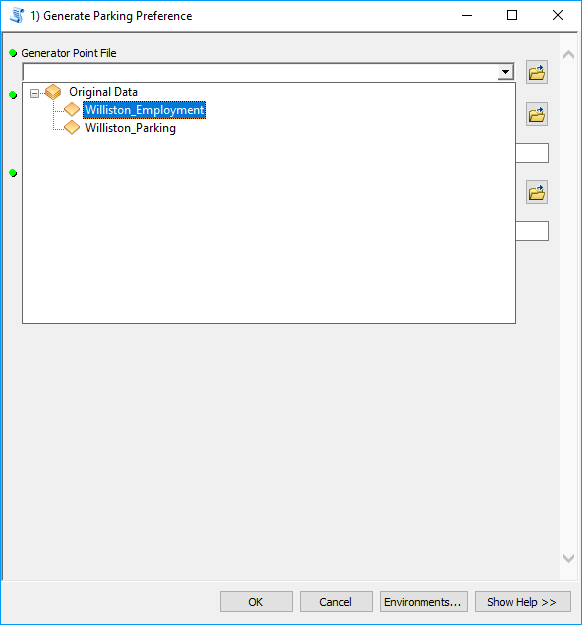


Figure : Use the pulldown menu when SELECTING data

Fill in the other three fields:

* Max Walking Distance – the maximum distance people will walk from a parking spot to a destination
* Output Data Folder – the folder the output data will be saved to
* Output Preference File – the name of the output data, do not include an extension

Click “OK” to run the tool.

After the tool runs, group the old data as Original Data, to simplify the Table of Contents. This will have all the parking lots and employers.

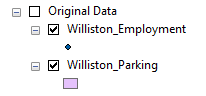


Figure : Original data grouped as "Original Data"

Load the Generator and Parking data created by the tool. This data is found in the “Working\_Data.gdb” geodatabase (/2\_Data/ Working\_Data.gdb, Figure 27).

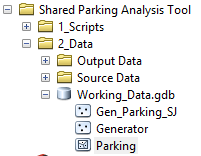


Figure : Generator and parking data created by the Generate Parking Preference tool

Group the new data as a logical description of this location, e.g. Maple Tree Place. This data will only contain the locations selected when the Generate Parking Preference tool was run.

This data also contains additional fields with unique identification (UID) numbers for each employer and parking lot

* Generator contains “Gen\_UID”
* Parking contains “Lot\_UID”

These UIDs are used in parking lot preference files created in the output folder:

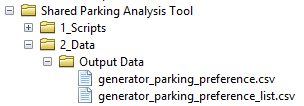
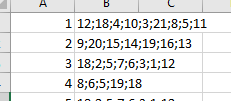
 

Figure : Parking lot preferences by generator

People visiting Gen\_UID 1 would first park in Lot\_UID 12, then Lot\_UID 18, etc.

## Create Generator and Parking Lots Input Files

### Generator File

This step will create a new Generators file similar to the Excel generator files provided with the tool. Open *Generators.xlsx* from /2\_Data/Example Data/ and Save As to the Source Data folder (/2\_Data/Source Data/).

In the Attribute Table of the new *Generator* file (ArcMap), export it as a dBase Table called *Generators.dbf*, but do not add it to the map if prompted.

In Excel, open the newly created *Generators.dbf* file so that both this dbf and the *Generators.xlsx* file are open.

In the **dbf** tab of *Generators.xlsx*, select all data from row 2 downward and press the “Delete” key. Do not right-click and select delete because this may break formulas in the workbook.

Copy and paste all data, including the headings, into cell A2 of the **dbf** tab of *Generators.xlsx*.

Copy all cells under the OBJECTID column (**dbf** tab) into the Location column of the **Generators**tab.

Open the \*\_list.csv file from root/2\_Data/Output Data/ (by default this file is named generator\_parking\_preference\_list.csv). Copy the first two columns from this file into the **ParkLots** tab of *Employment.xlsx*. The Parking Lots column on the **Generators** tab should populate automatically.

On the **Generators** tab, the user will need to modify data which does not populate automatically. Changes or check the following:

* If there are more rows of OBJECTID numbers than rows of formulas, copy and paste the formulas down until they match the number of OBJECTID numbers.
* In the LUC column, assign each row an LUC value from the **LandUse** tab. These are the land use code (LUC) numbers for this tool. ULI Second Edition does not have land use codes, so these numbers are arbitrary.
* Once the LUC values are entered, the Units column will populate automatically.
* The Size column automatically chooses the square footage of the generator from the **dbf** tab. If the units for a row are not GLA (gross leasable area) or GFA (gross floor area), then the user will need to enter the units manually, e.g. number of seats in a Cineplex.
* Check the square footage of the other generators. The data may not be accurate.
* The ParkingLots column lists parking lots in order of proximity with the closest lots first. However, this may not be the best order of preference. The user should check the parking lots list for each row to ensure that the lots are listed in the correct order of preference, that all lots relevant to a particular generator are included in its row, and that lots which should not be associated with a generator are not included.
  + In Office 365, the Excel function TEXTJOIN() can be used to concatenate a range of strings with a delimiter in between. If the user finds it easier to put each parking lot number in a separate cell, use TEXTJOIN() or CONCATENATE() to combine them.

### Parking Lots File

This step will create a new ParkingLots file similar to the Excel ParkingLots files provided with the tool. Open *ParkingLots.xlsx* from /2\_Data/Example Data/ and Save As to the Source Data folder (/2\_Data/Source Data/).

In the Attribute Table (ArcMap) of the new *Parking* file, export it as a dBase Table, but do not add it to the map if prompted.

Open the newly created *Parking.dbf* file and the *ParkingLots.xlsx* file in Excel.

In the **dbf** tab of *ParkingLots.xlsx*, select all data from row 2 downward and press the “Delete” key. Do not right-click and select delete because this may break formulas in the workbook.

Copy and paste all data, including the headings, into cell A2 the **dbf** tab of *ParkingLots.xlsx*. The top left cell should be A2 so that the numbers above the column headings are still present.

Copy all cells under the Lot\_UID column (**dbf** tab) into the Lot\_UID column of the **Lots** tab.

The “Name” and “Space” tab should populate automatically.

If there are more rows of OBJECTID numbers than rows of formulas, copy and paste the formulas down until they match the number of OBJECTID numbers.

## Create Adjustment Factors File

The adjustment factors file contains all possible combinations of adjustment factors for all land uses. It is a \*.p[[2]](#footnote-3) file created by the generate\_parking\_factors.py script. This script reads an excel file containing the possible factor values across dimension with each dimension stored on a separate tab[[3]](#footnote-4). The script creates a pandas[[4]](#footnote-5) DataFrame[[5]](#footnote-6) to store and access these adjustment factors.

The user can run this script from the ArcMap tool “0) Create Adjustment Factors” in the Parking Analysis Tools toolbox. The user chooses the excel file which contains these factors – in this case *Parking Demand and Adjustments.xlsx* in the Source Data folder. To create a new factors.p file, first adjust the factors in *Parking Demand and Adjustments.xlsx*. The user may make changes such as adding a land use type or changing a parking generation ratio. A user should be familiar with the shared parking methodology and input data before making these changes.

1. Smith, Mary S. *Shared Parking*, Second Edition. Washington, D.C.: ULI-the Urban Land Institute and the International Council of Shopping Centers, 2005. [↑](#footnote-ref-2)
2. This is a pickle file which can be read by the python programming language. See the file generate\_parking\_factors.py and <https://docs.python.org/3/library/pickle.html> for more information. [↑](#footnote-ref-3)
3. See the excel file “Parking Demand and Adjustments.xlsx” in the Source Data folder [↑](#footnote-ref-4)
4. <https://pandas.pydata.org/> [↑](#footnote-ref-5)
5. <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html> [↑](#footnote-ref-6)