Peace of Mind for Life on the Road

A Python-Based Trip-Planning Utility
for Vehicle-Mounted, Mobile, PV Systems

By Zayne Khouja

December 14, 2021

18-883, Photovoltaic System Engineering
Carnegie Mellon University

Table of Contents

Executive Summary

For years, the daily commute has been an integral part of the American lifestyle. School and work both occurred in specific offices or buildings a short drive, bike ride, or train ride away from our homes. Yet, with Coronavirus overwhelming society, our obligations have shifted to a remote format, wherein we can now work or study from anywhere in the world as long as internet access is available.

With this shift, many chose to move their lives onto the road, seaking a more affordable, adaptable, and mobile lifestyle. “Vanlife” rose in popularity, with social media influencers and films such as “Nomadland” highlighting the trend. Yet making a living from a vehicle is not as simple as tossing your laptop into the passenger seat and hitting the open road. Many have opted to use diesel generators to supply the electricity demands of remote work, yet others chose a more green approach of roof mounted solar panels.

This project aims to provide a utility for those who opt for a self-sustaining Photovoltaic System in their vehicle. Understanding the expected power output of your PV system no matter where you are in the country is an integral part of working remotely from your van, RV, or truck. Furthermore, this project helps identify how the feasibility of van-life across the US changes seasonally by providing the option to compute charge time for any month of the year.

Given a planned road-trip route plotted in Google Maps, I have designed a python script that annotates your route with the expected hours of drive time per day to ensure your PV system charges your battery to capacity before you park for the night. With this tool, you can be sure that when you settle down to connect to the internet, cook a meal, or heat your vehicle, your battery has enough charge to power the appliances you need.

The Python Script and a thorough explanation of how to use it can be found on my GitHub at https://github.com/zayne98/photovoltaics-final-project. Anyone can clone the project and use it on a route they plan. Please see the “Program Overview” and “Example Program Flow” sections below for an elaboration on how to use the script and demonstration of the insight it can provide.

Background

This project may appear relevant to only a small subset of the population, yet it is worth noting that hundreds of thousands of Americans pursue a nomadic lifestyle, living and working out of their vehicle on the road[1]. Looking at instagram alone, #vanlife trends have grown by over 310% since 2017, with over 7 million tagged posts in 2020 alone[2]. Evidently, a trip-planning utility for a mobile lifestyle has a widespread audience.

Project Goal

When living out of a vehicle for an extended period of time, electricity-dependent comforts and modern commodities are just as critical as in a typical established house. Heaters and fans are important to regulate temperature regardless of the season. Signal boosters are crucial hardware components to ensure you can receive cellular signals and work on the internet when in remote areas. Induction stove tops are valuable to cook while living away from civilization, and pose an efficient and clean alternative to gas burning stoves. Evidently, there are a multitude of applications which would draw from a reliable PV system.

Importantly, however, each of these appliances will only be in use for a short period of time each day. Some (such as heaters, fans, stoves) may only be used a few days a week. Therefore, the exact load on the PV system is quite variable. Instead, such a PV system simply strives to have substantial charge in a reasonable battery for the occasions when you depend on it most.

This trip-planning utility aims to provide peace of mind when living on the road, so you can rest assured that your battery has enough charge each day. It is designed to couple seamlessly with Google Maps and serve as a secondary tool for analysing your route before you embark. Its primary purpose is to ensure you do not find yourself in a part of the country where you cannot power the appliances needed to exist safely and comfortably out of your vehicle. Additionally, by providing the option to annotate your route with charge time for any month of the year, this project can also be used as a utility for visualizing how the feasibility of a mobile PV system fluctuates with the seasons.

It is worth noting that these mobile PV systems are not designed to charge the battery of the vehicle itself. For example, charging your Tesla or other other electric vehicle off small-scale solar panels on your roof is simply not feasible. Instead, such a system would typically charge a

[1] *How veterans of #vanlife feel about ... - The New York Times*. New York Times. (n.d.). Retrieved December 5, 2021, from https://www.nytimes.com/2021/04/02/us/living-in-a-van-coronavirus-pandemic.html.

[2] Mike, & About The Author Mike Mike leads research. (2021, January 11). *Vanlife statistics & trends in 2021*. Where You Make It. Retrieved December 5, 2021, from https://whereyoumakeit.com/converted-vehicles/stats/.

secondary, deep-cycle battery which powers your appliances directly or through an inverter when AC is required.

Motivation

This may seem like a strange project for a 23 year-old graduate student to propose and develop, yet it is directly pertinent to one of my primary hobbies. Starting in December of 2020, I purchased a cheap, used, 2006 Toyota Tundra 4WD truck with a topper over the 6-foot bed. Thanks to Covid-induced remote school and my amplified desire to procrastinate as a second-semester senior in college, I regularly planned and executed multi-day road trips across the country, tuning into zoom classes from random coffee shops.

I built out my truck only minimally, setting up a comfortable mattress, constructing additional storage features in the truck bed, and adding curtains to the windows. However, I did not build any electrical systems or do any fancy wiring. I cooked off a gas stove and used a small 10,000 mAh battery pack to power a small lightbulb I hung over the mattress. The photos below show my setup and truck.

Once I graduated, I spent 3 months over the summer living out of my truck and driving all over the country. During the hundreds of hours spent driving, I often thought about the feasibility of adding solar panels to the 10 foot roof of my truck so I could live more comfortably and genuinely work out of my vehicle. Yet, I disregarded this idea thinking a PV system would be too unreliable to really provide the power I needed.

However, with this course, I was provided an opportunity to explore this idea in greater detail. The script I designed is a direct product of the concerns I had with implementing a mobile PV system for life on the road. As such, I have a particular interest in this project and believe it has a genuine use for many who find themselves in a similar situation as mine.

Program Overview

This project is a Python program that anyone can download and use with a trip that they are planning. As such, it is important to discuss how the program works, what it offers, and what its limitations are. Please refer to https://github.com/zayne98/photovoltaics-final-project for the project code.

This project was written in python and uses Google's Geocoding (and reverse geocoding) APIs to parse a google maps route and convert locations to their coordinates. It then maps these coordinates to the states they fall within. Then, using state-level data from renewables.ninja, it maps these states to irradiance and temperature values per month, and uses these values to compute charge time. A detailed explanation of the data I used can be found in the "Data and Limitations" sub-section below.

How to Use the Program

In this section, I will outline the steps that anyone can take to use this program for their own purposes.

1. Plan your trip on Google Maps.

   Google Maps is the most ubiquitous mapping and routing software available, and it is the only service my program currently supports. Simply open maps.google.com or use the Google Maps app from your phone. Start with your starting point routed to your final destination, then add waypoint stops along the route to shape your drive as you plan.

   Once you are done, copy the full URL for your Google Maps route and save it for step 3

   Note: For long, cross-country routes, it is highly recommended that you add many waypoints, as this helps improve the accuracy of the program.

2. Download the Python script.

   From the GitHub link above, either download a zip of the project directly (and extract the zip), or clone it onto your local machine from your command line or terminal shell.

   As this project uses Python, there are several libraries you must install for the code to run. These libraries are detailed in the "Troubleshooting" section of the README.md file included in the project repository. This file also discusses the project's codebase in additional detail.

Once you have the code and relevant libraries installed, you should be ready to use the program. If your Google Maps URL has any "!" and you are running the program from a bash shell (like the Terminal on a Mac), you may need to run "set +H" first to fix how bash handles exclamation points.

3. Run the program with your Google Maps route to gain insight on charge time.

   From your command line or terminal shell, navigate to the directory where you downloaded the project, then navigate into the *code* directory.

   The script is called *project.py*, and it takes one required argument and a second optional argument. The first argument is the URL of your Google Maps route, and the second is the month for which you would like to visualize charge time. This second argument can be given as a full month name (ie. January or March) or as a month's 3 letter abbreviation (ie. Jan or Mar). If no month argument is provided, the script defaults to using July's data.

   Run your program from the command line as follows, replacing <URL> with your Google Maps URL from step 1, and replacing <MONTH> with the month you would like data for: **python project.py <URL> <MONTH>**

   The program should take roughly 30 - 60 seconds to finish. It outputs 3 plots along the way, and stalls execution each time a plot is displayed. Therefore, you will have to exit the first two plots for the program to complete.

4. Analyze the program's output.

   Once executed, this program prints various logging outputs and generates 3 plots. First, it parses out the separate waypoint locations along the route and plots them, then it interpolates a straight line between these points and plots that route. Finally, it maps each coordinate to an irradiance and temperature value for the given location and uses this to resolve the current generated by the solar panel. With this info and some stats on your battery, it computes the expected daily drive time to fully charge the battery along your route and plots this data with separate legs of the trip labeled with their drive time in hours per day.

You can then use this final generated plot to determine the pace of a road trip and understand roughly how long you must drive to make sure the battery can power all your appliances once you stop driving for the night.

By rerunning the program with a different month, you can compare expected charge times for different times of the year.

Features

The program is designed in a highly modular fashion, meaning it can be applied to any route and to a wide range of PV systems or setups. Given the capacity of your battery system and the datasheet for the panel you are using, you can easily adjust the program to your specific needs. In the *project.py* file of the project, you will find numerous constants defined at the top of the file. Below the comment "*Constants for a deep cycle battery"* and the comment "*Constants for Max Power Point calculations"*, you can set the constant values according to the specifications of your system.

Furthermore, the program can analyze any Google Maps route you plot, as long as the route stays within the USA. This geographic limitation is discussed further in the "Data and Limitations." This means you can plan local routes within your state, cross-country drives, or massive road-trip loops all over the nation.

Additionally, thanks to the data I have available, I can analyze a route for any month of the year. This provides an additional utility for considering the temporal aspects of your prospective road trip to understand how the seasons may impact your drive time.

Data and Limitations

The inaccuracies and limitations of this project are the direct result of the data that is available to me for free. Sources of error come initially from how I translate a Google Maps route into a series of coordinate waypoints and, secondly, from how I retrieve irradiance and temperature data for each of these coordinates. If I had full, unrestricted access to the services I will describe below, then I believe this project would be, in theory, quite accurate. As it stands, it serves as a valid proof of concept that reveals trends and valuable insight into route planning with a mobile PV system.

Firstly, the full Google Maps API is restricted behind a paywall managed by Google Cloud. When my program retrieves and parses the Google Maps URL the user provides, it tokenizes each waypoint you plotted in the route. Then, it uses Google Cloud's Geocoding API to translate each location to the geographical coordinate that captures it. In building a more granular route out of these waypoints, my program would ideally ping various Google Maps APIs to deduce specific routing info for the drive between each waypoint. In this way, my program would capture every turn and the exact road driven on your road trip to increase the accuracy of the output charge times. However, these APIs all require payment, so, instead, my script simply interpolates a straight line between each pair of waypoints, then drops granular coordinates roughly every 30 miles along the route (this granular spacing is also customizable in the script). For a long distance road trip route, straight line approximations are a fairly accurate representation of how state highways and interstates weave throughout the country. As such, this is not a major source of inaccuracy, yet full API access would certainly improve the script's potential.

Secondly, in order to compute drive time, my program attempts to calculate the expected current output of the PV system given the geographic locations along the route. This involves translating coordinates, panel tilt, and date and time values to a predicted or expected irradiance datapoint. The website service Solar Radiation Data (SoDa) provides an API for programmatically retrieving historical irradiation data for customizable parameters and coordinates across the world[3]. They expose this API for "curl" or "Wget" access. Unfortunately, this API requires a paid subscription. I tried sending an email to their customer service requesting temporary student access to the API service, but, regrettably, I have yet to receive a response. If at some point in the future I receive access to this API my program accuracy would increase drastically.

As a result of the payment restricted SoDa API, I used an alternative, free service to retrieve irradiance and temperature data. Renewables Ninja's website provides downloadable CSV files containing daily and hourly irradiance and temperature data on a per-state granularity for past years[4]. For each of the 50 states and the District of Columbia, I downloaded separate 22 MB datasets. Then, I used a script to condense the 1.1 GBs of data into a single 19 MB file containing irradiance and temperature values tagged by their date and state (both the script and condensed CSV are available on my GitHub in the *code* directory named *dataset_cleaner.py* and *irradiance.csv* respectively). Therefore, to translate coordinates to irradiance data, I map the coordinates to the state that captures them, then retrieve the maximum irradiance values for the requested month in that state. This effectively breaks the road trip route up into distinct per-state legs, allowing me to annotate each state section with a specific charge time. Additionally, this means my program only supports road trips routes that entirely exist within the USA, as I do not have data outside of the country. As this is the only data available to me for free, it is the best

[3] http://www.soda-pro.com/help/automatic-access/helioclim-3-archives
[4] https://www.renewables.ninja/

option for my program as it stands. Clearly, however, a more granular approach to translating coordinates to data values would improve the script's accuracy.

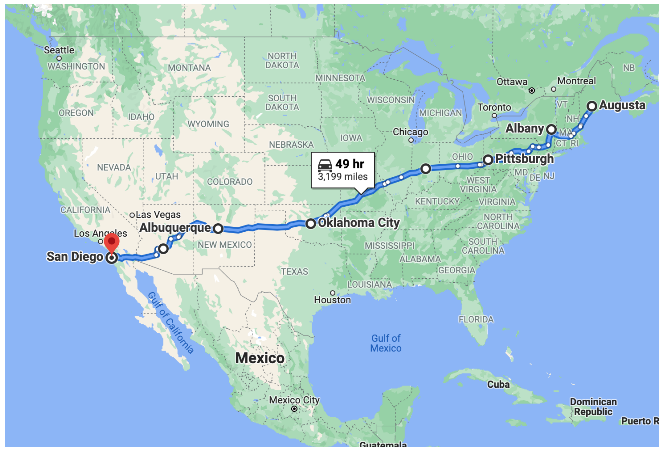
Due to these API limitations, it is evident that my script currently generalizes a route both in terms of routing and irradiance data. I believe both these shortcomings lead to an underestimate of charge time, as the route is abstracted to be noticeably shorter than it is in reality. In an effort to account for this overestimation, I have also idealized the battery to increase charge time and theoretically cancel out these limitations. Specifically, I ignore the potential depth of discharge for the battery your system uses. Most lithium ion batteries discharge only 80% of their capacity, and lead acid batteries discharge only 50%. Instead of computing charge times as the time required to refill only these fractions of the battery, I consider charging the full battery to hopefully skew my data towards a worse case situation and provide more conservative charge times.

If I had access to the full, paid, versions of these two APIs, I would adjust my modelling of the battery system. With these changes, the output of this program would, in theory, be fully accurate.

Example Program Flow

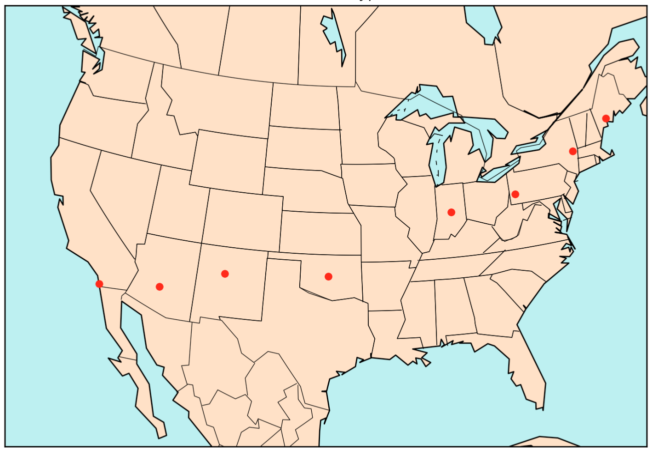
As this project is an open-source script available for anyone to use, it is worth walking through a complete example of how to use this program to analyze your road trip route. The "How to Use the Program" section above discusses how to provide input for the script. This section will instead explain how to interpret the output of the program.

Consider a cross country road trip route that spans the entire country from Augusta, Maine, to San Diego, California (pictured to the right, as plotted on Google Maps). A chose to use this route as an example as it spans an extensive range of latitudes and geographic features from the east coast Appalachian Mountains to the deserts of the Southwest. In this way, it helps demonstrate the seasonal and geospatial trends that relate to the efficiency of a PV system. Charts for this route are also included on the GitHub repo in the *example* directory.



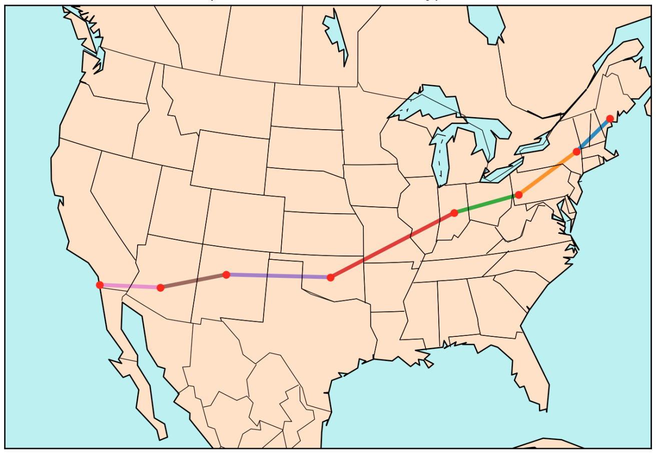
For this route, I added waypoints in Albany NY, Pittsburgh PA, Indianapolis IN, Oklahoma City OK, Albuquerque NM, and Phoenix AZ. These waypoints help improve the accuracy of my script and also emulate how a genuine route would be annotated with waypoints. The map on the right shows the first output when the above route is run with my program. It simply shows how I translate locations to coordinates and plot them on a map of the US.

Initial Route Waypoints

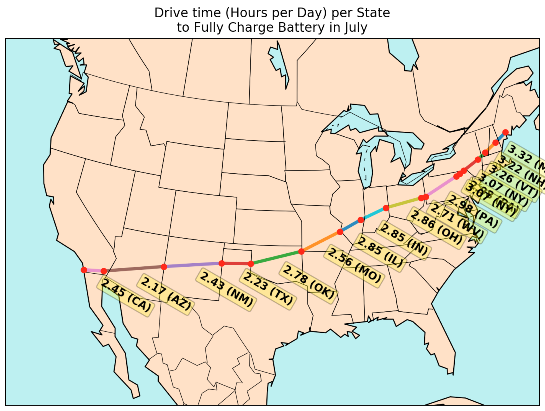


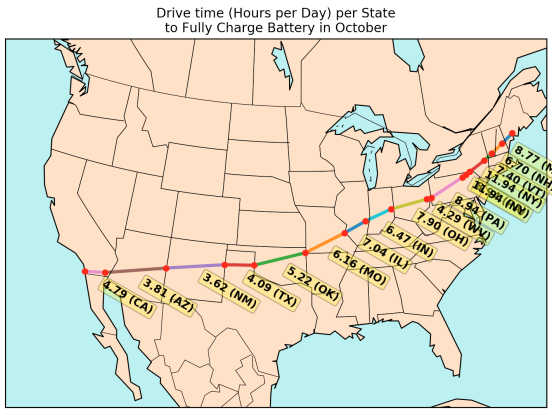
The next step in the program flow is interpolating between these waypoints with straight lines to achieve a more granular visualization of the route with additional waypoints placed along these lines. The linear simplification of the route is discussed in the "Data and Limitations" subsection above, and serves as a rough approximation of a route. This map is shown at right.

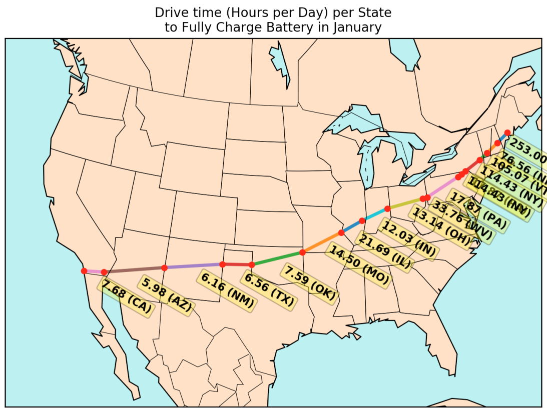
Interpolation Between Route Waypoints

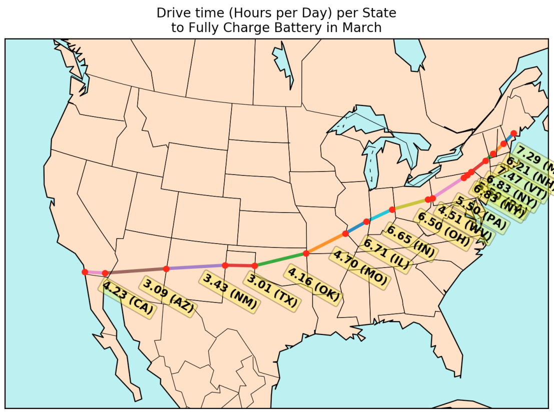


Finally, the script outputs a final chart annotating the interpolated route with per-state charge time values. The title of this final chart reflects the month you provided as input, and the annotated charge times reflect the irradiance and temperatures for those respective months. Below, find four of these annotated plots all for different months of the year spread out across the seasons (July for Summer, October for Autumn, January for Winter, and March for Spring). Below these charts is a table containing the condensed charge time values for each of the states on the route. Units in the table represent charge time in hours per day.









	CA	AZ	NM	TX	OK	MO	IL	IN	OH	WV	PA	NY	VT	NH	ME
July	2.5	2.2	2.4	2.2	2.8	2.6	2.9	2.9	2.9	2.7	3.0	3.1	3.2	3.2	3.3
October	4.8	3.8	3.6	4.1	5.2	6.2	7.0	6.5	7.9	4.3	8.9	11.9	7.4	6.7	8.8
January	7.7	6.0	6.2	6.6	7.6	14.5	21.7	12.0	13.1	33.8	17.9	114.4	105.1	76.7	253.0
March	4.2	3.1	3.4	3.0	4.2	4.7	6.7	6.7	6.5	4.5	5.5	6.8	7.5	6.2	7.3

Now, it is worth discussing the implications of this output. As we can see, the general trend for all months shows that as you drive further southwest, charge time decreases considerably. In the hot summer month of July, all states display charge times between 2-4 hours. This reveals how one can expect considerable direct sunlight during this time of the year, allowing a complete road trip wherein you can fully charge your battery every day. Similarly, in March, charge times span between 3 and 8 hours, which are also realistic time frames to allow full charge every day. However, in October we see NY's predicted charge time is almost 12 hours, which is not realistically feasible given the hours of sunlight during this month. As such, this chart helps someone living out of their van realize that a route through New York may not be feasible if they require a reliably full battery every day. This observation is further amplified in the January chart wherein we can see that east of Oklahoma, a full charge is simply unachievable. January in Maine suggests that 253 hours of direct sunlight is needed to charge your battery. Clearly, self-sustaining life off the grid is not a possibility in Maine during the Winter. As such, the output of my program can help ensure safety and reliability for the season you are travelling.

Additionally, it is worth commenting on some anomalies displayed in these charts. Specifically, while the route through southern California is seemingly at a similar geographic and latitudinal level as the route through Arizona, the predicted charge times for California are always higher. This reveals a shortcoming of my per-state irradiance data, where California's predicted irradiance captures the entire state from north to south and desert to mountains. On the other hand, Arizona, as a state, is fairly uniform, meaning that the route through that state captures southern desert data values. I believe West Virginia is a similar example of this anomaly that results from my data limitations. Surprisingly, in October and March, the predicted charge time values are far lower than those of neighboring Appalachian states. This trend does not make sense until we consider the high altitude, alpine regions of West Virginia that have sparse, if any, tree cover. By averaging irradiance across the state, a route that passes through the deep, forested valleys of WV will draw data averaged with the exposed plateaus higher in the state. As mentioned in the "Data and Limitations" sub-section above, these anomalies should theoretically be avoided if the payment-protected accurate APIs were available.

Technical Assessment

As this project serves more as a utility for an existing PV system, the technical assessment is minimal. Nonetheless, this section discusses potential PV arrays, potential battery specifications, and potential appliances which could run off a mobile PV system. It is important to note that roof space defines the number of panels you can mount, and this directly impacts how big your battery can realistically be. As such, the user of a mobile, automobile PV system must restrict their appliance usage to an absolute minimum and only for essential applications due to the PV scaling limitations.

PV Array Sizing

Unlike other static PV systems, the size of your array is dependent on your automobile's roof size. RVs often have more than 30 feet of roof space, which could accommodate several full sized panels. Vans may have 20 foot roofs which restricts space further. Individuals living out of cars and trucks may only have space for a single panel on their roof. My program currently operates under this single-panel case to provide a worst case output for the minimal PV Array size. Depending on the panel your system uses, you can adjust the constants used in the program to reflect your system design.

Battery Capacity Sizing

The battery design is also flexible given your price point and solar array. Lithium Ion and Lead Acid batteries are both valid options for a setup like this. It should be noted that a deep cycle battery is ideal, as it will better accommodate the possibility of completely discharging your battery regularly. For the current program and the example presented above, I assumed a 65 Ah battery, specifically referencing the data sheet for a Duralast Deep Cycle Marine Battery[5]. This size means the battery can realistically be charged fully in a day during summers. However, if a user may only be drawing power every few days, they could consider a higher capacity battery system that takes several days to fully charge.

Load Estimation

[5] https://www.autozone.com/batteries-starting-and-charging/rv-battery/p/duralast-29dp-dl-group-29-deep-cycle-marine-and-rv-battery/95824_0_0

Realistically, most appliances draw more power than a small-scale PV system can provide. Therefore, usage must scale as a result of the array design. Some common appliances for a vanlife setup include a signal booster to ensure internet connection when in remote locations, a heater or fan to regulate temperature throughout the year, an electric stove for cooking, and lightbulbs. However, depending on your vehicle size, your lifestyle, and your price point, many other appliances can be considered. Additionally, if any appliances require AC power, an inverter is an additional component that is required.

Conclusion

Vanlife is the hot new lifestyle of the decade, yet actually living out of your vehicle requires a consistent source of power to run appliances such as stoves, coolers, heaters, fans, or lightbulbs. Some opt for diesel generators, yet the greener solution of a roof-mounted solar panel wired to a deep-cycle battery can supply passively generated electricity for these applications. Yet, depending on the season and where you are on the road, the output of your PV system may be unreliable or insufficient. This project provides a valuable utility for ensuring peace of mind as you travel across the country in your RV, van, or other vehicle equipped with a PV system.

By providing a tool for computing how many hours your solar panels need direct sunlight depending on your road trip location, you can drive assured that your battery will be fully charged when you park for the night. This project therefore resolves a major issue that PV systems have which a simple diesel generator lacks: reliability. As such, the program has applications for people with existing PV systems, but it can also serve as motivation for others to transition to a solar-powered mobile lifestyle who may have been nervous about the reliability-related drawbacks.

Regrettably, the freely available APIs impose several limitations on the accuracy of the charge times the program outputs. With full access to SoDa and Google Maps' API, the accuracy of this program would improve. Nonetheless, in its coarse, current form, it serves as a proof of concept for the validity of a vanlife PV system and still yields reasonable results that can be applied to any USA road trip route.

From a personal perspective, I have previously considered mounting a solar panel onto the roof of my truck, yet the reliability issue made me apprehensive to invest in the technology. However, after developing and experimenting with this program, I feel confident in the feasibility of powering my appliances off a solar panel, and am planning to save up for and install a PV system on my truck.

Overall, access to the paid APIs represent a key future improvement to this project, but as it stands, the Python script published on GitHub provides a relevant and useful utility for anyone planning to live out of their vehicle. With a few tweaks and additions, this script could be ported over to a mobile app, so anyone could translate their routes into charge time estimates straight from their phone while on the road.