SCC Client Report
An Analysis Plan for Evaluating Location of a New Vance County EMS Station

November 16, 2023

1 Background

The Vance County EMS system currently operates four ambulances and two stations, one located in the Southern district and one in the Central district. This setup has led to residents in the Northern district being under–served; average response times are noticeably higher there than in the other two districts, with delays being significant enough to change outcomes in potentially fatal scenarios. Coupling this with a rising demand for EMS services from the North, Vance County is considering a northern station in one of two proposed locations; your plan to use past call data to inform your decision. These originate from written and digitally recorded records. The dates of each trip have been manipulated into a mock dataset to protect patient privacy. Each row represents a single trip with details such as the dispatch station, the coordinates of the patient's address, various logged times (dispatch, en route, arrival to site, leaving site, arriving at hospital, and clear time), and other details about the trip. You want to use this information to analyze travel times, the associated system load, and assess different station location and vehicle allocation scenarios. You have asked us to help answer the two following research questions: 1) which of the proposed North district station locations would better serve the community?, and 2) how should the four available ambulances be allocated to best serve the community?

2 Recommendations

Scenarios	S0(Current)	S1	S2	S3	S4
Far North	0	0	1	0	1
Near North	0	1	0	1	0
South	3	3	3	2	2
Central	1	0	0	1	1

Table 1: Summary of the five possible station location and vehicle allocation scenarios that are under consideration.

Our recommendations are structured around the multi–step analysis pipeline that we suggest for your analysis. In overview, we propose that you use caller coordinates and Google's Distance Matrix API to generate additional travel time estimation features from the possible stations, then construct a model using these data to estimate both ambulance travel time and load on the emergency system for each scenario of interest. The proposed analysis pipeline is depicted in Figure 1 and begins with data quality checks, including the identification of out–of–county and other atypical trips, followed by the utilization of the Google Distance Matrix API to estimate travel times from all potential station locations. At this point, there is the possibility of constructing additional models to improve the accuracy of the Google API estimates. The next step is to use estimated travel times to compute travel time and system load summaries of the various station scenarios, then to verify if the timing data is consistent and correct. We propose several measures of system load and highlight an example analysis using one of them Finally, the travel time analysis focuses on modeling the dependency of estimated travel times on scenarios.

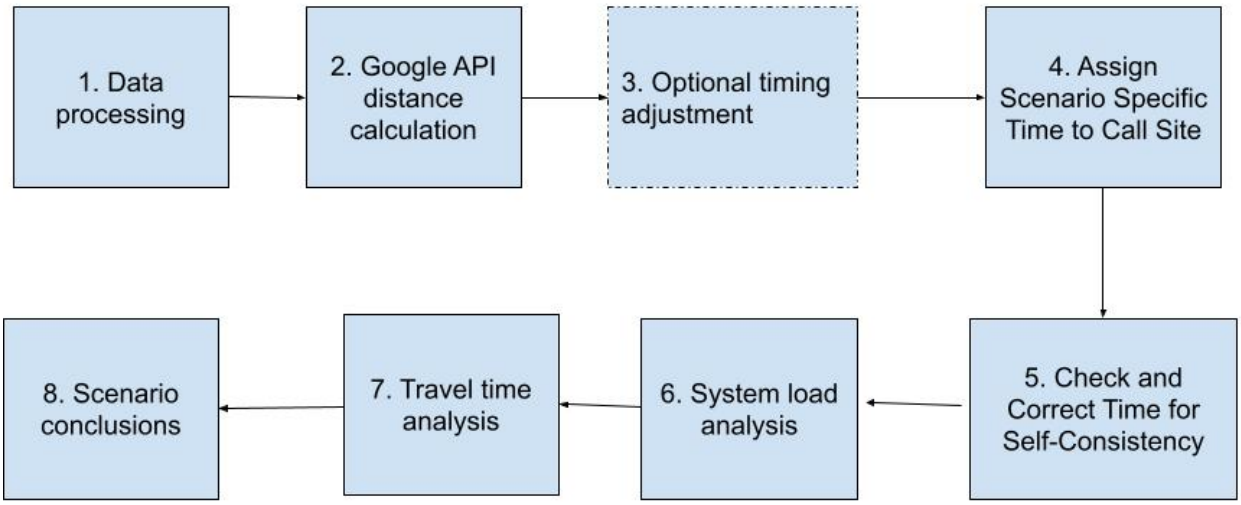

Figure 1: Graphical depiction of the proposed analysis pipeline.

2.1 Preliminary Data Processing and Computations

Prior to proceeding with the analysis, we recommend that you familiarize yourself with the data, check for unexpected features and correct any errors you find. For example:

- Verify that all trips within the data fall within Vance county boundaries.
- Decide whether to include calls with atypical features, for example calls where latitude and/or longitude is not within Vance County boundaries.
- Determine how frequently expected dispatch rules are followed, for example that all calls are addressed by an ambulance from the closest station if one is available.

- Familiarize yourself with the temporal and spatial patterns in the data, for example:

  - Average distance and travel time from a call to each station.
  - The call volume for each region (North, Central, and South).
  - For each region, call volume and average travel time by day of the week, hour of the day and if attainable, month of the year.

Effort spent upfront identifying and correcting data errors and developing a comprehensive understanding of the data will improve the quality of subsequent analyses.

The Google Distance Matrix API is a service of the Google Maps platform that accepts multiple origin and destination locations and returns a matrix of expected travel times and distances. The API is flexible, and offers a variety of features that a user can incorporate into their estimated travel time for a given route. One key feature that we make use of in our recommendation and we will highlight is the `traffic_model` parameter: a user can input `optimistic`, `pessimistic`, or `best_guess` which corresponds to the API's method of estimating travel time for the given `traffic_model` scenarios. Read more about the Distance Matrix API here.

The steps to get started can be found by following the table of contents at the link above. Briefly, this will include setting up your Good Cloud project, creating an API key and then calling the Google Distance Matrix API with the `R mapsi` package, which we will describe further below.

The Distance Matrix API allows for usage of built–in parameters to better resemble real-life scenarios. Various inputs include arrays of origin and destination locations, method of travel (walking, transit, biking, driving), as well as specific transit and driving options. We will focus on the `driving_options` parameter. These driving options contain specifics regarding traffic models: best guess, pessimistic, and optimistic. These different traffic models slightly manipulate the times of transportation by adjusting the `duration_in_traffic` metric using historical traffic conditions as well as live traffic information. This is valuable in the context of this project because it allows for better alignment of estimated travel times with both emergency (lights on) and non-emergency scenarios. While the Google distance matrix API is helpful for estimating travel times, it only can estimate travel times in the future. For this project, this implies that the dates of the trips in the dataset have to be changed to a date in the future. This could lead to trips being estimated on a different day of the week than they might've taken place and possible distortion from accurate traffic scenarios.

Be mindful of the limited bandwidth available when making calls/requests to the API. Conducting a comprehensive, full analysis will likely require numerous individual requests separated by brief time delays. In all the scenarios used (Best Guess, Pessimistic, Optimistic, Green Light), you will observe the command `Sys.sleep(0.5)`. This command introduces a brief pause at the end of each iteration within the for-loop, allowing the API sufficient processing time. Consequently, it may be necessary for this code to run overnight due to the extended duration required for data collection.

2.2 Estimating Expected Scenario–Specific Trip Assignments and Travel Times

2.2.1 Model–Based Adjustment of the Google Travel Times

The construction of a multiple regression model can provide insight into whether there is value in adjusting the Google API estimates prior to analysis. By setting the observed travel times from the station to the site of the call as the response variable, you can use predictor variables, such as various API estimated travel times, time of day, and distance traveled, to check model fit and accuracy regarding precise time estimates. Overall, this optional step is helpful because factors such as traffic patterns, route changes, or even specific local travel conditions could affect the API's ability to accurate capture travel time – in which case, adjustments may improve the accuracy of the analysis.

For the purpose of your analysis, there are multiple models that you can choose from, such as linear models, as well as the Bayesian Additive Regression Trees Model, also known as the BART Model. At a high level, the BART model utilizes a Bayesian sum–of–tree framework to help capture flexible, non–linear patterns in data (Chipman et al., 2010; Tan and Roy, 2019), which can be helpful with travel time deviations and other factors that could affect your API estimates. Linear models are effective when the relationship between the predictor and response variables are approximately linear, while tree–based methods, such as the BART model, can model intricate non–linear relationships within the data and discover and account for interactions among the covariates. The use of one model type over the other depends on the nature of your data and the patterns within it.

As a starting point, we recommend the construction of a variable based on observed the "best guess" travel times retrieved from the Google API. For each trip between the destination and the Central or South station, you will fill in the travel times as recorded by the API. These observed "best guess" travel times, along with additional variables such as time of day, day of the week, season, and the emergency/non-emergency nature of the call, form the basis for the corrective modeling step. If you find this step to be useful, you can use your chosen model to correct the estimated Google travel times to all of the observed call locations from each station.

Potentially useful variables to use at this step include dispatch time, the API travel time estimates, time of day, distance traveled, day of week, season, emergency status.

Compare several models using different formulations (e.g. linear and tree–based; different covariates), against the "straw man" approach of using the Google best guess estimates without adjustment. Divide the observed trip data into training and test sets, the former for model estimation, that latter for model evaluation. Evaluate each model using Root Mean Squared Error (RMSE) on the test data. The lower the RMSE value, the better the model fit. If one or more of the adjustment models improves RMSE enough to warrant the corrective step, use the model with lowest RMSE for this purpose. If not, skip this step and use the un–adjusted best guess estimates.

The following summarizes the corrective analysis step:

1. Build multiple regression and BART models with response and predictor variables.

2. Calculate RMSE for these predictions. Also, calculate RMSE when only using Google API's "best guess" values.

3. Compare both RMSE values for all of the models and identify which models are the most accurate fits.

4. Proceed by either adjusting API estimates or keeping them as is.

2.2.2 Assign and Check Scenario–Specific Travel Times

Prior to analysis, the Google API travel times, adjusted as above or not, must be combined into scenario–specific travel time variables that map the various calls to responding stations following rule sets that mimics real–world dispatch decisions and fill in estimated travel times based on those dispatch rules. For example, the 23rd observed call in the mock data set you provided has `REF.GPS.LAT` of 36.4240 and `REF.GPS.LON` of -78.4538, which is located at the Northern end of Vance County. For this observed call, the ambulance departure location might be assigned Central station, Near North station, Far North station, Near North station, and Far North station (for Current Scenario, Scenario 1, Scenario 2, Scenario 3, and Scenario 4 respectively).

Once the process of assigning scenario–specific departure locations for each observed call is complete, you will choose to use the raw Google best guess times or the corrected times (using the model mentioned above). Note that the scenario–specific ambulance departure location assignment method described above assumes that there is a vehicle available at each of the scenario–specific stations at any given point in time. This is not always the case. These situations will need to be resolved before it is possible to accurately assess the effects of the various scenarios on average travel times.

It is important that the corrective step accounts for overlapping trips, as it affects the availability of vehicles. Since the dispatch times are in temporal order, this can be done by identifying trips whose dispatch times precede the estimated clearance times of one or previous trips. The estimated clearance times are scenario–dependent as they account for the estimated scenario–specific travel times and can be calculated by adding the difference between the dispatch and en route times to the dispatch time, the estimated travel time to the site, and the difference between the time of arrival and the time of availability. This process is illustrated in the script we provide.

2.3 Load Analysis

In this section, we explain how to conduct a load analysis to examine how the different scenarios would increase or decrease stress on Vance County's EMS System. Several methods for evaluating system load were considered, including:

- number of ambulances available at any given time of day
- amount of time each ambulance is in use
- a binary indicator for whether the ambulance at the closest station is available

The last metric above has the advantage that it is relatively easy to compute, as you will see in the script we provide. For this reason, we highlight our recommended load analysis using it as an example. The value of this binary indicator should be zero if the ambulance at the closest station is available when the call is placed, and one if the ambulance at the closest station is not available. The variable indicates that a call presented a load stress on the system.

Our exploratory analysis of the mock data indicates that many calls are not informative for cross–scenario differences in load or travel time. These appear to mostly be calls that result in dispatch from the central station as it is staffed under all scenarios. For this reason, we suggest a

two–staged approach to studying load: (1) examine spatial and temporal call features that characterize call events where the response is sensitive to scenario versus not sensitive; and (2) characterize the variation of load across scenarios only when it does vary.

The first analysis involves estimating the probability of whether a call's closest station varies across scenarios. We suggest you create a 0-1 binary indicator (dummy) variable for the event that load is sensitive to scenario (i.e., non–constant across scenarios). This will identify the calls for which ambulance placement does make a difference versus those where it does not. A logistic regression model can then be constructed to identify time– or location–related variables such as season, hour of day, event ID, etc., that are associated with this variable. Alternately, you may prefer a more descriptive analysis.

The second analysis will use only the calls that are observed to vary across scenarios, and it will estimate the dependence of the load metric on scenario (i.e., how the load metric varies between different scenarios). The data set for this analysis will include a predicted load measurement under each scenario for each call. It therefore will include five repeated data points per call, one for each scenario. This is a classic repeated measures design and it is necessary to account for this additional source of variation. This can be accomplished by fitting a mixed effects model. Since the load metric is binary, this will take the form of a logistic link binomial Generalized Linear Mixed Model (GLMM). The model should be formulated with binary load as the response variable, scenario (and possibly other covariates) as a fixed effect and call/event ID as a random effect. We illustrate this modeling approach in the script we provide.

2.4 Travel Time Analysis

Ambulance travel time, a significant component of response time, is the metric you aim to optimize by comparing the average travel time across various ambulance distribution scenarios. Our proposed approach to analyzing travel times is, for the same reasons as above, also two–staged. The initial step in modeling travel time is checking whether the estimated travel times exhibit variation across scenarios, which enables the identification of instances where travel time is influenced by the scenario under consideration, and those where it remains the same. A binary outcome, hereby referred to as "differ," is formulated, wherein "1" denotes a variation in travel time across scenarios, and "0" indicates an absence of variation. The analysis of this outcome proceeds as described above for the analogous load variable. It is modeled using a logistic regression model, specifically utilizing the `glm()` function in `R`, due to the binary nature of the outcome. The response variable is the binary outcome "differ," and the predictor variables, or covariates, encompass factors such as season, hour of the day, day of the week, priority, and system load. This model provides estimates of the probability of observing a variation in travel times across scenarios, conditional on the covariates. For instance, during peak summer seasons or specific hours of the day, the model may reveal a heightened probability of variation in travel times across various scenarios, potentially attributable to factors such as escalated traffic or road closures.

A key focus of your study is determining how typical travel times would change under the various alternative scenarios. The next step is to model the association between travel times and scenario using the calls with travel times that are dependent on scenario (i.e. when `differ=1`). The design and analysis for this stage parallels that described above for load variation. Since travel times are continuously measured, we propose here that you use a Linear Mixed Model (LMM) for the analysis. These models can be fit using the `lme4::lmer()` function in `R` (Bates et al., 2015). In this model, travel time is the response variable, scenario (and possibly other covariates) is a fixed

effect and call/event ID is a random effect. We illustrate this modeling approach in the script we provide.

References

Bates, D., M. Mächler, B. Bolker, and S. Walker (2015). Fitting linear mixed-effects models using lme4. *Journal of Statistical Software 67*(1), 1—-48.

Chipman, H. A., E. I. George, and R. E. McCulloch (2010). BART: Bayesian additive regression trees. *The Annals of Applied Statistics 4*(1), 266 – 298.

Tan, Y. V. and J. Roy (2019). Bayesian Additive Regression Trees and the general BART model. *Statistics in Medicine 38*(25), 5048–5069.