Optimization of solid waste in Blantyre, Malawi

Nicolas Seemann-Ricard

# Abstract

# Introduction

The project seeks to minimize the costs of operation of the municipal solid waste management service in Blantyre Malawi, while limiting overflowing of the communal skips.

## Justification and Research Questions

1. How can we model with limited data
2. How many trucks are needed to service all skips? What would be the mileage and cost of servicing all skips without overflow?
3. What routing schedule?

# Data analysis

In order to formulate feasible and pertinent recommendations, parameters reflecting the situation need to be calculated. Specifically, the rate at which bins are filling, so as to know the frequency at which they need to be emptied.

## Provided data

### Set of skip locations

A set of skip, assumed to be all the community skips to be serviced in Blantyre, along with GPS coordinates is given. Each skip is 7m3. Some data points have the same name, when several skips are in the same area (e.g. three skips in the same market). It is unknown if more than one skip can be represented by one coordinate point. Furthermore, there is no information about the intended nature of the waste (organic or inorganic). The locations of the municipal dump (Mzedi dump), the truck storage facility and the compost facility are also used.

|  |
| --- |
| Figure 1: Locations of skips, dump, compost facility and truck storage |

### Skips filling data

Filling data for a number of skips is provided. The time series are named in the following fashion: “area+type of waste+number”. The names of the areas do not always match the ones from the set of skip locations, in addition to the ambiguity of several skips present in a certain area. The type of waste also complicates the analysis.

Over a certain period of time (depending on the skip), a measurement on a scale from 1-5 was taken visually (generally) every day at some skips. A score between 0 and 4 indicate the estimated fullness of the skip, while a 5 means the skip was overflowing. Four of these are shown in [fig. 2](#fig-raw). [Fig. 2 (a)](#fig-Naizi_inorganic_1_raw) seems to be an ideal profile for data extraction. There are not too many spikes (taken as measurement errors), the ramps are discernible, and there is a good number of them to draw averages. [Fig. 2 (b)](#fig-Bangwe_inorganic_1_raw) presents a bigger challenge, as it is difficult to make out the difference between measurement errors (outliers) and very fast filling of the skip. [Fig. 2 (c)](#fig-Chigumula_inorganic_2_raw) is a timeseries from which filling rates cannot be extracted. Finally, [fig. 2 (d)](#fig-BCA_inorganic_2_raw) only has one useful ramp, from which a filling rate can be computed.

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | (a) Naizi inorganic 1 | | |  | | --- | | (b) Bangwe inorganic 1 | |

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | (c) Chigumula inorganic 2 | | |  | | --- | | (d) BCA inorganic 2 | |

Figure 2: Typical skip filling time series

### Dump arrivals log

A separate time series is provided. It lists arrivals at the Mzedi dump, along with the origin of the skip carried by each truck. The origins match exactly the set of skip locations. The period of this series is 2020-12-05 to 2021-12-31. The sum of arrivals in each week during this period is shown in [fig. 3](#fig-mzedi_arrivals). A sizeable gap is noticeable for almost the whole of February 2021. This is reflected in the skips filling data, where many skips were overflowing and not emptied during this period. The reason for this gap is unknown, but assumed here to be the service simply not operating.

|  |
| --- |
| Figure 3: Mzedi arrivals |

## Data analysis

### Skips filling data

The filling data from individual skips is used to extract an estimated filling rates. To do this, an algorithm is used to fit ramps onto the raw data. The slope of these ramps gives the filling rate, which will be useful in determining the frequency of service. Here, the algorithm is shown for the skip named Naizi Inorganic 1. The algorithm performs a series of forward and backward passes, iterating through the time series data sequentially.

#### 1. Removal of erroneous measurements

In this case, erroneous measurements are quick (upward or downward) spikes that return to the original value. In most series, those spikes interrupt ramps or can be seen in periods of overflow. As the source of these spiking events are unknown, they are assumed to be measurement/recording errors.

#### 2. Ramp tops detection

Since a level 5 encompasses a wide range of quantity of waste, ramps ending in level 4 offer more precision. As such, when a rising trend reaches level 4, the ramp is ended at this point. However, if it skips level 4, the ramp continues to level 5.

The following code blocks show the forward pass, which detects the top of ramps, as well as the backward pass, which brings those top points to the earliest point at which this level occured. This forward/backward approach has the advantage that it can include levels other than 4 and 5 as top levels.

Forward pass:

BEGIN   
INITIALIZE t=0;dt=0  
WHILE t < T:  
 WHILE t + dt < T:  
 IF L(t+dt+1) >= L(t+dt):  
 IF L(t+dt) == 4 AND L(t+dt+1) == 5  
 ADD to top\_ends  
 t = next decreasing level date  
 BREAK  
 ELSE:  
 dt++  
 END IF  
 ELSE:  
 ADD to top\_ends  
 t = t+dt+1  
 BREAK  
 END IF  
 END WHILE  
END WHILE  
END

Backward pass:

BEGIN   
 INITIALIZE t=T; dt=0  
 FOR EACH t in top\_ends:  
 WHILE t - dt - 1 > 0:  
 IF L(t-dt-1) < L(t-dt):  
 ADD to top\_beginnings  
 BREAK  
 ELSE:  
 dt++  
 END IF  
 END WHILE  
 END FOR  
END

#### 3. Removal of derating periods

Next, an effect is observed, whereby a “derating” occurs. This can be seen when the level is at 5, then it drops to 4 for a certain period, then climbs up to 5 again. This might occur for several reasons (non exhaustive list):

* Trash being burned to eliminate the overflowing waste
* Waste being cleaned up by another entity
* An error in data collection

The forward and backward passes detect these oscillations as a top value each time, which creates erroneous ramps later on. These derating tops are eliminated using the following algorithm:

BEGIN   
 INITIALIZE t=T; dt=0  
 FOR EACH t in top\_beginnings:  
 WHILE t - dt - 1 > 0:  
 IF L(t-dt-1) < 4:  
 ADD to top\_beginnings\_no\_derating  
 BREAK  
 ELSE:  
 dt++  
 END IF  
 END WHILE  
 END FOR  
END

#### 4. Ramp bottoms detection

Finally, the bottoms of each ramp is found by walking back from each top until a point where the next increases again. This point is assumed to be the next day after the skip was serviced, and after which the skip begins filling up again. The algorithm is:

BEGIN   
 INITIALIZE t=T; dt=0  
 FOR EACH top\_beginnings\_no\_derating:  
 WHILE t - dt > 0:  
 IF L(t-dt-1) > L(t-dt):  
 ADD to bottom\_beginnings  
 BREAK  
 ELSE:  
 dt++  
 END IF  
 END WHILE  
 END FOR  
END

# Model

# Model optimization

# Methods

## Site selection

## Sample Size

## Ethics

## Experimental Design

### Experiment 1

### Experiment 2

# Results and Discussion

## Discussion

## Tables and Figures

## Tables

## Figures

# Conclusions and Recommendations

# References