Expectation Based Spatio-Temporal Scan Statistic

Chance Haycock & Edward Thorpe-Woods

June 2020

Contents

Notation

Symbol	Description
$\mathcal{D}$	Spatial Domain in $\mathbb{R}^2$
T	Time domain in $\mathbb{R}$
W	Length of Time Domain; $T = \{0, 1, \ldots, W-1\}$
$\mathbf{S}$	Set of space-time regions in $\mathcal{D} \times T$
S	Space-time region in $\mathcal{D} \times T$
$s_i \in \mathbb{R}^2$	Spatial locations of Detectors
c_i^t	Count data at spatial location s_i at time t.
b_i^t	Baseline data found from time series analysis at spatial location s_i at time t.
H_0	Null Hypothesis
$H_1(S)$	Set of alternative hypothesis at space-time region $S \in \mathbf{S}$.
N	Number of partitions per spatial axis

1 Introduction

Here, we introduce our implementation of the *Expectation-Based Scan Statistic* as first described by Daniel Neill[Nei09]. It is hoped that this methodology can be applied to the SCOOT data set, consisting of hourly vehicle count data of roughly 10,000 traffic sensors located in central London. The aim of this work is to implement a framework which enables the detection of emerging space-time clusters in London.

We beginning by first implementing the *frequentist* approach: consisting of a large search over many different shaped regions spanning the domain of interest. Although computationally intensive, it provides a good first step into understanding the usefulness of such a model. Later approaches may include providing prior information and hence the ability to build a Bayesian alternative. The methodology can be split into two disjoint parts: time series analysis and the spatial scan. The former uses historical data to estimate baseline values of the vehicle count data. The model then compares these estimated *baseline* values with the actual recorded *count* data. Our search is then tuned to find regions of space-time that are most likely to represent clustering.

With both types of model, there are four *parameters* of the model which are dependent on the scientific question at the heart of the research; these are

- Which Spatial Region to observe and how to partition the space?
- How far back in time to look for clusters?
- Which method of time series analysis to use?
- Which metric to use? e.g. $F(S)$ defined below.

1.1 Hypotheses of Interest

$$H_0 : c_i^t \sim \text{Po}(b_i^t) \ \forall \ s_i^t \in \mathcal{D} \times T \tag{1}$$

$$H_1(S) : c_i^t \sim \text{Po}(qb_i^t) \ \forall \ s_i^t \in S \text{ and } c_i^t \sim \text{Po}(b_i^t) \ \forall \ s_i^t \notin S \text{ for some } q > 1. \tag{2}$$

2 Time Series Analysis

The premise of expectation scanning is that there is an expected number of counts as forecast by some Time Series Analysis. This is an essential step, as spatial-temporal clusters will nto be recognised

unless we have a baseline for what the *should* be. Daniel Neil outlines some methods of TSA in his original paper, here we repeat those methods which are best suited for out problem with adjustments along with some extra methods which we intend to introduce

2.0.1 Hourly adjusted moving average

when considering TSA let $t = 0$ represent the hour at the start of our forecasting period, and $t > 0$ represent the historical counts, our moving average is defined as follows:

$$b_{i,0}^t = \frac{1}{L_b} \sum_{u=t+1\ldots t+L_b} c_i^u \tag{3}$$

where b_i^t is out baseline forecast at time t at location i. L_b is the number of time intervals we want to look back on to create out moving average, for example if we wanted a daily moving average $L_b = 24$. A standard moving average is obviously a very poorly suited forecast for counts which vary so much depending on their time of day. For that reason we suggest an hourly adjusted moving average, following for the example of D. Neils day of the week adjustment. To do this we consider a local adjustment factor (MALD) β_i^j which is multiplied by the moving average to account for hourly fluctuation. Our hourly MALD factor is calculated as follows:

$$\text{MALD}: \quad \beta_i^j = \frac{\sum_{t=j,j+24,\ldots,j+24\times d} c_i^t}{\sum_{t=1,\ldots,L_b} c_i^t} \tag{4}$$

a simple proportion of counts that occurred at that hour in our look back period, where d is the number of days (or 24 hour periods) in our look back. Thus, our hourly adjusted baseline forecast for time t at detector i is $b_i^t = 24 \times b_{i,0}^t \times \beta_i^j$ this forecasting method accounts for hourly variation well, but misses out on any slower trends such as seasonal trends

2.0.2 Holt-Winters

The multiplicative Holt–Winters' (HW) method is a more dyunamic forecasting method that takes into account smaller periodic trends, as well as long term trends (such as seasonal trends). The Holt–Winters' forecasts are determined by from the historical counts c_i^t by iterating the three equations given below. It has three main components: the smoothed value S_t , trend component Tt , and the periodic component I_t

$$\begin{aligned} S_t &= \alpha \tfrac{c_i^t}{I_{t+24}} + (1-\alpha)\left(S_{t+1} + T_{t+1}\right) \\ T_t &= \beta\left(S_t - S_{t+1}\right) + (1-\beta) T_{t+1} \\ I_t &= \gamma \tfrac{c_i^t}{S_t} + (1-\gamma) I_{t+24} \end{aligned} \tag{5}$$

α, β, and γ are out hyper parameters, which have to be fine tuned for increased accuracy. The baseline estimate for the next hour is then given by $\underline{t}_i = \left(S_{t+1} + T_{t+1}\right) I_{t+24}$. To forecast the whole period you then just continue the iteration forward of the above equations with the baseline estimates instead of counts. This method comes with the major limitation of a large number of hyper parameters (3 for each detector) which must be fine tuned effectively to provide a reliable forecast

2.0.3 LSTM

A more formal approach to forecasting to to use a recursive machine learning method. This provides an extremely accurate forecast, which can be updated straightforwardly with new data. A standard deep learning method for time series prediction is the Long Short-Term Memory Network, with input nodes equal to the number of hours in the lookback period, and output nodes equal to the number of hours in the forecasting period. The LSTM is then trained by iterating through all the historical data. There are two possible approaches

2.1 Missing Values

3 The Scan

3.1 The Naive Approach

Given our spatial domain $\mathcal{D} \subset \mathbb{R}^2$, our first step is to partition the grid into N^2 equally sized rectangles; ideally so that at least one detector sits within each grid cell. We plan to search over all rectangles whose maximum length along a given axis does not span more than $N/2$ partitions. Hence, the spatial search alone is $\mathcal{O}(N^4)$. Combining this with a possible W forecasted time steps, the total run-time complexity of this search is $\mathcal{O}(N^4 W)$.

3.2 The Fast Spatial Scan

The ideas is to compute some metric for each space-time region of the search, and then return to the user regions of space-time which have the most significant value of the metric. Here, we use a Poisson likelihood metric.

3.3 Poisson Likelihood Metric

For the main scan, we search through the different shaped space-time regions, S and compute the value of $F(S)$, defined here as the likelihood ratio of our hypotheses.

$$\begin{aligned} F(S) &:= \frac{\mathbb{P}(\text{data } | H_1(S))}{\mathbb{P}(\text{data } | H_0)} \\ &= \frac{\max_{q>1} \prod_{c_i^t \in S} \mathbb{P}(c_i^t \sim \text{Po}(q b_i^t))}{\prod_{c_i^t \in S} \mathbb{P}(c_i^t \sim \text{Po}(b_i^t))} \\ &= \max_{q>1} \prod_{c_i^t \in S} e^{(1-q)b_i^t} q^{c_i^t} \\ &= \max_{q>1} e^{(1-q)B_S} q^{C_S} \\ &= \left(\frac{C_S}{B_S}\right)^{C_S} e^{B_S - C_S} \end{aligned} \tag{5}$$

with $C_S := \sum_S c_i^t$ and $B_S := \sum_S b_i^t$. Above we use that the maximum value of $F(S)$ is achieved for $q = \max(1, C_S/B_S)$.

In our setting, the values of B_S and C_S, represent the number of cars that passed a certain region of space-time, S. For some areas, these can become too large to compute (5); we therefore scale B_S and C_S down by a factor of 1 million.

3.4 Kulldorf's Metric

3.5 Generalised Likelihood Metric

4 Significance Testing

Now that we have calculated $F(S)$ for each searched space-time region S, we need a way to report on regions which have a significantly large score of $F(S)$. [Nei09] explain how there are two main methods to find the value of F_{thresh} such that we report on all regions with $F(S) > F_{\text{thresh}}$.

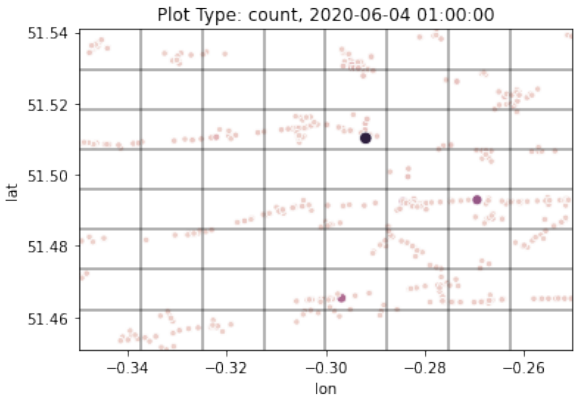

Figure 1: Detector locations in the Ealing area of London. Darker/larger dots signify higher counts of vehicles.

4.1 Randomisation Testing

A naive approach is to first consider all 'actual' counts c_i^t as a Poisson random variable with mean b_i^t. By doing this for all events in $\mathcal{D} \times T$, we can re-run the spatial-scan and find the region S^* such that

$$S^* = \arg\max_{S \in \mathbf{S}} F(S) \tag{6}$$

By simulating this M times, we can build up a distribution of of $F(S^*)$ scores and compare our realised scores with this distribution. This approach is computationally intensive ($\mathcal{O}(N^4WM)$) and prone to high false-positive rates as explained in [Nei09].

4.2 Historical Data Testing

Perhaps a better approach is to exploit historical data, given that it is available. Suppose we are interested in the $W = 24$ case. Neill explains that we can take a year's worth of historical data, and compute the maximum $F(S)$ score for each of the 365 days. Although initially intensive, each day's maximum $F(S)$ can be stored in memory; the current day's score can then be compared against this empirical distribution in relatively little time.

5 Ealing Example

5.1 Setup

Here, we look at an explicit example using data from SCOOT sensors in the area of Ealing between the 4th June 2020 and 5th June 2020. In longitude and latitude co-ordinates, we define

$$\mathcal{D} = (-0.3497, -0.2504) \times (51.4508, 51.5409)$$

and we set $W = 24$; i.e. looking back at the last 24 hours of data for space-time clusters. For a first example, we set $N = 8$ yielding 64 grid cells on the Ealing area. An example plot is shown in Figure 1. Here, $\mathcal{D}$ contains 596 detectors and 14238/14304 valid readings.

5.2 Time Series Analysis

For this particular example, we train a LTSM model to forecast baseline values for the most recent 25 hours. Figure 2 shows how the actual counts and baseline estimates compare over the 24 hours of interest. This procedure takes roughly 30 minutes for this amount of detectors.

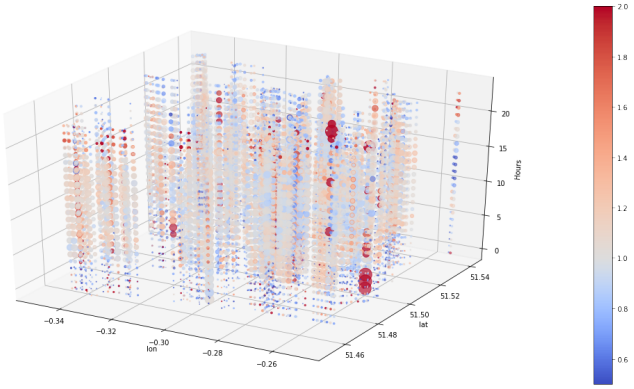

Figure 2: The global space-time region to be searched. The flat plane represents the locations of the detectors in space (lon/lat), and time is displayed upwards. Colour/size represents the ratio of the actual count to the predicted baseline value; i.e. c_i^t/b_i^t.

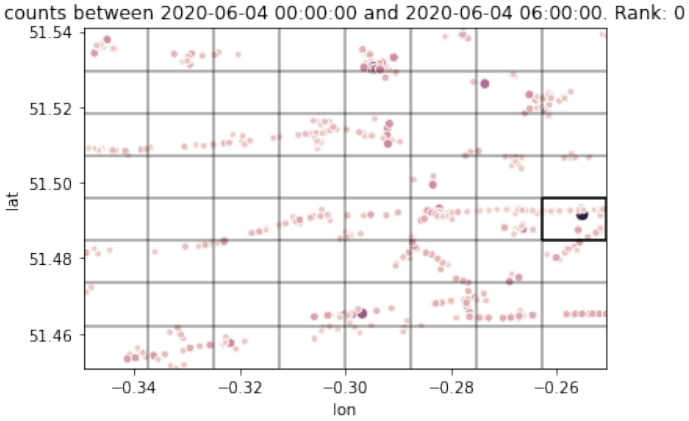

Figure 3: The best scoring region is highlighted in bold. A shown from the title, this occurs during the 12am and 4am on 4th June 2020. With such a large count value, it maybe hat we're looking at an anomoly instead.

5.3 The Scan

An 8x8 grid over 24 hours yields 16224 space-time regions to scan over and compute $F(S)$. This takes roughly 45 seconds on modern laptop in serial. Figure 3 shows the location and time of the highest scoring region $F(S)$. Figure 4 shows the behaviour of these detectors in this spatial region over the time window of interest. It is indeed clear that the scan is flagging an anomalous count from the detector; this is an area of improvement which is currently being updated in the time series analysis stage of the method.

5.4 Significance Testing

As this part of the process is computationally intensive, we present example timings for only 5 simulations.

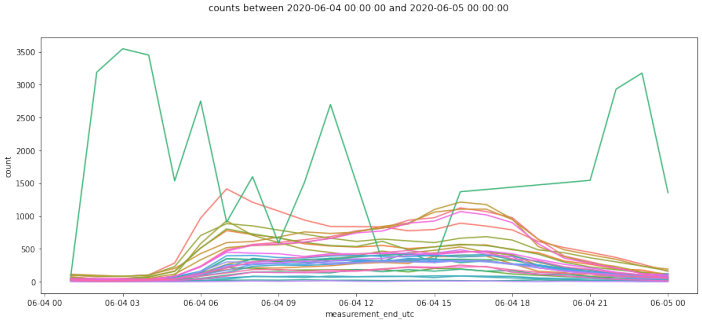

Figure 4: Time series data for the detectors in the highest scoring region highlighted in Figure 3 over the whole prediction period ($W = 24$ hours). Clearly, the scan is flagging an anomalous count from the detector.)

```
Found a grid partition = 8
Searching over the region spanning 24 hours

====================
Beginning Simulation
====================
Performing simulation 5 of 5.
Time Elapsed: 398.434152576 seconds
```

Using 100 simulations results in a run-time of approximately 2.5 hours for this particular region. For comparison, a larger area of London ($\mathcal{D} = (-0.3497, -0.1500)x(51.35067, 51.5499)$) containing 2665 detectors takes roughly 60 seconds to compute the $F(S)$ scores in a scan. Simulating these counts gives the following output:

```
Found a grid partition = 8
Searching over the region spanning 24 hours

====================
Beginning Simulation
====================
Performing simulation 5 of 5.
Time Elapsed: 867.489024811 seconds
```

Extrapolating to 100 simulations gives an estimated run-time of 5 hours. In addition, the time series forecasting for this region takes roughly 4 hours in serial.

References

[Nei09] Daniel Neill. Expectation-based scan statistics for monitoring spatial time series data. *International Journal of Forecasting*, 25:498–517, 07 2009.