A clustering algorithm to organize satellite hotspots data for the purpose of tracking bushfires remotely

by Weihao Li, Emily Dodwell, and Dianne Cook

Abstract An abstract of less than 150 words.

Introduction

Bushfires are a major problem for Australia, and many other parts of the globe. There is concern that as the climate becomes hotter, and drier, that the impact of fires becomes much more severe and extensive. In Australia, the 2019-2020 fires were the worst on record causing extensive ecological damage, as well as damage to agricultural resources, properties and infrastructure. The Wollemi pine, rare prehistoric trees, required special forces intervention to prevent the last stands in the world, in remote wilderness areas, from being turned into ash.

Contributing to the problem is that many fires started in very remote areas, locations deep into the temperate forests ignited by lightning, that are virtually impossible to access or to monitor. Satellite data provides a possible solution to this, particularly remotely sensed hot spot data, which may be useful in detecting new ignitions and movements of fires. Understanding fires in remote areas using satellite data may provide some help in developing effective strategies for mitigating bushfire impact.

This work addresses this topic. Using hot spot data, can we cluster in space and time, in order to determine (1) points of ignition and (2) track the movement of bush fires.

This paper is organised as follows. The next section provides an introduction to the literature on spatiotemporal clustering and bush fire modeling and dynamics. Section Algorithm describes the clustering algorithm, and section Application illustrates how the resulting data can be used to study bush fire ignition.

Background

Spatiotemporal clustering

Bushfire modeling

Algorithm

Data pre-processing

Steps

This algorithm runs in a temporal manner. Starting from the first hour of the first day or the bushfire season, hotspots are grouped, and then agglomerated spatially. This proceeds to the next hour.

1. Divide hotspots by hour

Show faceted plots of hotspot data (from full map) for first five hours, all in one row of plots

Hour is used as the basic unit of time, to simplify the computation, but it could be a different time resolution.

2. Start from the first hour

Plot of one hour, small area, so we can show how the algorithm does grouping

It first selected entries of the first timestamps, which was the first hour in the hotspots data.

3. Connect adjacent hotspots and active centroids (3km)

Show connection of points from previous plot

The algorithm then calculated the matrix of pairwise geodesic distances between all points being selected. With the geodesic distances matrix, an "adjacent distance" as one of the hyperparameters in this algorithm was used to determine the adjacency matrix. If a geodesic distance between two points was less than the "adjacent distance", the corresponding entry in the adjacency matrix would be assigned with integer 1, otherwise it would be assigned with integer 0. Normally, this "adjacent

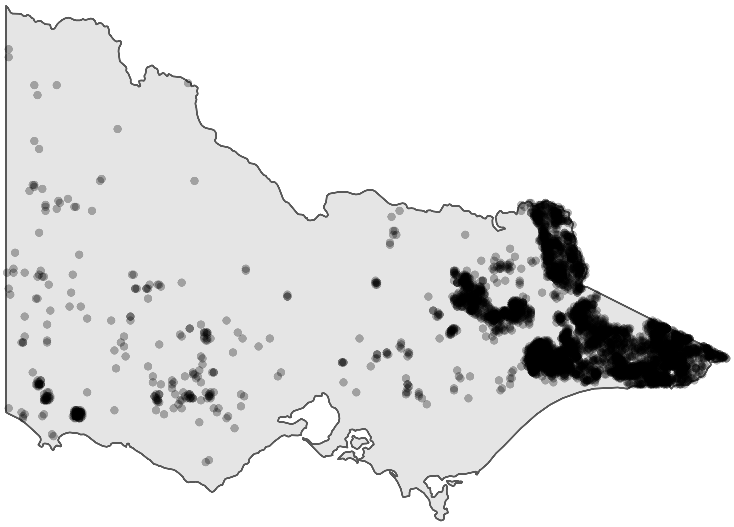


Figure 1: Hotspot locations in Victoria during 2019-2020 season.

distance" would be set between 0 to 100000 meters. Using the adjacency matrix, the algorithm then constructed a undirected unweighted graph. For each connected component in this graph, a unique integer was assigned as the membership. In our hotspots data, components could be recognised as bushfires. Points in the same component shared with the same membership.

4. For each point, if there is a connected nearest active centroid, join its group

Meanwhile, the longitude and the latitude of centroids in each component were calculated by taking the average of longitude and the average of latitude for all points in the corresponding component. Those centroids along with memberships would then be recorded and labelled as active groups. In other words, their "active" attributes were assigned with integer 0.

5. Otherwise, create a new group for each connected graph

Show grouped observations

6. Compute centroid for each group

Show centroids

7. Keep the group active until there is no new hotspots join the group within 24 hours

When the algorithm moved to the next timestamps, it subtracted 1 from "active" attributes. Another hyperparameter "group active time" was used for selecting active groups. Conventionally, "group active time" was set to be 24 hours. If any centroid had an "active" attribute greater than the negative of "group active time", it would be selected as active groups.

For the second and the later timestamps, the algorithm first combined centroids of active groups with the hotspots data in the corresponding timestamps. It then calculated geodesic distances matrix, filled adjacency matrix and constructed graph as before. There was an additional step which was to find the nearest active group within the same component for each point. If a point shared the same component with active groups, it would be assigned with the membership of the nearest active group. Otherwise, points shared with the same component would be assigned with a new membership. Therefore, points in one component would not necessary had the same membership if there were more than one active groups within a component. All centroids of active groups and new group would then be recalculated and updated using only the current timestamps hotspots data. Their "active" attributes were set to be 0.

This algorithm worked till the last timestamps. The end result was a vector of memberships with length equal to number of rows in hotspots data and a time-series record of all groups.

For computational performance, we stored the hostpots data in a SQLite database. Relevant operation was done by using packages `DBI` and `RSQLite`. Geodesic distances matrix was calculated using package `geodist`. Graph operation was done by using package `igraph`.

8. Repeat this process to the last hour

The code implemented this algorithm is "clustering.R".

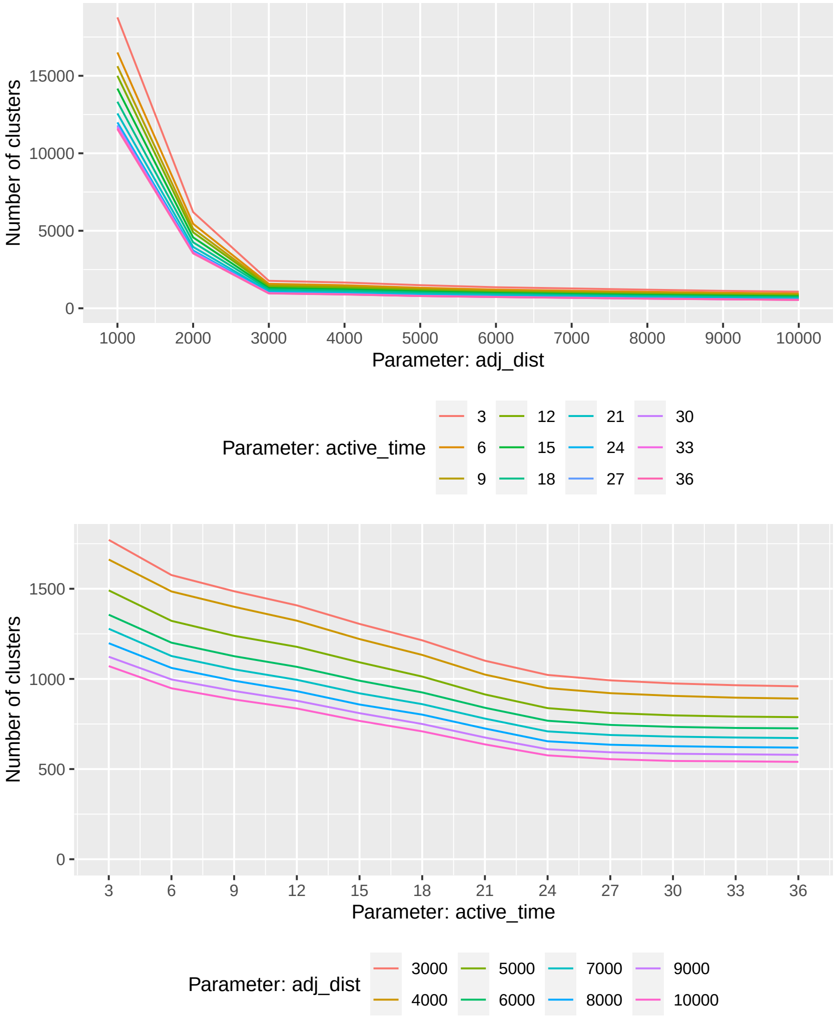


Figure 2: Number of clusters under different parameter choices

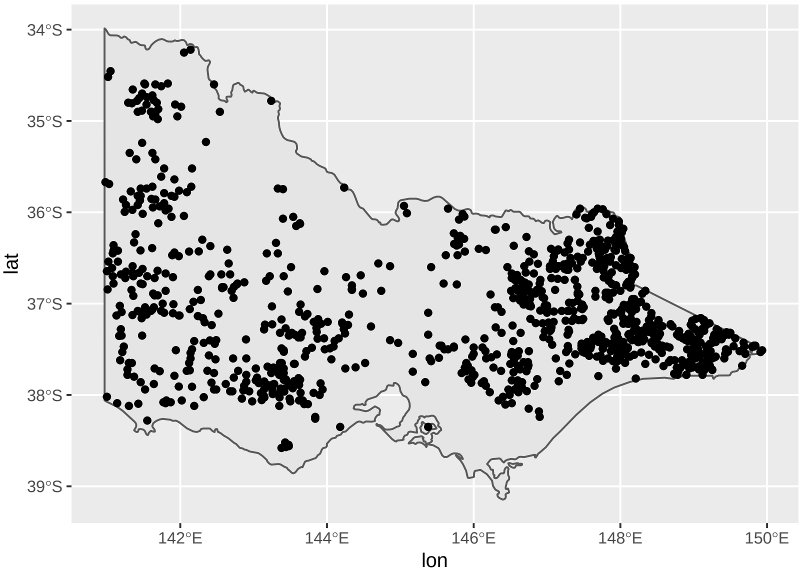
Effects of parameter choices

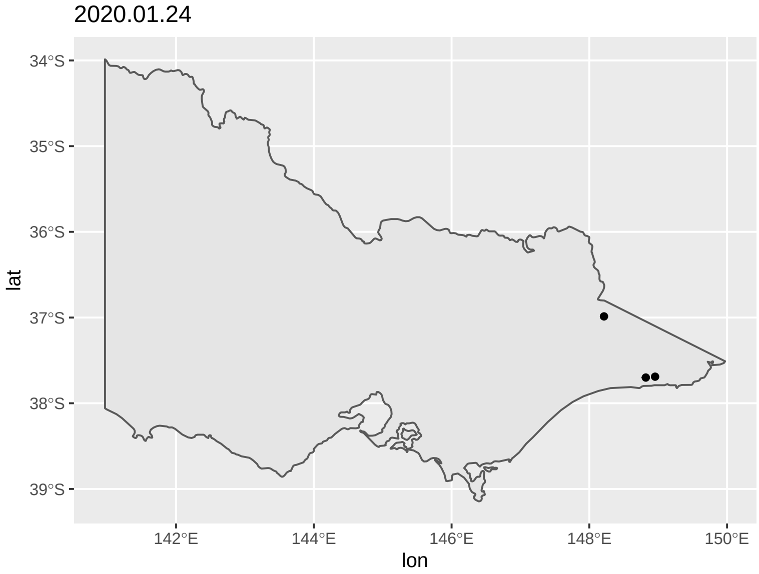
There are two parameters that can be tuned in this algorithm. They are `adj_dist`, which is the density distance and `active_time`, which is the .

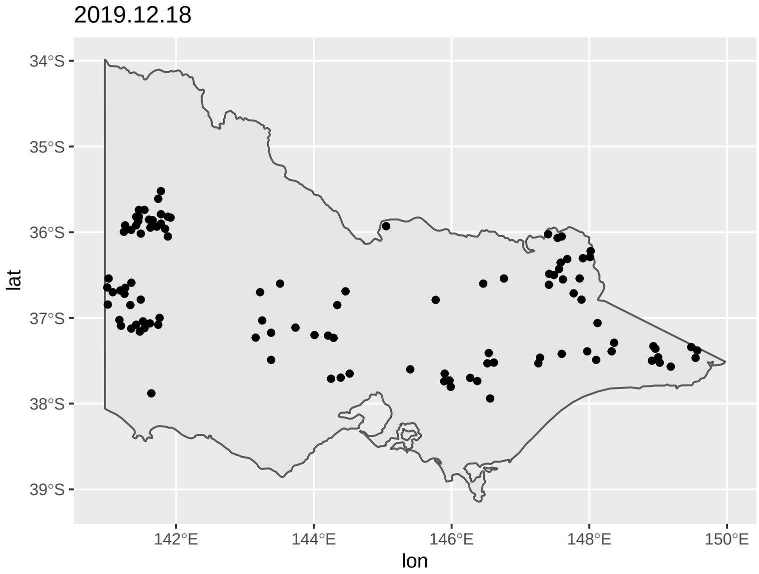
Application

Determining the ignition point and time for individual fires

Show ignition points for a particularly heavy day and another for a particularly light day


34°S
35°S
36°S
37°S
38°S
39°S
lat
142°E
144°E
146°E
148°E
150°E
lon


2020.01.24
34°S
35°S
36°S
37°S
38°S
39°S
lat
142°E
144°E
146°E
148°E
150°E
lon


2019.12.18
34°S
35°S
36°S
37°S
38°S
39°S
lat
142°E
144°E
146°E
148°E
150°E
lon

Tracking fire movement

Display showing how a fire moves over time, maybe two or more fires

Allocating resources for future fire prevention

Merging data with camp sites, CFA, roads, . . .

Summary

Acknowledgements

- The code and files to reproduce this work are at XXX
- Data on hotspots can be downloaded from XXX

Weihao Li
Monash University
line 1
line 2

wlii0039@student.monash.edu

Emily Dodwell
AT&T
line 1
line 2

emily@research.att.com

Dianne Cook
Monash University
line 1
line 2

dicook@monash.edu