Journal of Geographical Systems
https://doi.org/10.1007/s10109-021-00350-w

ORIGINAL ARTICLE

Open-source approaches for location cover models: capabilities and efficiency

Huanfa Chen[1] · **Alan T. Murray**[2] · **Rui Jiang**[3]

Received: 14 May 2020 / Accepted: 1 April 2021

Abstract
Location cover models are aimed at siting facilities so as to provide service to demand efficiently. These models are crucial in the management, planning and decision-making of service systems in public and private sectors. As a result, location cover models have been incorporated in a range of GIS tools, either closed or open source. Among them, open-source tools are advantageous due to transparency and reproducibility. Nonetheless, the capabilities and limitations of location cover tools remain largely unknown, necessitating further investigation and assessment. To this end, this paper provides an overview of the open-source tools that are capable of structuring and solving location cover models. Case studies are provided to demonstrate access of location models through different open-source tools as well as exploring solution quality, scalability, computing performance and reproducibility. Directions for improving location cover models accessible through open-source tools are summarized based on this review.

Keywords Spatial optimisation · Location cover model · Open source · GIS

JEL classification C61 · C88 · Q15

1 Introduction

Location modelling is crucial to service system planning for both public and private sectors. In the public sector, the location choices of service facilities involve not only fiscal responsibility but also social good and accessibility (White 1979).

✉ Huanfa Chen
huanfa.chen@ucl.ac.uk

[1] Centre for Advanced Spatial Analysis, University College London, London, UK

[2] Department of Geography, University of California at Santa Barbara, Santa Barbara, CA, USA

[3] Department of Civil, Environmental and Geomatic Engineering, University College London, London, UK

Published online: 21 April 2021

Local governments, as an example, provide a range of services, such as libraries and schools, where the intent is to maximise accessibility to people in the community. In the private sector, retail outlets are concerned with locating stores and distribution centres to best serve customers and optimise revenue.

Location modelling, a component of spatial analytics, has been supported in geographic information systems (GIS). Commercial GIS packages such as ArcGIS and TransCAD explicitly formulate and solve location models using heuristic methods (Murray et al. 2019). For example, locations models within ArcGIS are solved using the Teitz and Bart heuristic (ESRI 2020) or GRASP heuristic. A major advantage of using a heuristic for solving a location problem is that it requires less time and fewer computing resources to identify a solution (Church and Murray 2009; Tong and Murray 2012). Nevertheless, for most heuristics, there is no guarantee that they will derive optimal or even good solutions. Moreover, even if a heuristic has good performance for one problem instance, the performance does not generalise when this heuristic is applied to another instance of the same type.

The inclusion of location analytics within GIS software has promoted the widespread use of location models in various contexts, including solid waste facilities (Khan et al. 2018), fire stations (Adesina et al. 2017) and health care centres (Jankowski and Brown 2014). A summary of these applications can be found in Murray et al. (2019). Yet, this software has technical limitations, with location model results proving to have uncertain solution quality as well as being computationally inefficient (Murray et al. 2019).

More concerning is the reproducibility of location models through commercial GIS. By definition, the reproducibility of a study "refers to the ability of a researcher to duplicate the results of a prior study using the same materials as were used by the original investigator" (Bollen et al. 2015). There are three major components to a reproducible study (Leeka and Peng 2015): first, the data are available. Second, the code used and the documentation is available. Third, a correct analysis has been performed. A detailed discussion of reproducibility in geographical analysis can be found in Kedron et al. (2019).

Specifically, location models in commercial GIS are rarely reproducible. The code of these models is not open. Further, the heuristics used within ArcGIS and TransCAD involve many parameters that affect speed and solution quality. However, the technical details in the heuristics used are hidden from users, and there are no mechanisms for users to know or interact with the parameters.

Because of the technical limitations of location models in commercial GIS, researchers have written custom software, often relying on an optimisation solver that can derive optimal solutions (Tong et al. 2009; Tong and Wei 2017). The benefits are that optimal solutions can be identified if such solutions exist, and that modification of models is possible. The custom software, however, cannot necessarily guarantee reproducibility, because the code is often not open for other users. This makes it challenging to reproduce reported approaches and results.

Open-source software is a more promising option for using location models. Specifically, open-source software has obvious advantages. First, under open source, the transparency and reproducibility of a method are guaranteed, eliminating the "black box" that hides implementation details (Rey 2009, 2018; Singleton

et al. 2016). Second, open-source software assures wide access to and full control of the source code for audit and modification, enabling redistribution of code. Third, it facilitates advanced user extension of software to fit their requirements as necessary. Moreover, although open source does not necessarily mean free of charge, most open source software can be used at no or a low price.

Spatial analysts have been embracing open-source access to algorithms and software. Open-source examples that support spatial analysis include GRASS GIS (GRASS Development Team 2017) and QGIS (QGIS Development Team 2009), but also PySAL (Rey and Anselin 2007) and GeoDa (Anselin et al. 2006). Buliung and Remmel (2008) and Rey (2009, 2018), among others, argue that open-source has the potential to revolutionise and enhance geospatial research and education and will play a critical role in facilitating discussions on methodological workflows for spatial analytics. Steiniger and Hunter (2013) propose a set of evaluation criteria in the selection of open-source GIS software, highlighting important considerations such as functionality, usability and customization options. The above discussion reinforces that it is crucial to understand function and performance of the open-source tools before deploying them.

Open-source alone is insufficient to guarantee reproducibility in location modelling. Available packages differ considerably with respect to model assumptions and capabilities. Proper model specification and implementation is critical, as is solution approach. While usage and application of these methods across a range of disciplines has increased dramatically in recent years (Murray et al. 2019; Xu et al. 2020) and open-source tools for location modelling are continuing to mature, what is significant at this stage is a lack of critical review of the capabilities of these packages. This study investigates open-source tools for applying coverage models. Of particular interest are assumptions, representation and computational efforts. Assessment and comparison against the methods available in commercial GIS software too are of interest. As most reported applications of location models are done using commercial GIS, as noted by Murray et al. (2019) and Xu et al. (2020), and we anticipate that open-source options would likely become popular in applications, we should compare both options in terms of their function and performance.

This paper is built upon previous work (Chen and Murray 2021) and distinguishes itself by making several noteworthy contributions to open source location modelling. First, we systematically compare relevant open source options for location modelling using two case studies. Second, we substantially extend the function of two important open source location models. Third, we discuss the conditions that guarantee optimal solutions for a location cover model and point out future directions in this field.

This paper is organised as follows. The background of the research is described in the next section. This is followed by an overview of coverage models to be investigated. A detailed comparison of location cover tools is then provided. After this, planning applications are presented to highlight the differences in modelling capacities. Finally, this paper ends with recommendations for future research in open-source developments to support location cover modelling.

2 Background

A range of location models have been proposed to support different management, planning and decision-making contexts. The focus of this paper is location cover models, which are aimed at siting facilities in order to best service demand. Location cover models have seen broad use in a range of applications, including bike-share stations (García-Palomares et al. 2012), disease treatment centre (Oviasu 2014), veterinarian clinics (Polo et al. 2015) and solid waste facilities (Khan et al. 2018). In particular, there has been an increasing interest in two location cover models: one is the location set covering problem (LSCP) and the other is the maximum covering location problem (MCLP). The LSCP (Toregas et al. 1971) involves planning applications in which the fewest facilities are to be sited so as to serve all demand within the designated service response standard of time or distance. The MCLP (Church and ReVelle 1974) was introduced to determine the locations of facilities of a fixed number that serve the most potential demand within the service standard. Since their appearance, these models have been applied and extended in various fields (see Church and Murray 2018).

Because of their prominence, both LSCP and MCLP have been implemented in a range of software tools. While there are more than ten categories of software (GNU Project 2019), three categories have been identified for tools that provide location cover models, as shown in Table 1.

Location models are available in proprietary GIS software, including ArcGIS and TransCAD. While the use of location models through proprietary GIS is common, limited attention has been given to solution quality and reproducibility issues. These commercial tools rely on heuristics for solving location models, generally not providing technical details or solution quality assurance. This point has been noted by Woodhouse et al. (2000), who reported that ArcGIS did not derive the optimal solution for the LSCP instances. Murray et al. (2019) also found that ArcGIS and TransCAD failed to optimally solve most of the 1059 MCLP and LSCP problems examined. Moreover, the heuristics used by ArcGIS and TransCAD involve parameters that influence speed and solution quality, yet users are prevented from knowing or interacting with these parameters (Church 2002; Murray 2010). Users are also forbidden from knowing or controlling the parameters of pre-processing steps in TransCAD, such as assigning points to a network (Gaboardi et al. 2020). In practice,

Table 1 Three types of software tools for location cover models

Type	Definition	Code availability
Proprietary software	Software whose use or redistribution is restricted in the way that users cannot do it without permission	Not publicly available
Custom software	Software developed for one organisation or company	Often not publicly available
Open-source software	Codes that anyone can inspect, apply, modify and enhance	Publicly available

the lack of transparency and solution quality assurance raises issues of significance in reported findings.

Some researchers have written custom software that directly structures and solves location models (Tong et al. 2009; Tong and Wei 2017), relying on general optimisation solvers. While custom software allows developers to specify their models, it comes with obvious shortcomings. It is challenging to develop new software, which requires specific technical skills—software design, tackling data input, model implementation and integration with solvers. Moreover, in academia, researchers generally neither disclose the technical details nor release the code of developed software. Consequently, it is hard to reproduce, validate and reuse location models found in custom software.

Open-source software is well-suited for the use of location cover models. Generally, open-source software is based on code that anyone can inspect, apply, modify and enhance. In particular, several open-source tools providing location cover models have been released. Among them, PySpatialOpt (Pulver 2019), a Python package, provides access to a number of cover models, including MCLP and LSCP. It relies on an optimisation solver to solve the problem. Another is Maxcovr (Tierney 2019), an *R* package, that formulates and solves MCLP and LSCP. As with PySpatialOpt, it uses an optimisation solver to derive optimal solutions. Another example is FLP Spreadsheet Solver (Erdoğan 2019; Erdoğan et al. 2019) that is based on Microsoft Excel and Visual Basic, which is not free of charge and requires an Microsoft Office license to use. Unlike PySpatialOpt and Maxcovr, this tool relies on a tabu search heuristic to solve the MCLP and LSCP. Some tools related to location models are excluded from this study for various reasons. One is SITATION (Daskin 2002), which solves five classes of location problems, including MCLP and LSCP. This software is free for use but not open-source and is therefore beyond the scope of this study. There are some open-source packages that solve the p-median problem, including orloca (Munoz-Marquez 2019) and tbart (Brunsdon 2015). Although both MCLP and LSCP can be transformed into an equivalent p-median problem (Church and ReVelle 1976) and then solved by these tools, they are not directly applicable to location cover problems.

3 Methods

As mentioned above, of interest in this paper are the LSCP and MCLP as representative location models that have been broadly applied. This section gives their mathematical formulations along with specifics for application context and extension. The following notation is defined as:

i: index of demand units (total n).
j: index of potential facility sites (total m).
d_{ij}: distance or travel time from facility site j to demand i.
S: maximum service standard of distance or time.
N_i: set of potential facility sites that can cover demand i within the service standard (e.g. $\{j|d_{ij} \leq \mathrm{S}\}$).

$$x_j = \begin{cases} 1 & \text{if a facility is sited at location} j \\ 0 & \text{otherwise} \end{cases}$$

Using this notation, the LSCP is as follows based on Toregas et al. (1971):

$$\text{Minimise} \quad \sum_j x_j \tag{1}$$

$$\text{Subject to} \quad \sum_{j \in N_i} x_j \geq 1 \quad \forall i \tag{2}$$

$$x_j \in \{0, 1\} \quad \forall j \tag{3}$$

The objective (1) seeks a minimal number of facilities. Constraints (2) require that each demand be covered by one or more sited facility. Constraints (3) require that the decision variables be binary.

The LSCP assumes that resources are sufficient to cover all demand, with the number of facilities the only factor in the objective (Church and ReVelle 1974). However, when resources are lacking to serve all demand, decision makers may attempt to cover the greatest amount of demand possible within their budget. This is the consideration of MCLP, which relaxes the hard constraint in LSCP that all demand be served. Additional notation is as follows:

a_i: estimated demand at location i
p: the number of facilities to be sited
$y_i = \begin{cases} 1 & \text{if demand } i \text{ is covered} \\ 0 & \text{otherwise} \end{cases}$

The formulation of the MCLP based on Church and ReVelle (1974) is:

$$\text{Minimise} \quad \sum_i a_i y_i \tag{4}$$

$$\text{Subject to} \quad \sum_{j \in N_i} x_j \geq y_i \quad \forall i \tag{5}$$

$$\sum_j x_j = p \tag{6}$$

$$y_i \in \{0, 1\} \quad \forall i \tag{7}$$

$$x_j \in \{0, 1\} \quad \forall j \tag{8}$$

The MCLP objective (4) seeks the maximal total demand that can be covered by the facilities located. Constraints (5) account for coverage of a demand location. Constraints (6) require that exactly p facilities be sited. Constraints (7) and (8) impose binary restrictions on decision variables.

There are numerous variants and extensions of the LSCP and MCLP (Church and Murray 2018). The spatial nature of the LSCP and MCLP lies not only in the variables x_j that correspond to the choice of facility locations, but also in the set N_i as it defines the subset of facilities that serve demand i within the service standard. One common form of this standard is the maximal service distance, which relies on the travel space (Cartesian or network) as well as movement potential (geodesic, Euclidean, rectilinear, network, etc.) of service providers. In addition, model variants have been introduced due to the representation of demand. Demands, as spatial features, can be in the form of points, lines or polygons (Church and ReVelle 1976; Murray and Wei 2013), and the coverage to the demand can be either binary or partial. Binary coverage requires that a demand is either covered or not covered by any facility. On the other hand, under partial coverage, a demand can be partially covered by a facility, possibly as a function of area served (Murray 2005; Tong and Murray 2009). Moreover, demands can have a uniform or variable weight. Another important extension has been the introduction of facility capacities. The capacity constraint, which requires the demand covered by a facility should not exceed its capacity, is essential in applications where facilities have limited potential to provide service. The formulations of the capacitated LSCP and capacitated MCLP are summarised in Church and Murray (2018). It is worth noting that the inclusion of capacity constraints on facilities makes the problem much more complicated and challenging to solve (Xu et al. 2020).

4 Access and solutions in location cover tools

There are multiple steps in an analysis that uses a location cover model, as presented in the workflow in Fig. 1. Using this basic workflow, a detailed comparison of different location cover modelling tools is possible. This comparison now follows and is summarised in Table 2.

4.1 Model type

All of the four tools discussed in Table 2 (PySpatialOpt, Maxcovr, FLP Solver, ArcGIS) provide access to MCLP and LSCP, and some tools also provide entry to other location models. For example, PySpatialOpt offers access to not only the LSCP and MCLP but also backup coverage location problem, threshold model, complementary coverage threshold model and the trauma resource allocation model (Pulver 2019). Further, FLP Solver also addresses the p-centre and p-median problems (Erdoğan 2019). In addition, these tools differ on dealing with location problems without feasible solutions. For example, when tackling an LSCP in which not all demand can be covered by the given facilities, ArcGIS would return a solution that ignores demand

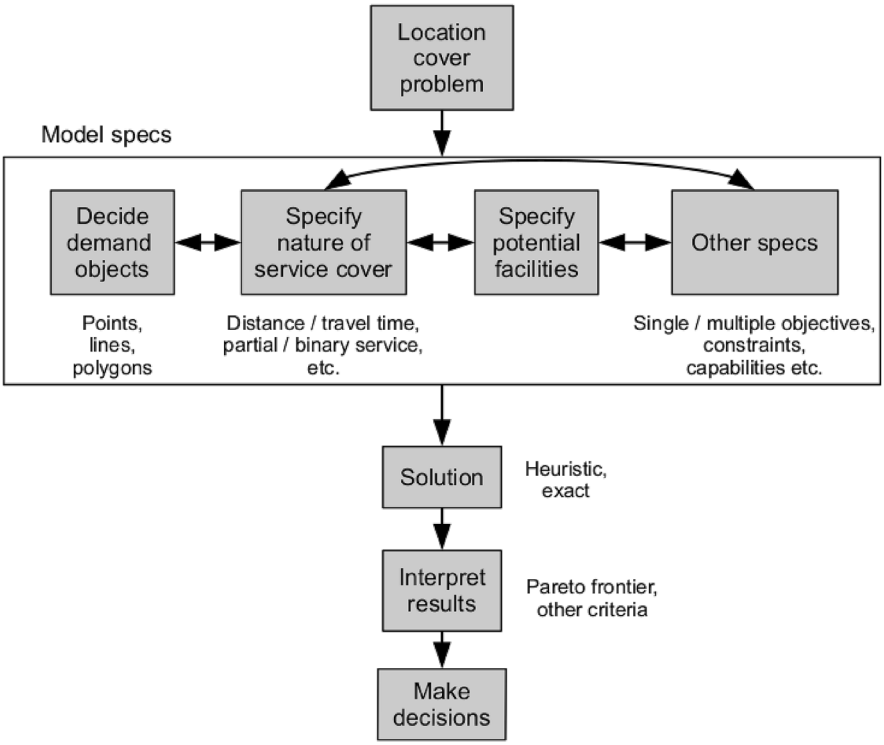

Fig. 1 The workflow of using a location cover model

Table 2 A comparison of different tools that support location cover modelling

	PySpatialOpt	Maxcovr	FLP Solver	ArcGIS
Model type	MCLP, LSCP others	MCLP, LSCP	MCLP, LSCP others	MCLP, LSCP
Allowing for facility capacity	No	No	Yes	Yes
Demand object shape	Point, polygon	Point	Point	Point
Demand weight	Variable	Variable	Variable	Variable
Space	Unrestricted	Unrestricted	Unrestricted	Road network space
Distance metric	Unrestricted	Unrestricted	Unrestricted	Network distance
Solution approach	Exact	Exact	Heuristic	Heuristic

not covered by any potential facility site. Such a problem is technically infeasible as LSCP requires all demand be covered, but ArcGIS deals with this in an ad hoc fashion. This could mislead users. In contrast, other tools would inform the user that there is no feasible solution.

4.2 Capacity

In location modelling, facilities may or may not have capacities. One common assumption in PySpatialOpt and Maxcovr is that each facility has an unlimited capacity, meaning there is no mechanism for addressing facility capacity. In contrast, ArcGIS and FLP Solver allow for specifying a capacity for each facility. As noted previously, the addition of capacity constraints likely increases problem difficulty in solution.

4.3 Demand unit shape and weight

Demand units are spatial features in the form of points, lines or polygons, which relates to how space is abstracted. According to Tong and Murray (2009), the MCLP is sensitive to the abstraction of space (as points or polygons), and uncertainty would be introduced in the approximation of geographic space. While point-based demand units are accepted by all tools discussed, polygon-based demand is only allowed in PySpatialOpt. Two coverage types for polygon-based demand can be found in PySpatialOpt: binary and partial. With binary coverage, the coverage is either zero or one, and a facility covers a demand unit when the entire demand polygon is contained by the service area of the facility. With partial coverage, the coverage is proportional to the intersection between the service area of a facility and the demand polygon. For the MCLP, demands can have different weights a_i, which is true for the models of ArcGIS, FLP Solver and PySpatialOpt.

4.4 Distance metric

The distance metric or travel time d_{ij} from demand unit i to potential facility site j is of central importance for location cover models. In particular, ArcGIS needs a transport network in order to structure and solve location models. For problems that are not network-based, a (complete) network has to be created in advance to reflect the distance metric used. Although the network created can reflect all distance metrics, in theory, this complete network would require more computational efforts by increasing needed computer memory and computational time. Maxcovr accepts a user-defined distance matrix as input, and if no distance matrix is provided, it supports only the Haversine formula distance detailed in Sinnott (1984). The Haversine distance is the distance between two points on the greater circle of the earth sphere (assuming that the earth is a sphere):

$$d = 2r\sin^{-1}\sqrt{\sin^2\left(\frac{\Phi_2 - \Phi_1}{2}\right) + \cos\left(\Phi_1\right)\cos\left(\Phi_2\right)\sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)} \tag{9}$$

where r denotes the earth radius; (Φ_1, λ_1) and (Φ_2, λ_2) are the latitude–longitude coordinates of the two points. PySpatialOpt does not contain a distance computing module, so the distance needs to be computed externally. This gives users

considerable flexibility in defining the distance metric between demand and facilities. Finally, FLP Solver provides the greatest flexibility in distance specification. It not only allows for user-defined distance, but also offers a range of distance options: Round Euclidean, Euclidean, Manhattan, big circle and the driving distance on Bing Maps.

4.5 Solution approach

In ArcGIS, location cover models are solved by a three-step heuristic method (ESRI 2017). A major issue with the heuristic in ArcGIS is that it is unknown whether it would yield optimal solutions for a problem. A recent study indicated that the heuristic in ArcGIS fails to identify optimal solutions for 50% of the location planning problems examined (Murray et al. 2019), though the deviation from optimality was generally low. As with ArcGIS, location models accessible in FLP Solver are solved using tabu search, a heuristic method. Tabu search has been applied in an early study that determines the optimal locations for ambulance bases (Adenso-Díaz and Rodríguez 1997). There is also no guarantee that tabu search will derive the optimal solutions for location cover models. Noteworthy is that the parameters of tabu search in FLP Solver are revealed and well explained, potentially aiding reproducibility. In contrast, both PySpatialOpt and Maxcovr rely on optimisation solvers that would derive optimal solutions for location models, if such solutions exist. Specifically, Maxcovr supports three solvers, i.e. lp_solve, Gurobi and GLPK, by directly using a solver's API. On the other hand, PySpatialOpt uses PuLP, which is a Python-based linear programming modelling environment and can connect to several solvers including lp_solve, CPLEX, XPRESS, Gurobi, GLPK, etc. Using modelling environments is preferable to directly using a solver, as these environments allow problems to be written in a concise and easy-to-read manner (Gearhart et al. 2013). In addition, these environments can connect to multiple solvers, thereby providing the flexibility between competing solvers.

5 Case studies

Two case studies are presented to compare the function and performance of the four tools being discussed. The PySpatialOpt and Maxcovr are backed by GLPK (GNU Project 2017), an open source optimisation solver. Although GLPK is not necessarily the most efficient solver (Meindl and Templ 2012; Gearhart et al. 2013), it is appropriate for comparing the performance of PySpatialOpt and Maxcovr against ArcGIS and FLP Solver.

All processing and computation are conducted on a remote desktop computer (Intel Xeon E5 CPU, 2.7 GHz with 256 GBytes memory). The Python (version 2.7) and R (version 3.4.0) programming languages are used, as both PySpatialOpt and ArcPy (version 10.6) supports only Python 2.7 instead of Python 3.6. The parameter settings in the tabu search heuristic used by FLP Solver are listed in Table 3. These are the default values in the documentation (Erdoğan et al. 2019) except for the time

Table 3 Parameter settings of FLP Solver in this study

Parameter	Value
Tabu tenure (the maximum duration of a location in the tabu list)	Equal to the number of facilities to locate (denoted as p)
Probability of removing all locations in the tabu list in each step	0.005
Time limit of tabu search	60 seconds

limit of 60 seconds. This new time limit was chosen as it led to optimal solutions for the MCLP and LSCP instances in the San Francisco case. For comparative purposes, results are also provided for ArcGIS methods called via ArcPy (version 10.6). The code for the case studies is fully open source and hosted on Github (Chen 2021).

5.1 Store location planning in San Francisco

A retail chain would like to choose store locations in San Francisco (see Fig. 2) that would maximise business. The primary objective is to locate stores close to population centres, based on the assumption that people tend to shop more at nearby stores than at ones that are far-away. The centroids of the 205 census tracts in this city are used as demand for service, with the population in the year of 2000 as demand weight, which totals 955,113. A set of 16 potential store sites are considered. The maximum service distance to access a store is assumed to be five kilometres on the road network. A facility-demand distance matrix was derived from ArcGIS Network Analyst extension, in which both census tract centroids and potential store sites were presumably snapped to the road network. This distance matrix serves as input to PySpatialOpt and Maxcovr. Here, ArcGIS was chosen to generate this matrix rather than open source software like QGIS, because this guarantees that the same pre-processing step of snapping points to network is used in the location models in ArcGIS and in generating a distance matrix.

Table 4 summarises the findings for applying the LSCP to derive the minimal number of stores needed to cover all census tracts in both studies, with the optimal facility numbers in bold. In this case, eight stores are suggested by all four packages as the minimum for coverage of all demand. Computationally, PySpatialOpt and Maxcovr are considerably more efficient than the ArcGIS method and FLP Solver.

The MCLP was applied to assess the impacts and coverage of fewer facilities. The possibility of siting between one and twelve stores was considered, and the findings are summarised in Table 5. Optimal coverage values and the shortest computation time are indicated in bold. Among all 12 problems, solutions from all four tools led to the same and optimal coverage.

Figure 2 shows the optimal spatial configuration of stores for $p=4$ from PySpatialOpt. Solution time for the MCLP varied, with ArcGIS requiring 12.66 s on average, and PySpatialOpt and Maxcovr taking only 0.24 and 0.01 s. The solution time of FLP Solver is close to the time limit of 60 s.

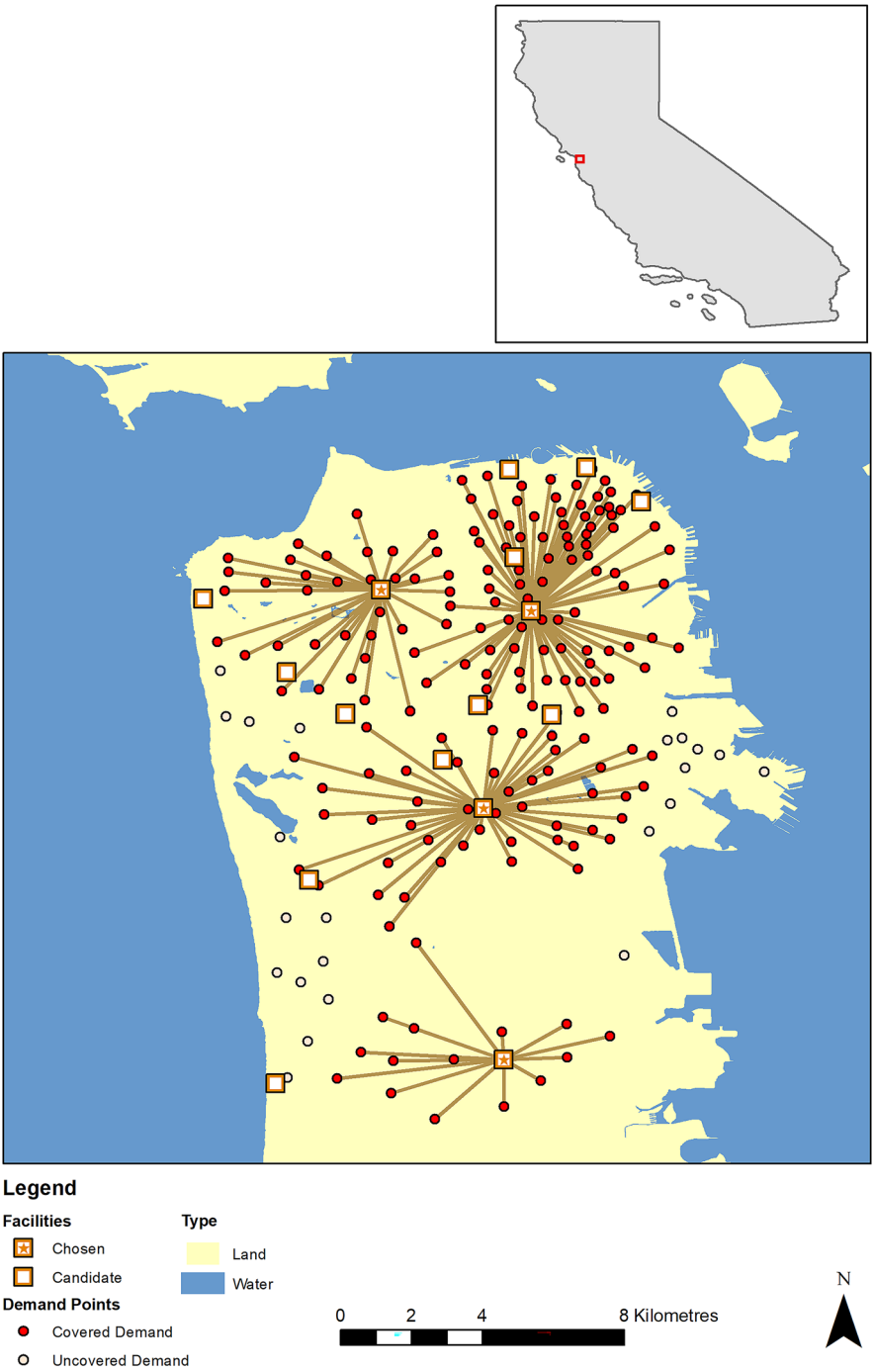

Fig. 2 MCLP derived store configuration ($p=4$) and coverage for serving the San Francisco area. The top right inset shows the location of the study area in California

Table 4 LSCP results by case study

Case study (n, m)*	Number of facilities needed (computation time in seconds)			
	ArcGIS	PySpatialOpt	Maxcovr	FLP_Solver
San Francisco (205, 16)	**8** (12.8)	**8** (1.3)	**8** (0.01)	**8** (60.0)
York City (591, 1821)	23 (18,591)	**22** (2.2)	**22** (0.02)	NA

*n and m represent the number of demand units and candidate facility locations, respectively

Table 5 MCLP results for San Francisco store case study

p	Percent demand covered (%)				Computation time (s)			
	ArcGIS	PySpatialOpt	Maxcovr	FLP	ArcGIS	PySpatialOpt	Maxcovr	FLP
1	**46.9**	**46.9**	**46.9**	**46.9**	12.3	0.2	**0.01**	64.1
2	**70.4**	**70.4**	**70.4**	**70.4**	12.4	0.2	**0.01**	63.7
3	**82.9**	**82.9**	**82.9**	**82.9**	13.8	0.2	**0.01**	64.2
4	**91.6**	**91.6**	**91.6**	**91.6**	12.6	0.2	**0.01**	63.6
5	**97.1**	**97.1**	**97.1**	**97.1**	12.4	0.2	**0.01**	65.0
6	**98.6**	**98.6**	**98.6**	**98.0**	12.6	0.2	**0.01**	65.8
7	**99.5**	**99.5**	**99.5**	**99.5**	12.5	0.2	**0.01**	64.1
8	**100.0**	**100.0**	**100.0**	**100.0**	12.7	0.2	**0.01**	65.0
9	**100.0**	**100.0**	**100.0**	**100.0**	12.8	0.2	**0.01**	65.8
10	**100.0**	**100.0**	**100.0**	**100.0**	12.4	0.2	**0.01**	65.8
11	**100.0**	**100.0**	**100.0**	**100.0**	12.5	0.2	**0.01**	64.0
12	**100.0**	**100.0**	**100.0**	**100.0**	12.6	0.2	**0.01**	69.1
Average	NA*	NA	NA	NA	12.6	0.2	**0.01**	65.0

*NA means 'Not Applicable'

5.2 Police surveillance towers in York

The second case study involves location planning to support police surveillance tower placement in order to guarantee rapid detection and response to crimes in the city of York, UK. Examined here were 591 crime incidents in a subarea of York (see top right inset in Fig. 3) during September 2016 (Odell 2019). The analysis considers each building with GPS coordinates as a potential tower site, totalling 1821 sites. The response distance standard was assumed to be 200 m, and the goal is to site surveillance towers in the most efficient way. Given the nature of service response and the needs for efficiency, the LSCP and MCLP represent meaningful yet different approaches for the tower siting problem. As the Haversine distance is used, a complete network reflecting Euclidean distance was structured for use in ArcGIS. FLP Solver was unable to solve this case, as there were 2412 locations involved (include demand and potential facility) and therefore over five million rows of pairwise distance, whilst the Excel software only

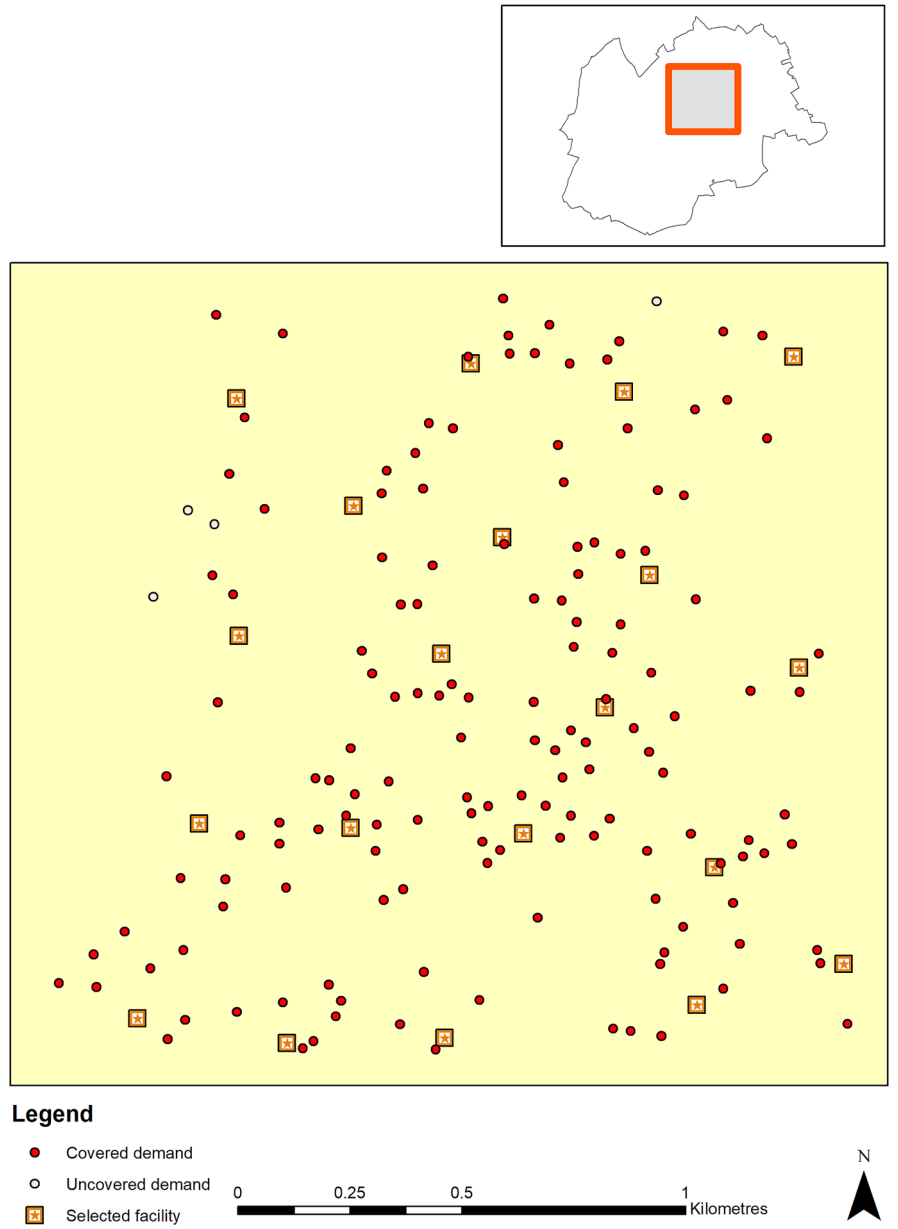

Fig. 3 MCLP derived surveillance tower configuration ($p=20$) and coverage for monitoring crimes in a subarea in the City of York, England. The top right inset shows the location of the study area in York

allows up to 1,048,576 rows (Microsoft Corporation 2018). This demonstrates one limitation of the FLP Solver.

Table 4 summarises the findings for using the LSCP to derive the minimal number of surveillance towers needed to cover all crime points in York. In this case, PySpatialOpt and Maxcovr suggested 22 towers were the minimum for coverage of all demand, whereas ArcGIS proposed 23 towers. Computationally, the solution times required by ArcGIS were significantly longer than that of PySpatialOpt and Maxcovr. The MCLP using three solvers was applied in order to access the impacts of locating fewer towers, which is summarised in Table 6. Optimal coverage values and the shortest computation time are indicated in bold. Of all 22 problem instances, both the PySpatialOpt and Maxcovr identified optimal configurations, and the ArcGIS solver identified only eight optimal solutions. The spatial configuration for 20 facilities by PySpatialOpt is shown in Fig. 3, which covers 585 incidents, whereas the identified configuration using ArcGIS covers only 581 incidents. The solution time required using ArcGIS (on average 21,483 s) was obviously longer than that of

Table 6 MCLP results for the York Tower case

p	Percent demand covered (%)			Computation time (s)		
	ArcGIS	PySpatialOpt	Maxcovr	ArcGIS	PySpatialOpt	Maxcovr
1	**24.0**	**24.0**	**24.0**	19,236.19	2.42	**0.01**
2	**39.9**	**39.9**	**39.9**	19,374.94	2.44	**0.03**
3	**50.3**	**50.3**	**50.3**	19,155.79	2.44	**0.01**
4	**58.5**	**58.5**	**58.5**	19,085.13	2.44	**0.01**
5	63.5	**65.5**	**65.5**	24,519.83	2.42	**0.02**
6	71.1	**71.4**	**71.4**	23,598.43	2.42	**0.01**
7	**75.5**	**75.5**	**75.5**	23,466.37	2.44	**0.01**
8	**78.7**	**78.7**	**78.7**	23,283.16	2.49	**0.01**
9	81.2	**81.7**	**81.7**	23,247.36	2.50	**0.01**
10	**84.4**	**84.4**	**84.4**	21,775.56	2.72	**0.01**
11	86.8	**87.0**	**87.0**	22,121.83	2.64	**0.01**
12	89.0	**89.2**	**89.2**	21,866.81	5.41	**0.02**
13	**91.2**	**91.2**	**91.2**	20,059.26	2.70	**0.01**
14	92.6	**92.7**	**92.7**	18,619.62	3.51	**0.01**
15	94.1	**94.2**	**94.2**	22,044.14	2.84	**0.01**
16	95.3	**95.4**	**95.4**	22,419.83	4.08	**0.01**
17	96.4	**96.6**	**96.6**	22,016.65	2.54	**0.01**
18	97.5	**97.6**	**97.6**	20,014.90	2.48	**0.01**
19	97.8	**98.3**	**98.3**	19,315.41	2.66	**0.02**
20	98.3	**99.0**	**99.0**	22,310.88	2.71	**0.01**
21	99.0	**99.7**	**99.7**	22,480.92	2.47	**0.01**
22	99.7	**100.0**	**100.0**	22,624.49	2.50	**0.01**
Average	NA*	NA	NA	21,483.52	2.78	**0.01**

*NA means 'Not Applicable'

PySpatialOpt (2.78 s) and Maxcovr (0.01 s). This highlights the computational efficiency of PySpatialOpt and Maxcovr in comparison with ArcGIS.

6 Discussion

The case studies highlight several issues regarding location cover open-source tools. In particular, given a location coverage problem and a solver, there are two conditions that guarantee an optimal solution for a given problem: first, the solver accommodates the distance metric specified in the given problem; second, the solver offers optimal methods. In the above cases, both conditions are satisfied in PySpatialOpt and Maxcovr, so optimal solutions are identified for all instances. Therefore, location models should allow users to specify a distance metric that is well-suited to a given planning problem. PySpatialOpt, Maxcovr and FLP Solver provide such flexibility via accepting a service area configuration or a facility-to-demand distance matrix as input. In contrast, ArcGIS requires a transportation network, which is particularly limiting in some cases. In the York case study, a network had to be created to reflect Euclidean space before the ArcGIS solver could be applied.

It is important to note that we have extended the function of PySpatialOpt and Maxcovr, which led to the satisfaction of the above criteria and improvement of computing efficiency. Specifically, we introduced the distance matrix into PySpatialOpt, which proved more efficient than using service areas. In addition, we incorporated the variable demand weights into Maxcovr, which originally required an equal demand weight. This reflects the importance of the freedom to inspect and extend software functionality, which is only made possible by open source access.

The case studies highlight a comparison of exact and heuristic methods. Both PySpatialOpt and Maxcovr derive optimal solutions using optimisation solvers. In comparison, the FLP Solver relies on a heuristic, which cannot guarantee optimal results. Interestingly, in the case study for locating towers, the ArcGIS heuristic failed to find optimal solutions to the LSCP and 14 MCLP problem instances (out of 22). The suboptimal solutions may be an issue as most users tend to treat results obtained from GIS software as ‘optimal’. When siting facilities that are costly to staff and maintain, any facilities above the minimum lead to unnecessary expenditure. Moreover, PySpatialOpt and Maxcovr perform significantly more efficiently than FLP Solver in solving LSCP and MCLP, mainly due to approach differences (similarly for ArcGIS).

The results also show that PySpatialOpt and Maxcovr are more scalable than FLP Solver and ArcGIS. While PySpatialOpt and Maxcovr scaled well to the York case study, FLP Solver failed to solve this case due to the one-million limit of the distance record. On the other hand, ArcGIS did not scale well to the York case as it required significantly longer computing time than PySpatialOpt or Maxcovr, which could be explained by the ArcGIS workflow that requires a special network reflecting Euclidean distance.

Another discussion point is associated with the documentation. Documentation for software is critical to ensure the code can be used by others, and the cases can be replicated or reproduced (Rey 2018). Documentation is also one of the key criteria

in selecting open source GIS software (Steiniger and Hunter 2013). While documentation in PySpatialOpt and FLP Solver is extensive and clear, documentation is limited for the MCLP in Maxcovr, yet necessary to explain that it requires a non-empty set of existing facilities. Of course, this is true as well for ArcGIS, with no documentation to explain that when an LSCP is not feasible, the model returns a facility configuration that covers only those demand that can be covered.

7 Conclusions

This paper has investigated location cover models (LSCP and MCLP in particular) available through open-source and proprietary tools. The tools reviewed are PySpatialOpt, Maxcovr and FLP Solver, and results are compared with those obtained using ArcGIS. Such comparison is critical since the future will no doubt see an increased use and application of these methods through open-source approaches, similar to how current application relies heavily on commercial GIS platforms like ArcGIS and TransCAD (see Murray et al. 2019; Xu et al. 2020). Overall, PySpatialOpt and Maxcovr outperform FLP Solver and ArcGIS in terms of solution optimality, scalability and distance types. The differences between these tools are noticeable in many aspects. Regarding problem assumptions, the open source options are more flexible than ArcGIS as they allow for a wide range of space and distance metrics. Moreover, PySpatialOpt and Maxcovr use exact methods and rely on mathematical solvers (e.g. GLPK) to solve location cover problems, in contrast to FLP Solver that relies on a heuristic (as does ArcGIS). These differences were demonstrated in two application studies that involve two LSCP and 34 MCLP instances. PySpatialOpt and Maxcovr were found to be relatively fast to solve (significantly faster than ArcGIS). Moreover, all solutions derived by PySpatialOpt and Maxcovr were optimal, which are important in facility planning practice. On the other hand, FLP Solver identified optimal solutions for the San Francisco instances, yet could not solve the York case due to the large problem size (591 demand and 1821 potential sites).

The findings of this study suggest some guide on reproducible location modelling from a software perspective. First, a package should have open code and be scrutable. Second, if possible, a package should rely on exact methods to solve location models. Where heuristics are used, a detailed technical description should be provided. Third, a package should accommodate a wide range of space and distance type. Ideally, a package should accept a distance matrix between demand and potential sites as input.

While open-source software is promising for location cover modelling, further work needs to be done with regards to quality control, maintainability and collaboration with other projects, as suggested in (Steiniger and Hunter 2013). First, software quality control measures such as documentation and testing should be implemented in the open-source location models, which would guarantee code usability. Second, the software maintainability (e.g. ease of modification, clear structure) of these tools should be enhanced, which is important to attract contributions from more developers. Third, as there is a trend of collaboration among open source GIS projects,

PySpatialOpt and Maxcovr can be coupled with other open source GIS libraries such as geopandas (Jordahl et al. 2020) and PySAL (Rey and Anselin 2007, 2010), which would add to the analytical capacity and attract wider users.

Funding No funding was received to assist with the preparation of this manuscript.

Declarations

Conflicts of interest The authors have no relevant financial or non-financial interests to disclose.

References

Adenso-Díaz B, Rodríguez F (1997) A simple search heuristic for the MCLP: application to the location of ambulance bases in a rural region. Omega 25(2):181–187. https://doi.org/10.1016/S0305-0483(96)00058-8

Adesina EA, Odumosu JO, Morenikeji OO, Umoru E, Ayokanmbi AO, Ogunbode EB (2017) Optimization of fire stations services in Minna metropolis using maximum covering location model (MCLM). J Appl Sci Environ Sustain 3(7):172–187

Anselin L, Syabri I, Kho Y (2006) GeoDa: an introduction to spatial data analysis. Geogr Anal 38(1):5–22. https://doi.org/10.1111/j.0016-7363.2005.00671.x

Bollen K, Cacioppo JT, Kaplan RM, Krosnick JA (2015) Social, behavioral, and economic sciences perspectives on robust and reliable science. National Science Foundation, Arlington, VA

Brunsdon C (2015) tbart: Teitz and Bart's p-median algorithm. https://cran.r-project.org/package=tbart. Accessed 13 Jan 2021

Buliung RN, Remmel TK (2008) Open source, spatial analysis, and activity-travel behaviour research: capabilities of the aspace package. J Geogr Syst 10(2):191–216. https://doi.org/10.1007/s10109-008-0063-7

Chen, H. (2021) Open source location cover models. https://github.com/huanfachen/Open_source_location_cover_models. Accessed 11 Mar 2021

Chen H, Murray AT (2021) Open-source approaches for location coverage modelling. In: Mobasheri A (ed) Open source geospatial science for urban studies lecture notes in intelligent transportation and infrastructure. Springer, Cham, pp 117–129

Church RL (2002) Geographical information systems and location science. Comput Oper Res 29(6):541–562. https://doi.org/10.1016/S0305-0548(99)00104-5

Church RL, Murray AT (2009) Business site selection, location analysis and GIS, 1st edn. Wiley, Hoboken

Church RL, Murray A (2018) Location covering models: history, applications and advancements. Springer, New York

Church R, ReVelle C (1974) The maximal covering location problem. Pap Reg Sci Assoc 32(1):101–118. https://doi.org/10.1007/BF01942293

Church RL, ReVelle CS (1976) Theoretical and computational links between the *p*-median, location set-covering, and the maximal covering location problem. Geogr Anal 8(4):406–415. https://doi.org/10.1111/j.1538-4632.1976.tb00547.x

Microsoft Corporation (2018) Excel specifications and limits—Excel. https://support.office.com/en-us/article/excel-specifications-and-limits-1672b34d-7043-467e-8e27-269d656771c3. Accessed 31 Oct 2019

Daskin MS (2002) SITATION facility location software. https://daskin.engin.umich.edu/software/. Accessed 28 Jan 2020

Erdoğan G (2019) FLP spreadsheet solver. https://people.bath.ac.uk/ge277/index.php/flp-spreadsheet-solver/. Accessed 19 Aug 2019

Erdoğan G, Stylianou N, Vasilakis C (2019) An open source decision support system for facility location analysis. Decis Support Syst 125:113–116. https://doi.org/10.1016/j.dss.2019.113116

ESRI (2017) ArcGIS Desktop: Release 10.6

ESRI (2020) Algorithms used by the ArcGIS network analyst extension—ArcMap|documentation. https://desktop.arcgis.com/en/arcmap/latest/extensions/network-analyst/algorithms-used-by-network-analyst.htm#ESRI_SECTION1_6FFC9C48F24746E182082F5DEBDBAA92. Accessed 23 Dec 2020

Gaboardi JD, Folch DC, Horner MW (2020) Connecting points to spatial networks: effects on discrete optimization models. Geogr Anal 52(2):299–322. https://doi.org/10.1111/gean.12211

García-Palomares JC, Gutiérrez J, Latorre M (2012) Optimizing the location of stations in bike-sharing programs: a GIS approach. Appl Geogr 35(1–2):235–246. https://doi.org/10.1016/j.apgeog.2012.07.002

Gearhart JJL, Adair KKL, Detry RJ, Durfee JD, Jones KA, Martin N (2013) Comparison of open-source linear programming solvers. prod-ng.sandia.gov

GNU Project (2017) GLPK (GNU Linear Programming Kit), version 4.65

GNU Project (2019) Categories of free and nonfree software. https://www.gnu.org/philosophy/categories.html. Accessed 29 Jan 2020

GRASS Development Team (2017) Geographic Resources Analysis Support System (GRASS GIS) Software, Version 7.2

Jankowski P, Brown B (2014) Health care accessibility modeling: effects of change in spatial representation of demand for primary health care services. Quaest Geogr 33(3):39–53. https://doi.org/10.2478/quageo-2013-0028

Jordahl K, Bossche J, Van den Fleischmann M, et al. (2020) geopandas/geopandas: v0.8.0

Kedron P, Frazier AE, Trgovac AB, Nelson T, Fotheringham AS (2019) Reproducibility and replicability in geographical analysis. Geogr Anal. https://doi.org/10.1111/gean.12221

Khan MMUH, Vaezi M, Kumar A (2018) Optimal siting of solid waste-to-value-added facilities through a GIS-based assessment. Sci Total Environ 610–611:1065–1075. https://doi.org/10.1016/j.scitotenv.2017.08.169

Leeka JT, Peng RD (2015) Opinion: reproducible research can still be wrong: adopting a prevention approach. Proc Natl Acad Sci USA 112(6):1645–1646. https://doi.org/10.1073/pnas.1421412111

Meindl B, Templ M (2012) Analysis of commercial and free and open source solvers for linear optimization problems. Citeseer, Vienna

Munoz-Marquez, M. (2019) orloca: operations research LOCational analysis models. https://cran.r-project.org/package=orloca. Accessed 12 Jan 2021

Murray AT (2005) Geography in coverage modeling: exploiting spatial structure to address complementary partial service of areas. Ann Assoc Am Geogr 95(4):761–772. https://doi.org/10.1111/j.1467-8306.2005.00485.x

Murray AT (2010) Advances in location modeling: GIS linkages and contributions. J Geogr Syst 12(3):335–354. https://doi.org/10.1007/s10109-009-0105-9

Murray AT, Wei R (2013) A computational approach for eliminating error in the solution of the location set covering problem. Eur J Oper Res 224(1):52–64. https://doi.org/10.1016/j.ejor.2012.07.027

Murray AT, Xu J, Wang Z, Church RL (2019) Commercial GIS location analytics: capabilities and performance. Int J Geogr Inf Sci 33(5):1106–1130. https://doi.org/10.1080/13658816.2019.1572898

Odell E (2019) {Ukpolice}: download data on UK police and crime. https://github.com/evanodell/ukpolice. Accessed 12 Jan 2021

Oviasu OI (2014) Determining the location of suitable satellite centres for Chronic Kidney Disease (CKD) treatment within Edo State, Nigeria. GeoJournal 79(5):527–538. https://doi.org/10.1007/s10708-013-9511-0

Polo G, Acosta CM, Ferreira F, Dias RA (2015) Location-allocation and accessibility models for improving the spatial planning of public health services. PLoS One 10(3):1–14. https://doi.org/10.1371/journal.pone.0119190

Pulver A (2019) PySpatialOpt: an open-source spatial optimization library. https://github.com/apulverizer/pyspatialopt. Accessed 12 Jan 2021

QGIS Development Team (2009) QGIS Geographic Information System

Rey SJ (2009) Show me the code: spatial analysis and open source. J Geogr Syst 11(2):191–207. https://doi.org/10.1007/s10109-009-0086-8

Rey SJ (2018) Code as text: open source lessons for geospatial research and education. In: Thill J-C, Dragicevic S (eds) GeoComputational analysis and modeling of regional systems. Springer, Cham, pp 7–21

Rey SJ, Anselin L (2007) PySAL: a Python library of spatial analytical methods. Rev Reg Stud 37(1):5–27

Rey SJ, Anselin L (2010) PySAL: a Python library of spatial analytical methods. In: Fischer MM, Getis A (eds) Handbook of applied spatial analysis: software tools, methods and applications. Springer, Berlin, pp 175–193

Singleton AD, Spielman S, Brunsdon C (2016) Establishing a framework for open geographic information science. Int J Geogr Inf Sci 30(8):1507–1521. https://doi.org/10.1080/13658816.2015.1137579

Sinnott RW (1984) Virtues of the Haversine. Sky Telesc 68:159

Steiniger S, Hunter AJS (2013) The 2012 free and open source GIS software map: a guide to facilitate research, development, and adoption. Comput Environ Urban Syst 39:136–150. https://doi.org/10.1016/J.COMPENVURBSYS.2012.10.003

Tierney, N. (2019) Maxcovr: A set of tools for solving the maximal covering location problem. https://github.com/njtierney/maxcovr. Accessed 12 Jan 2021

Tong D, Murray AT (2009) Maximising coverage of spatial demand for service. Pap Reg Sci 88(1):85–97. https://doi.org/10.1111/j.1435-5957.2008.00168.x

Tong D, Murray AT (2012) Spatial optimization in geography. Ann Assoc Am Geogr 102(6):1290–1309. https://doi.org/10.1080/00045608.2012.685044

Tong D, Wei R (2017) Regional coverage maximization: alternative geographical space abstraction and modeling. Geogr Anal 49(2):125–142. https://doi.org/10.1111/gean.12121

Tong D, Murray A, Xiao N (2009) Heuristics in spatial analysis: a genetic algorithm for coverage maximization. Ann Assoc Am Geogr 99(4):698–711. https://doi.org/10.1080/00045600903120594

Toregas C, Swain R, ReVelle C, Bergman L (1971) The location of emergency service facilities. Oper Res 19(6):1363–1373. https://doi.org/10.1287/opre.19.6.1363

White AN (1979) Accessibility and public facility location. Econ Geogr 55(1):18. https://doi.org/10.2307/142730

Woodhouse S, Lovett A, Dolman P, Fuller R (2000) Using a GIS to select priority areas for conservation. Comput Environ Urban Syst 24(2):79–93. https://doi.org/10.1016/S0198-9715(99)00046-0

Xu J, Murray A, Wang Z, Church R (2020) Challenges in applying capacitated covering models. Trans GIS 24(2):268–290. https://doi.org/10.1111/tgis.12608