Front. Environ. Sci. Eng. 2020, 14(3): 44
https://doi.org/10.1007/s11783-020-1221-5

RESEARCH ARTICLE

PyLUR: Efficient software for land use regression modeling the spatial distribution of air pollutants using GDAL/OGR library in Python

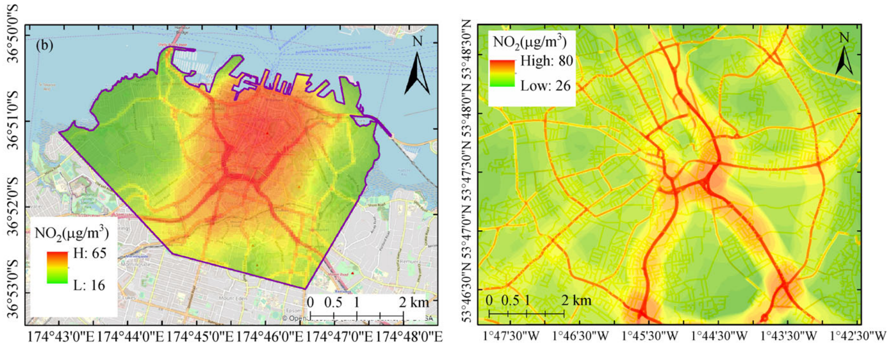
Xuying Ma (✉)[1,2], Ian Longley[2], Jennifer Salmond[1], Jay Gao[1]

1 School of Environment, Faculty of Science, University of Auckland, Auckland 1010, New Zealand
2 National Institute of Water and Atmospheric Research, Auckland 1010, New Zealand

HIGHLIGHTS

- PyLUR comprises four modules for developing and applying a LUR model.
- It considers both conventional and novel potential predictor variables.
- GDAL/OGR libraries are used to do spatial analysis in the modeling and prediction.
- Developed on Python platform, PyLUR is rather efficient in data processing.

GRAPHIC ABSTRACT



ARTICLE INFO

Article history:
Received 24 June 2019
Revised 24 December 2019
Accepted 13 January 2020
Available online 9 March 2020

Keywords:
LUR
Air pollution modelling
GIS spatial analysis
GDAL/OGR Python
Pollutant concentration mapping

ABSTRACT

Land use regression (LUR) models have been widely used in air pollution modeling. This regression-based approach estimates the ambient pollutant concentrations at un-sampled points of interest by considering the relationship between ambient concentrations and several predictor variables selected from the surrounding environment. Although conceptually quite simple, its successful implementation requires detailed knowledge of the area, expertise in GIS, statistics, and programming skills, which makes this modeling approach relatively inaccessible to novice users. In this contribution, we present a LUR modeling and pollution-mapping software named PyLUR. It uses GDAL/OGR libraries based on the Python platform and can build a LUR model and generate pollutant concentration maps efficiently. This self-developed software comprises four modules: a potential predictor variable generation module, a regression modeling module, a model validation module, and a prediction and mapping module. The performance of the newly developed PyLUR is compared to an existing LUR modeling software called RLUR (with similar functions implemented on R language platform) in terms of model accuracy, processing efficiency and software stability. The results show that PyLUR out-performs RLUR for modeling in the Bradford and Auckland case studies examined. Furthermore, PyLUR is much more efficient in data processing and it has a capability to handle detailed GIS input data.

1 Introduction

Due to rapid population growth and urban expansion air pollution has recently become a popular and important research topic (Li et al., 2019; Liu et al., 2017; Zou et al., 2016). Air pollution is usually monitored at a discrete number of sites using one or more different kinds of measuring devices. Despite advances in technology, which have made it possible to develop dense networks of sensors, these measurements often do not adequately capture the complex and heterogeneous spatial patterns of air pollutants realistically. Further, data- even from high-density networks of observations- usually need to be averaged in time and space to generate useable maps of air quality for management or epidemiological purposes. The challenge of developing reasonable and reliable models to estimate the spatial distribution of air pollutants remains a

✉ Corresponding author
E-mail: xma295@aucklanduni.ac.nz

significant task in air quality management, and is crucial to reducing pollution-monitoring costs and improving the effectiveness of monitoring networks (Keller et al., 2015).

Land use regression (LUR) models are a well-established approach for simulating spatial patterns in air quality. LUR models provide a regression-based approach to estimate the ambient pollutant concentrations at un-sampled points of interest using different kinds of predictor variables generated by ‘near’ and ‘buffer’ analyses in a Geographic Information System (GIS) within a specific location or area (Briggs et al., 1997). After the LUR model is developed and validated, it can be used to predict the pollutant concentration at each point of interest accurately, but only within the study area (the developed LUR model only has limited transferability due to its nature as a statistical model), and with the assumption that all required GIS data are available. For pollutant concentration mapping and prediction, the LUR model can be used to generate a general pollutant concentration map for the whole study area (using fishnet to create each prediction point) or provide precise concentration predictions at specific locations.

So far, LUR models have been widely presented in numerous publications (Meng et al., 2015; Zhai et al., 2016; Masiol et al., 2018; Muttoo et al.,2018; Weissert et al., 2018; Saucy et al., 2018; Ma et al., 2019; Xu et al.,2019a; Xu et al.,2019b). The principles of LUR modeling can be summarized in five steps. First, air pollution monitoring and GIS data are collected within the scope of the study area. Secondly, different kinds of potential predictor variables for each site are generated using GIS buffering or other geospatial analysis methods. Thirdly, multiple regression analysis is carried out to develop one regression equation establishing the relationship between the observed air pollutant concentrations and significant predictor variables selected from a pool of all potential predictor variables. Fourthly, model performance is evaluated using holdout or cross-validation. Finally, once the model is successfully validated, it can be applied to predict the concentration at un-sampled points of interest or generate an air pollutant concentration map of the whole study area (Morley and Gulliver, 2018).

LUR modeling is not very complex theoretically and can be implemented using GIS to manually extract predictor variables used in modeling and prediction parts using commercial GIS software such as ArcGIS or QGIS, accompanied by a statistical tool such as R or Stata (for regression modeling and validation). However, the manual processing is inefficient, time-consuming, and error-prone, especially in high-resolution concentration mapping where many manual GIS operations are required. Due to this intensive computational requirement and the limitation of a manual modeling strategy, many previous LUR studies did not provide results in the format of a concentration map or only provide maps at coarse spatial resolutions (Liu et al., 2015; Marcon et al., 2015; Miskell et al., 2015; Wu et al., 2015; Miskell et al., 2018; Weissert et al., 2018). Developing a LUR model but without providing a high-resolution spatial distribution map of the air pollutant reduces the potential of the work to be applied to further studies or applications.

To build and apply a LUR model efficiently it may be advantageous to develop a purpose-specific software or tool kit to carry out the complex spatial analyses and calculations automatically. However, there are very few publicly available documents describing the process of LUR modeling software at present. Akita (2014a; 2016b) developed an ArcGIS extension Toolbox named ‘LUR Tools’ that could be used to extract potential predictor variables. However, this tool kit could only generate limited kinds of potential predictor variables; regression modeling, validation and prediction were not considered. Morley and Gulliver (2018) developed a piece of software named as ‘RLUR’ for LUR modeling and prediction based on the R language platform. It is capable of running a LUR model from predictor variable generation, through regression modeling, to final prediction. RLUR is open-source software with a user-friendly, GUI-based operation environment. An advantage of RLUR is its ability to allow users inexperienced in statistic and spatial analysis to run LUR modeling in their studies. However, major shortcomings of RLUR are its processing inefficiency and instability issues. The predictor variable extraction processing will become very slow if detailed GIS inputs were uploaded and the program would crash each time if the predicted air pollutant concentrations were mapped at a high spatial resolution (e.g., a 25 m resolution for a study area of 10 km^2 using RLUR test data sets).

This problem is overcome in this study by developing a more efficient and stable software based on Python. The objective of this paper is to describe this self-developed software named PyLUR and its implementation of LUR modeling on the Python platform. Taking advantage of the GDAL/OGR libraries (Python binding) and Python platform, PyLUR can develop a LUR model and generate pollutant concentration maps efficiently for any given study area if suitable air pollution and GIS data are available. Also described in this paper are its four constituent modules. The performance of the newly developed PyLUR is then compared to RLUR and assessed in terms of model accuracy, processing efficiency and software stability using two different test data sets.

This paper is divided into 5 sections and organized as follows. Section 2 gives a brief overview of the methodology of LUR modeling. Section 3 introduces the structure and core modules of PyLUR and shows how it works. Section 4 compares the performance of PyLUR with RLUR in terms of model accuracy and efficiency of data processing. Finally, Section 5 presents a summary and conclusion of this study.

2 Spatial modeling of air pollutants

A LUR model uses a statistical optimization approach to estimate the spatial distribution of air pollutant concentrations in a designated study area. A linear regression equation is formulated following some certain criteria. It relates air pollutant concentration at each monitoring site with selected predictor variables representing sources of air pollutants and modifiers in the vicinity of the monitored site.

2.1 Extraction of potential predictor variables

The regression equation of a LUR model describes the relationship between the air pollutant concentration at monitoring sites and their predictor variables representing pollution sources and modifiers of the surrounding environment. These variables considered in an LUR modeling process- reflecting the influence of surrounding features of emission, modifier and sink on air pollutant concentration of a monitoring site- are called potential predictor variables. The potential predictor variables included in the final LUR model are called predictor variables. Potential predictor variables are derived from GIS-based buffering analysis and near analysis. Most potential predictor variables considered in PyLUR follow the manual of the ESCAPE (European Study of Cohorts for Air Pollution Effects) project (ESCAPE, 2010). In addition, several novel variables are also included in this software.

2.2 Regression modeling

Most previous LUR models are largely based on the multiple linear regression technique (Meng et al., 2015; Miskell et al., 2015; Masiol et al., 2018; Miskell et al., 2018; Muttoo et al.,2018; Saucy et al., 2018; Weissert et al., 2018). Multiple linear regression is a statistical method to establish the relationship between the dependent variable and several independent variables. In general, a multiple linear regression model can be expressed as:

$$Y = a_0 + a_1X_1 + a_2X_2 + a_3X_3 + \cdots + a_nX_n \quad (1)$$

In which $X_1, X_2, \cdots, X_n$ are n samples of independent variables; Y is the dependent variable; $a_0, a_1, a_2, \cdots, a_n$ are coefficients of the independent variables. In the case of a LUR model based on multiple linear regression, an equation will be built to establish the relationship between the measured concentrations of the pollutant at monitoring sites (dependent variable) and predictor variables (independent variables) generated from their surrounding geographic features.

In this study, the standard supervised forward stepwise method is applied to establish a LUR model (ESCAPE, 2010; Beelen et al., 2013). The technique can be summarized in five steps. (a) n (assume there are n kinds of potential predictor variables) individual regression equations are built to regress air pollution measurements against a single kind of potential predictor variable based on univariate regression (finally this step will generate n regression equations). The regression equation from these n options with the highest R^2 is determined as the ‘potential start LUR model’. This ‘potential start LUR model’ would be confirmed as a ‘start model’ if the direction of effect of the predictor variable in this model is the same with its pre-defined criteria. (b) Each time only one potential predictor variable from the remaining pool is included to the ‘former model’ built from steps (a) or (b–c) and the R^2 of the ‘new model’ is stored. An iteration is used to loop step (b) k (k indicates the number of potential predictor variables in the remaining pool at the beginning of step (b) times and the R^2 of each ‘new model’ is stored respectively. (c) Comparing all ‘new models’ generated from step b, the new model with the highest additional increase in R^2 is determined as the ‘final new model’ if 1) the additional increase of R^2 is $>1\%$; 2) the direction of effect of each predictor variable in the ‘new model’ is still the same with its pre-defined criteria. (d) Iterate steps b–c until the additional increase of R^2 is $< 1\%$ if all the remaining potential predictor variables are tried to update a ‘new model’. (e) The predictor variables in the ‘current model’ with p-value >0.10 are excluded from the model equation consecutively, beginning from the variable with the maximum p-value, until all variables in the final model are statistically significant. After forming the final LUR model based on the above steps, there is still a need to check if the LUR equation meets criteria below. Influential observations are checked by Cook’s Distance (normally $<$ 1.0). Collinearity is checked by the variance inflation factor (VIF, any predictor variable with a VIF >3.0 is not acceptable). Moran’s I is used to check if there is any spatial autocorrelation (normally $p > 0.05$). Both heteroscedasticity and normality of the residuals are checked visually by plotting them out.

2.3 Model validation

After regression modeling, the developed LUR model still needs to be assessed the extent to which it can simulate the pollutant concentrations accurately. The performance of the built LUR model is assessed using R^2 and root mean square error (RMSE) generated from a comparison between the predicted and measured concentrations.

In this study, three kinds of validation methods are considered: Holdout Validation (HV), K-fold Cross-Validation (KCV), and Leave-One-Out Cross-Validation (LOOCV). Each of these methods can be applied to validate the developed LUR model according to different measurements and users’ needs. In HV, all samples (monitoring sites) are divided into two groups randomly; one data set (training data set) is used to develop the LUR model, and the other data set (testing data set) is used to validate the developed model, respectively (Kim, 2009). Usually, the training data set is much larger than the testing data set. In KCV, all samples are divided into *K* equal sized groups. Among these groups, each time only one group of samples is selected as the testing data set, and all the remaining samples from other *K*–1 groups are used to develop the LUR model. Looping the above-mentioned process *k* times, every group will be used just once to validate the developed LUR model (Kohavi, 1995). In LOOCV, assuming from the data set there are *n* samples; leave one sample to test the LUR model developed based on a training data set of the remaining *n*–1 samples at each time. Iterating this step *n* times, the developed LUR model is validated by comparing measurements with the prediction at each site generated from this *n* iterations sequentially (Hoek et al., 2008).

2.4 Prediction and concentration mapping

If the developed LUR model passes the validation process successfully, the final LUR model can be used to predict the pollutant concentration at any pre-determined points of interest within the study area or generate a distribution map of the pollutant concentration for the whole study area at a user-specified spatial resolution. The process of prediction using the validated LUR model at unsampled points can be summarized as: (1) a layer showing unsampled points of interest or regular grids covering an area is made; (2) at any unsampled point, values of predictor variables shown in the final LUR model can be extracted and then put into the LUR regression equation to estimate the concentration at this point; (3) step 2 can be applied to each unsampled point. For the concentration mapping, the resolution of the map can be set by changing the spacing of output grids. If predicted grids of the fishnet are not dense enough (e.g., a coarse resolution map predicted using LUR model), geostatistical interpolation techniques (such as inverse distance weight and ordinary Kriging) can be used in ArcGIS to interpolate the map into a higher resolution one. However, the quality and accuracy of the high-resolution map generated in this indirect way is not as good as that of a high-resolution map directly generated by the LUR model (which requires a large number of spatial computations).

3 PyLUR software

The PyLUR software described here, which realizes all the functions described in Section 2, is developed based on Python platform (Sanner, 1999). It contains four self-developed modules- a potential predictor variable generation module, a regression modeling module, a model validation module, and a prediction and mapping module- represented in different colors in Fig. 1. Also illustrated in the figure is a flowchart describing the precedures of LUR modeling introduced in Section 2. PyLUR could be used for both long- and short- term air quality modeling and concentration mapping. The temporal-resolution of the LUR model and its corresponding concentration map generated by PyLUR is decided by the temporal-resolution of the input monitoring data (If the monitoring data provided is an annual/monthly/daily/hourly average based data set, then the developed LUR model and its corresponding concentration map are also annual/monthly/daily/hourly average based results). To explain how PyLUR is developed and works, a set of monitored nitrogen dioxide (NO_2) and GIS data for Auckland (New Zealand) were used with PyLUR (named as Auckland test data sets) to test the software. This monitoring data set contains annual averaged (2017) NO_2 concentration at 42 monitoring sites. The distribution of these sites is shown in Fig. 2(a).

3.1 Potential predictor variable generation module

The main purposes of the potential predictor variable generation module are to first upload the monitoring data set and all relevant GIS layers, and then generate corresponding potential predictor variables at each monitoring site. To upload the input data, a simple GUI of pop-up windows is developed using Python tkinter module to instruct users to upload the monitoring data and various GIS data. GDAL/OGR module is imported and used to read GIS data and implement spatial analysis. GDAL (Geospatial Data Abstraction Library) is an open source library designed to manipulate raster data, such as Digital Elevation Model (DEM). OGR (OpenGIS Simple Features Reference Implementation) is another open source library designed to manipulate vector data. These two libraries are written in C/C ++ and binding to Python (OSGF, 2008; Westra, 2013).

As a single kind of GIS input data could be represented in numerous data formats, data preparation to conform to a fixed format is required before running the program. The file of the monitoring data should be a point shapefile showing the observation and location of each site (If the raw monitoring data is stored in the csv format, it can first be imported in ArcGIS or QGIS in the data preparation step and then readily converted to a point shapefile). GIS data that can be uploaded into PyLUR includes: land use maps, road network (including road types and traffic volume), population and household information of each census unit, footprint of each building with height information, sky view factor (SVF) map (involving two files: the first is a point shapefile converted from the original raster map during data preparation and the second

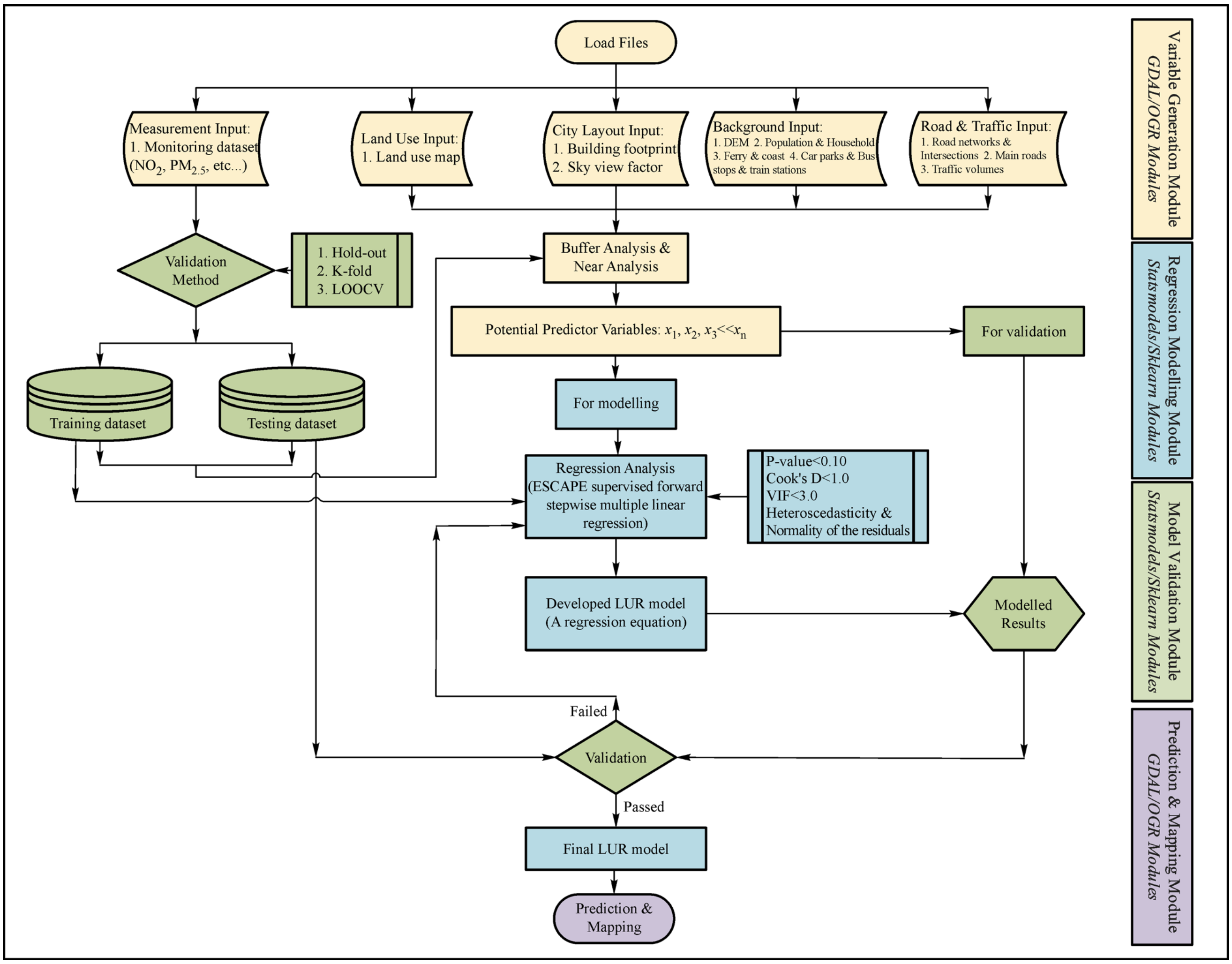

Load Files
Measurement Input:
1. Monitoring dataset
(NO_2, $PM_{2.5}$, etc...)
Land Use Input:
1. Land use map
City Layout Input:
1. Building footprint
2. Sky view factor
Background Input:
1. DEM 2. Population & Household
3. Ferry & coast 4. Car parks & Bus stops & train stations
Road & Traffic Input:
1. Road networks & Intersections 2. Main roads
3. Traffic volumes
Validation Method
1. Hold-out
2. K-fold
3. LOOCV
Buffer Analysis & Near Analysis
Potential Predictor Variables: x_1, x_2, x_3<<x_n
For validation
Training dataset
Testing dataset
For modelling
Regression Analysis (ESCAPE supervised forward stepwise multiple linear regression)
P-value<0.10
Cook's D<1.0
VIF<3.0
Heteroscedasticity & Normality of the residuals
Developed LUR model (A regression equation)
Modelled Results
Failed
Validation
Passed
Final LUR model
Prediction & Mapping
Variable Generation Module
GDAL/OGR Modules
Regression Modelling Module
Statsmodels/Sklearn Modules
Model Validation Module
Statsmodels/Sklearn Modules
Prediction & Mapping Module
GDAL/OGR Modules

Fig. 1 The flowchart of the LUR modeling and module structures of the PyLUR software.

is a field added in the monitoring shapefile using value extraction from the original raster map), DEM (should extract value to the monitoring shapefile during data preparation), location maps of car parks, ferries, coastline, train stations and bus stops. The format of most input monitoring and GIS data adheres to the criteria introduced in the ESCAPE project manual (ESCAPE, 2010) and Morley and Gulliver (2018). Figure 2 shows maps of the Auckland case-study monitoring data and parts of its corresponding GIS data sets. Since many novel GIS data and corresponding potential predictor variables can be considered in PyLUR, there are also several other self-defined data formats. For instance, the GIS data of building footprint information is a polygon shapefile: a vector file for each building's footprint with a field ('Height') specifying the height of the building (unit: m). Coordinate system (spatial reference) is a very important issue when handling spatial data. For simplicity, users can use any coordinate system for these monitoring and GIS input data as long as all of them are in the same coordinate system (If not, coordinate system can be converted to the same one using ArcGIS or QGIS in the data preparation step).

After uploading of all the input data, spatial analysis can be implemented to generate potential predictor variables. Table 1 shows potential predictor variables being applicable for PyLUR. All of these variables can be classified into four categories according to their nature: land use variables, urban configuration variables, background variables, and road network and traffic variables. All these variables may influence emissions characteristics, and/or the dispersion and removal of air pollutants. For instance, land use variables reflect the influence of surrounding land use features on the pollutant concentration at a monitoring site. Land use categories such as industry or port are assumed to increase the ambient air pollutant concentration (through increased emissions); while urban green and natural features are assumed to reduce the ambient air pollutant concentration (through decreased emissions and/or increased dispersion). Buffering analysis (circular buffers are used for all buffering

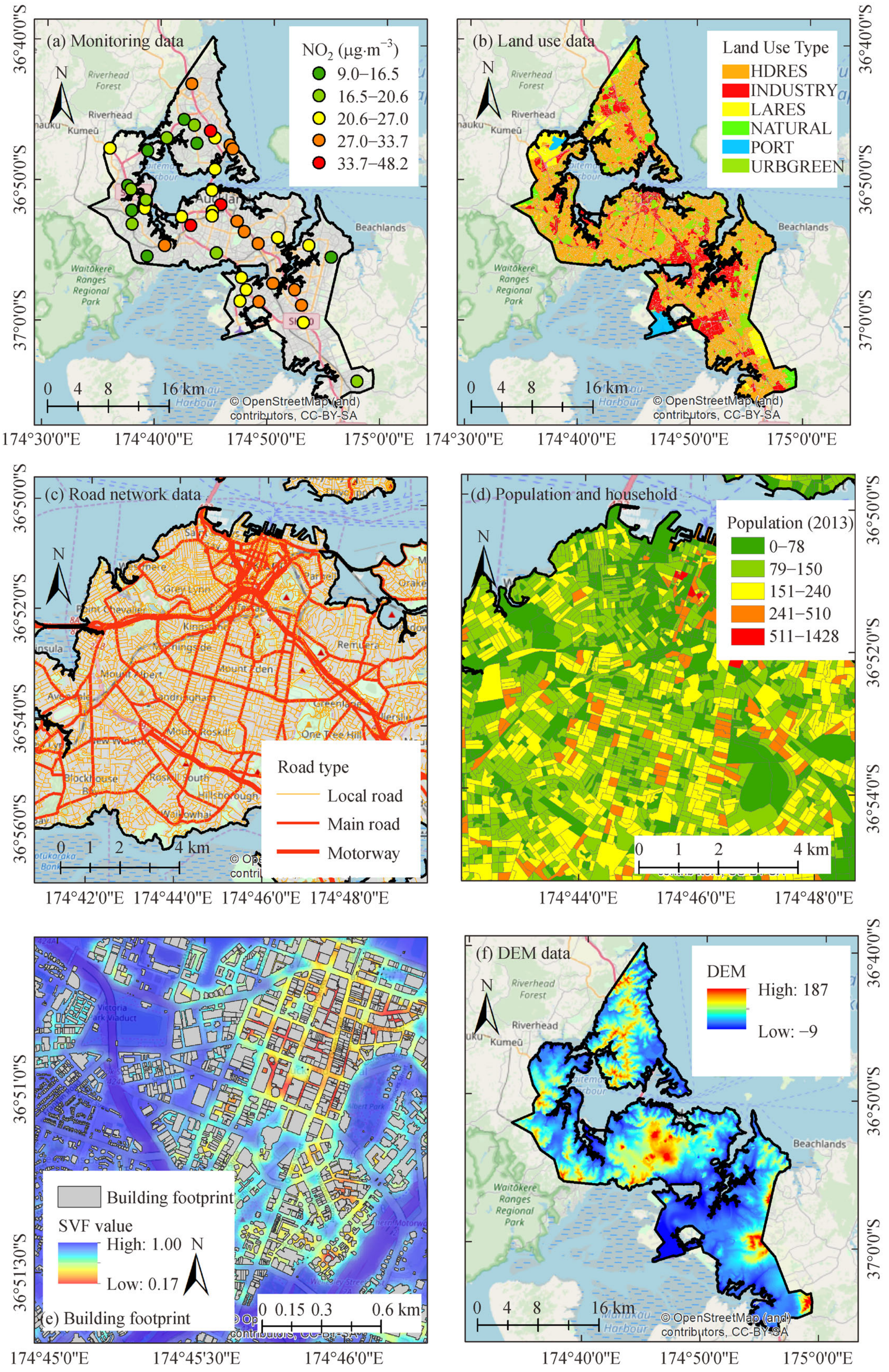


Fig. 2 Maps of monitoring and main input GIS data sets to test PyLUR.

Table 1 Potential predictor variables considered in PyLUR

Potential predictor variable	Variable code
Land use:	
Buffer radii (m): 5000, 2000, 1000, 500, 200, 100	
High density residential zone	HDRES (+)
Low density residential zone	LDRES (+)
Industry	INDUSTRY (+)
Port	PORT (+)
Urban green	URBGREEN (-)
Natural features	NATURAL (-)
Urban configuration:	
Buffer radii (m): 1000, 500, 350, 200, 100, 50	
Sum of building footprint areas	SOBFA (+)
Number of buildings	BUILDNUM (+)
Sum of all building height	SOBH (+)
Average building height	Ratio_BH_BN (+)
Total building volumes/area of buffer	BuildingDensity (+)
Sky view factor	SkyViewFact (-) *
Mean SVF within the buffer circle	MeanSVF (-)
Background:	
Buffer radii (m): 5000, 2000, 1000, 500, 200	
Altitude of the place	Elevation (-) *
Number of inhabitants	POP (+)
Number of households	HHOLD (+)
Number of carparks[1]	CARPD (+)
Inverse of distance to ferry	DISTINVFERRY (+) *
Inverse of distance to coast	DISTINVCOAST (-) *
Number of bus stops[2]	BusStopNums (+)
Inverse of distance to the nearest bus stop	DISTINVNrBusStop (+) *
Inverse of distance to the nearest train station	DISTINVNrTrainStn (+) *
Road network and traffic:	
Buffer radii (m): 1000, 500, 300, 200, 100, 50, 25	
Number of road intersections	RdInterNum (+)
Inverse of distance to the nearest road intersection	DISTINVNrRdIntSe (+) *
Length of all main roads within the buffer circle	MAINROADLENGTH (+)
Inverse of distance to the nearest main road	DISTINVNrMainRoad (+) *
Total traffic load of all major roads	TRAFMAJORLOAD (+)
Total heavy-duty traffic load of all major roads	HTRAFMAJORLOAD (+)
Length of all major roads within the buffer circle	MAJROADLENGTH (+)
Total heavy-duty traffic load of all roads	HEAVYTRAFLOAD (+)
Length of all roads within the buffer circle	ROADLENGTH (+)
Total traffic load of all roads within the buffer circle	TRAFLOAD (+)
Heavy-duty traffic volume on the nearest major road	HEAVYTRAFMAJOR (+) *
Inverse of distance to the nearest major road	DISTINVMAJOR1 (+) *
Inverse of distance squared to the nearest major road	DISTINVMAJOR2 (+) *
Product of TRAFMAJOR and DISTINVNEAR1	INTMAJORINVDIST (+) *

(*Continued*)

Potential predictor variable	Variable code
Product of TRAFMAJOR and DISTINVNEAR2	INTMAJORINVDIST2 (+) *
Traffic volume on the nearest major road	TRAFMAJOR (+) *
Traffic volume on the nearest road	TRAFNEAR (+) *
Inverse of distance to the nearest road	DISTINVNEAR1 (+) *
Inverse of distance squared to the nearest road	DISTINVNEAR2 (+) *
Product of TRAFNEAR and DISTINVNEAR1	INTINVDIST (+) *
Product of TRAFNEAR and DISTINVNEAR2	INTINVDIST2 (+) *
Heavy-duty traffic volume on the nearest road	HEAVYTRAFNEAR (+) *

Notes: (+), (-) after the code name indicates the direction of effect of the predictor variable;
[1] use a buffer radii (m): 1000, 500, 200, 100; [2] use a buffer radii (m): 300, 200, 150, 100, 50;
* indicates: buffer analysis is not applied.

Table 2 Summary statistics of the final LUR model and validations using the Auckland test data.

LUR Model	Coefficient	Std.error	p-Value	VIF
Intercept	8.827994	1.9515	< 0.001	–
MAINROADLENGTH_100	1.264385e–02	0.0024	< 0.001	1.054
Ratio_BH_BN_50	1.343607	0.2778	< 0.001	1.1664
TRAFLOAD_1000	7.642508e–09	< 0.001	0.0421	1.2240
Validation method	R^2	RMSE ($\mu g/m^3$)		
None	0.68	4.35		
K-fold (5-fold)	0.63	4.40		
LOOCV	0.62	4.51		

analysis in PyLUR) at different buffer radii is carried out to calculate the total area of a specific land use type around each monitoring site.

Functions of buffer generation, distance between features and layer intersection are used to write buffer and near analysis tools in Python to extract potential predictor variables. After running the potential predictor variable generation module, a table in an Excel file (.xlsx) including values of all potential predictor variables at each monitoring site is created.

3.2 Regression modeling module

The main purposes of the regression modeling module are to optimise a regression equation to explain maximally the variability of measured air pollution data and at the same time to ensure that all the requirements discussed in Section. 2.2 are met. Two Python packages of Statsmodels and Sklearn are imported and used to form and check the regression equation. Statsmodels is a package in Python designed to implement statistical modeling, analysis and tests (Seabold and Perktold, 2010). Sklearn is another package in Python that contains various functions and tools initially designed for machine learning (Pedregosa et al., 2011). The regression and model selection tools in Statsmodels are used in the software development. As more than 100 potential predictor variables are generated in Section 3.1, massive loops and comparisons would be involved to realize the model selection algorithm described in Section 2.2. Loop structures with too many iterations in a program are time-consuming and can slow down the speed of data processing. To speed up the model selection, correlation analysis is first implemented to exclude insignificant potential predictor variables (the absolute value of the correlation coefficient between one kind of potential predictor variable and measurements is smaller than 0.20).

After running the regression modeling module, an Excel file (.xlsx) including selected predictor variables, corresponding coefficients and model check statistics of the final model will be output. Table 2 shows the result of an LUR model developed based on the given Auckland test data set. In addition, plots of regression diagnostics are also generated to allow the user to confirm the correctness of the proposed LUR model. Figure 3 shows the corresponding diagnostics plots of the LUR model presented in Table 2.

3.3 Model validation module

The main purpose of the model validation module is to evaluate the effectiveness of the developed LUR model. Statsmodels and Sklearn packages are once again

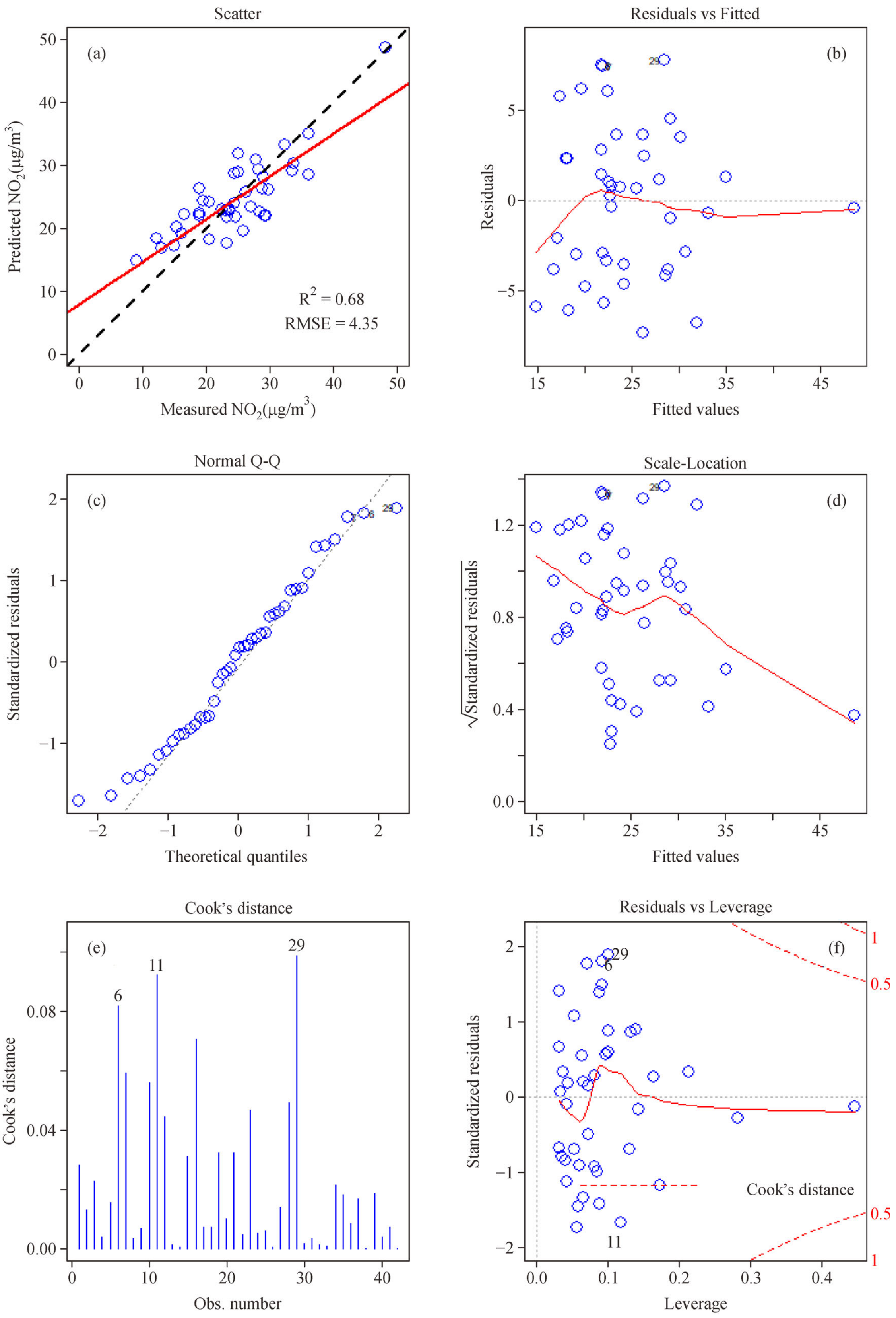


Fig. 3 Diagnostics plots for the final LUR model developed using the regression modeling module of PyLUR. (a). Scatter plot; (b). Residuals vs. fitted plot; (c). Q-Q plot; (d). Scale-location plot; (e). Cook's distance; (f). Leverage plot.

incorporated into this module. Single or multiple options of validation methods can be selected from HV, KCV and LOOCV via a pop-up window shortly after uploading all monitoring and GIS data sets. After running the model validation module, R^2 and RMSE of the corresponding validation methods will be output. Table 2 also shows validation results using different validation methods. “None” in Table 2 means that all sites are used to develop the LUR model that is subsequently validated by the data already used in the model building process instead of using a cross-validation method.

3.4 Prediction and mapping module

The main purposes of the prediction and mapping module are to predict the concentration at the points of interest and generate a concentration map for a certain area. Functions and tools used in this module are largely the same ones as those already developed in the potential predictor variable generation module. Because the procedures of a prediction are to first generate predictor variables involved in the final LUR model and then put these values into the LUR equation to estimate the concentration of each point of interest. The input of this module is a vector shapefile of points of interest or grid points for an area. The output is an Excel file (.xlsx) including the siteID with its corresponding predicted concentration of each point. After joining the output Excel file with the input shapefile in ArcGIS, the concentration map can be either displayed on screen or exported for other uses. Figure 4(a) shows fishnet grids of 50m-resolution for central Auckland and the centroid of each grid is extracted into a point shapefile as the input of this module. Figure 4(b) shows the corresponding NO_2 concentration map of a 50 m resolution for central Auckland output after running this module.

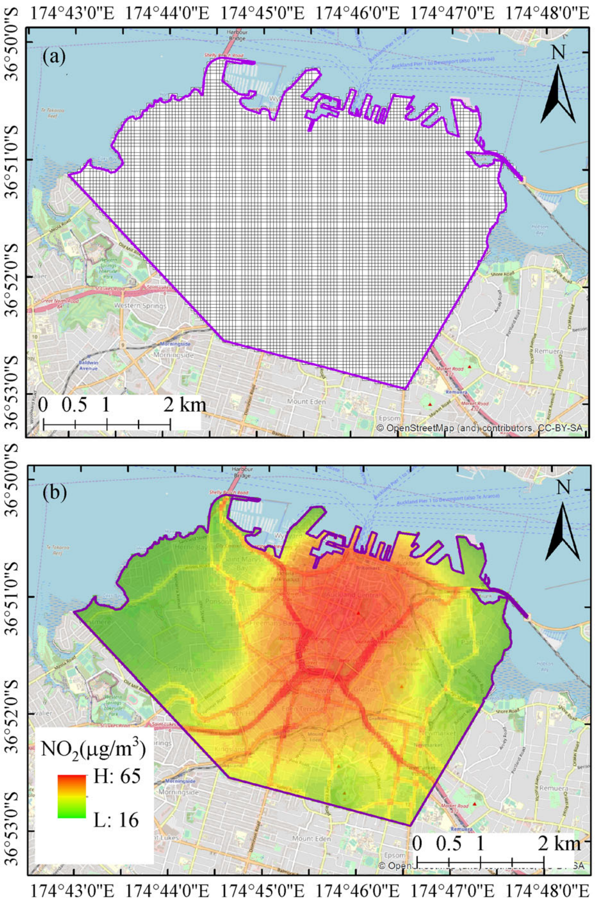

174°43'0"E 174°44'0"E 174°45'0"E 174°46'0"E 174°47'0"E 174°48'0"E
36°50'0"S
36°51'0"S
36°52'0"S
36°53'0"S
(a)
N
0 0.5 1 2 km
© OpenStreetMap (and) contributors, CC-BY-SA
(b)
N
NO_2(μg/m^3)
H: 65
L: 16
0 0.5 1 2 km

Fig. 4 Input of 50m-resolution fishnet grids of central Auckland (a) uploaded and its corresponding output concentration map (b) generated by the prediction and mapping module in PyLUR.

4 Comparison between PyLUR and RLUR

Although a large number of studies have been carried out about LUR modeling, there are only a handful documents about LUR modeling software. To our knowledge, RLUR (Morley and Gulliver, 2018) might be the only publicly accessible and open-source LUR modeling software. To assess the correctness and computational efficiency of PyLUR, a comparison between PyLUR and RLUR is carried out. The test data used for this comparison is the same as that provided by RLUR. This data set contains NO_2 measurements (annual averaged) at 48 sites in Bradford, UK (named as Bradford test data set). Details about the monitoring campaign and corresponding GIS data of this test data set can be found via the website of www.sciencedirect.com/science/article/pii/S1364815217-302621. The testing platform is a HP desktop running Windows 10 Enterprise, with a 2.50 GHz i5-6500T processor and 8GB of DDR3 RAM (random-access memory). In the modeling process, around 90 potential predictor variables are considered in both PyLUR and RLUR (because RLUR just supports basic potential predictor variables and its test data does not include some background and urban configuration GIS data, PyLUR can also be run if these additional GIS data sets are not available).

Table 3(a) shows the results of the two LUR models developed by PyLUR and RLUR respectively. The two regression equations have exactly the same predictor variables; however, their corresponding coefficients are slightly different in the two models. This is caused by the use of different spatial analysis packages for the predictor variable generation step in PyLUR and RLUR. In PyLUR, Python version binding is used to extract value of each predictor variable. In RLUR, R version binding is used to extract value of each predictor variable. The input GIS data is the same, but the outputs from these two bindings are not exactly the same (means the value of a predictor variable extracted by PyLUR and RLUR could be slightly different). Several potential predictor variables randomly selected at a few monitoring sites are extracted manually in

Table 3 Summary of model structure (a), validation (b) and time used (c) for each LUR model developed by PyLUR and RLUR respectively

(a) Model structure

Model	Coefficient	Std.error	p-Value	VIF
PyLUR model				
Intercept	2.509690e + 01	2.8122	< 0.001	–
HEAVYTRAFLOAD_300	7.404403e–06	< 0.001	< 0.001	1.4190
MAJORROADLENGTH_25	1.657081e–01	0.0330	< 0.001	1.2802
LDRES_1000	4.034466e–06	< 0.001	0.0011	1.3010
NATURAL_5000	4.188823e–07	< 0.001	0.0045	1.1074
RLUR model				
Intercept	2.519481e + 01	2.9810	< 0.001	–
HEAVYTRAFLOAD_300	7.777755e–06	< 0.001	< 0.001	1.4150
MAJORROADLENGTH_25	1.657495e–01	0.0341	< 0.001	1.2735
LDRES_1000	3.821112e–06	< 0.001	0.0035	1.4020
NATURAL_5000	6.045010e–07	< 0.001	0.0118	1.1942

(b) Model validation

Model	Model R^2	Model RMSE	LOOCV R^2	LOOCV RMSE
PyLUR model	0.79	5.13	0.73	5.45
RLUR model	0.78	5.24	0.71	5.76

(c) Computation time

Model	Potential predictor variable generation	Prediction	
		500 m-resolution (357 points)	10 m-resolution (168597 points)
PyLUR	1 min 04 sec	40 sec	2 h 14 min 21 sec
RLUR	3 min 56 sec	19 min 45 sec	After 1.5 d crashed

ArcGIS 10.5 to check the correctness of potential predictor variables generated by PyLUR and RLUR. This check shows values of potential predictor variables generated in PyLUR are the same or very close to the results manually generated in ArcGIS 10.5. However, there are slight but still acceptable biases between the RLUR results and ArcGIS 10.5 ones. This is owing to the fact that the spatial data reading and analysis libraries (GDAL/OGR) used in PyLUR are the same as those in ArcGIS 10.5. The spatial analysis libraries in RLUR are slightly different versions (rgdal, rgeos, sp).

As shown in Table 3(b), both PyLUR and RLUR can achieve a similar model fitting R^2 (PyLUR: 0.79 and RLUR: 0.78, a higher R^2 means the model can explain more variability). PyLUR’s model RMSE of 5.13 is slightly smaller than 5.24 of RLUR (smaller RMSE means a higher accuracy of the model). Comparing the performance of LUR models from PyLUR and RLUR in terms of LOOCV R^2 and LOOCV RMSE, a similar conclusion can be drawn. Even though the differences are small, and not statistically significant, the results still indicate PyLUR outperforms RLUR for modeling in this case. It should be noted that in reality, the superior performance of PyLUR over RLUR is probably better than that shown in Tables 3(a) and 3(b) as it allows more potential predictor variables to be considered in the model development (in this case for comparison PyLUR is constrained to only use potential predictor variables which can also be generated by RLUR).

The comparison of PyLUR and RLUR in terms of data processing efficiency and software stability is carried out based on the time cost for potential predictor variable generation and model prediction of heavy workloads (concentration mapping at a fine-resolution), respectively. The model selection and validation in RLUR have to be done manually, so these two steps are excluded in the comparison. The time needed for running PyLUR and RLUR is shown in Table 3(c). It took nearly four times longer for RLUR to generate all potential predictor variables than that of PyLUR. Furthermore, PyLUR is much more efficient than RLUR in predicting the spatial distribution of the pollutant. It took RLUR 19’45” to complete the prediction for the urban area (10 km^2) of Bradford at a spatial resolution of 500 m. This duration is 27 times longer than 40 s needed by PyLUR. When the resolution of the concentration map was increased to 10 m, requiring prediction at 168597 points, it took PyLUR 2 h 14 min 21 sec to complete the prediction process. Figure 5 shows the NO_2 concentration map for the urban area of Bradford, UK at a 10m-resolution. However, RLUR crashed each time after 1.5 days' unfinished prediction running.

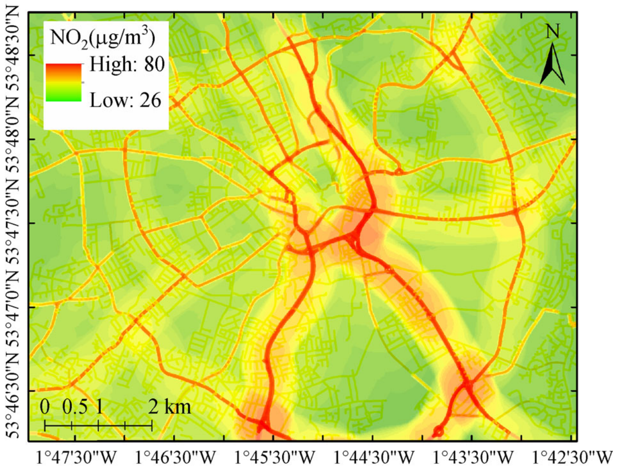


Fig. 5 A distribution map (10 m-resolution) of NO_2 concentration (μg/m^3) for the urban area of Bradford, UK generated by PyLUR. No similar results can be obtained from RLUR because it crashed after running for 1.5 days.

To find the reason of RLUR's relative inefficiency and tendency to crash, its source code was inspected and the change of RAM during the running of RLUR was also monitored. It was found during testing that the difference of running speed for PyLUR and RLUR occurred mostly during spatial data processing (predictor variable generation in the modeling and prediction parts respectively). For RAM usage situations: during the running of PyLUR, the memory in use stayed at a relative stable level throughout the entire prediction processing. However, during the running of RLUR, the memory in use would gradually increase with the increment of the amount of predictions until the program crashed at some point.

Comparing spatial analysis packages used and program designs between them, both used GDAL/OGR packages (PyLUR uses a Python version binding of GDAL/OGR package; RLUR uses an R version binding rgdal/ogr package) to manipulate geospatial data. These two version bindings both can be used to process geospatial data and carry out spatial analysis. However, there are significant differences between them and the differences are not just from the syntax aspect but also from the data processing logic aspect. For R version, the binding is adapted to the R style. The spatial data processing is data-frame- and layer-based in R. However, for Python version, the binding is more complex and requires the user to have a much deeper understanding of GIS principle. The spatial data processing is more likely to be feature-based as the Python GDAL/OGR binding is a very fundamental package not like the R version has been somehow adapted to the R style. This means that one needs to write codes for the layer-based operation based on these feature-based fundamental functions.

Two factors might lead to the difference of processing efficiency and software stability between PyLUR and RLUR. On the one hand, in RLUR, built-in layer-based functions are used directly to do spatial analysis. In PyLUR, one needs to first understand the entire inside procedures and then use some feature-based functions to self-define the layer level operations, which could reduce unnecessary procedures in a built-in layer-based function and save some time in some cases. On the other hand, in RLUR, arguments passed into a spatial analysis sub-function are usually represented as layer-based objects (the whole data-frame structure or a row of the data-frame). In PyLUR, feature objects are usually passed into a sub-function. Generally, passing a feature (or a temporary layer containing only a single feature) needs less time than passing data-frame structures. Considering the huge workload of spatial analysis needed in the fine-resolution prediction and mapping, these two factors, especially the latter one, could cause issues such as processing inefficiency and poor software stability.

To explore the performance and compare them further, the RLUR is applied to run the Auckland test data (used in Section. 3). This comparsion is constrained in potential predictor variable generation step since not all input GIS data of Auckland test sets are supported by RLUR (developed LUR model structures could be totally different). The same potential predictor variables are also generated in PyLUR to compare the efficiency of data processing. It took PyLUR 4′13” and RLUR 41′46” to complete this task, respectively. Thus, PyLUR is 10 times faster than RLUR in this case. Compared with the results in Table 3 for the Bradford test data (48 sites, it took PyLUR 1′04”), PyLUR is four times longer in generating the same potential predictor variables for the Auckland data set that is 17.5 times (70 MB vs 4 MB) the size of the Bradford

Table 4 Comparison of the size between Auckland and Bradford test data

Input GIS data	Auckland test data		Bradford test data	
	Size of file	Number of feature	Size of file	Number of feature
Land use	47311 KB	16984	170 KB	183
Road network	22596 KB	59252	3354 KB	30611
Population	450 KB	10456	434 KB	15845

data set. However, it took RLUR more than 10 times longer to complete the same task (Bradford: 3′56”, Auckland: 41′46”). Table 4 shows the size of GIS input data from Auckland and Bradford test data sets, respectively. The Auckland test data is larger; therefore, it needs more computation time for the same spatial analysis. This comparison shows that PyLUR is more capable of manipulating large volume GIS input data. The advantage of PyLUR’s computation efficiency in data processing is more obvious when the input data is much larger in volume.

5 Conclusions

In this contribution, we present a piece of self-developed software named PyLUR for LUR modeling and pollution mapping using GDAL/OGR libraries based on Python platform. Tests and comparisons show that PyLUR has a better performance in data processing efficiency and software stability. The innovations and advantages of PyLUR are: (a) it supports more kinds of potential predictor variables in the modeling; (b) an automatic regression modeling module is considered; (c) it could handle detailed GIS data and output fine-resolution predictions efficiently and stably; (d) it is open-sourced and the source code could benefit researchers who is interested in LUR modeling or programming using GDAL/OGR Python binding. A limitation of PyLUR is the lack of a sophisticated GUI like RLUR. However, this can be overcome by undertaking GIS data exploration and visualization of the output concentration maps separately in commercial GIS software. At present, PyLUR will be released and open-sourced as ‘PyLUR 1.0’. The authors are making efforts to develop a user-friendly GUI for PyLUR and would upgrade it to ‘PyLUR 2.0’ in the near future.

Acknowledgements This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors. The development of PyLUR was inspired by the open source software RLUR. The authors thank Prof. John Gulliver and Dr. David W. Morley for providing public accessible study materials about LUR modeling on the internet. The authors also thank Abley for providing reorganized traffic volume data in Auckland.

References

Akita Y (2014a). LURTools: ArcGIS Toolbox for Land Use Regression (LUR) Model, Available online at the website of www.unc.edu/~akita/lurtools

Akita Y, Baldasano J M, Beelen R, Cirach M, De Hoogh K, Hoek G, Nieuwenhuijsen M, Serre M L, De Nazelle A (2014b). Large scale air pollution estimation method combining land use regression and chemical transport modeling in a geostatistical framework. Environmental Science & Technology, 48(8): 4452–4459

Beelen R, Hoek G, Vienneau D, Eeftens M, Dimakopoulou K, Pedeli X, Tsai M Y, Künzli N, Schikowski T, Marcon A, Eriksen K T, Raaschou-Nielsen O, Stephanou E, Patelarou E, Lanki T, Yli-Tuomi T, Declercq C, Falq G, Stempfelet M, Birk M, Cyrys J, von Klot S, Nádor G, Varró M J, Dėdelė A, Gražulevičienė R, Mölter A, Lindley S, Madsen C, Cesaroni G, Ranzi A, Badaloni C, Hoffmann B, Nonnemacher M, Krämer U, Kuhlbusch T, Cirach M, de Nazelle A, Nieuwenhuijsen M, Bellander T, Korek M, Olsson D, Strömgren M, Dons E, Jerrett M, Fischer P, Wang M, Brunekreef B, de Hoogh K (2013). Development of NO_2 and NO_x land use regression models for estimating air pollution exposure in 36 study areas in Europe–The ESCAPE project. Atmospheric Environment, 72: 10–23

Briggs D J, Collins S, Elliott P, Fischer P, Kingham S, Lebret E, Pryl K, Van Reeuwijk H, Smallbone K, Van Der Veen A (1997). Mapping urban air pollution using GIS: A regression-based approach. International Journal of Geographical Information Science, 11(7): 699–718

European Study of Cohorts for Air Pollution Effects (2010). ESCAPE exposure assessment manual. Available online at the website of www.escapeproject.eu/manuals

Hoek G, Beelen R, De Hoogh K, Vienneau D, Gulliver J, Fischer P, Briggs D (2008). A review of land-use regression models to assess spatial variation of outdoor air pollution. Atmospheric Environment, 42(33): 7561–7578

Keller J P, Olives C, Kim S Y, Sheppard L, Sampson P D, Szpiro A A, Oron A P, Lindström J, Vedal S, Kaufman J D (2015). A unified spatiotemporal modeling approach for predicting concentrations of multiple air pollutants in the multi-ethnic study of atherosclerosis and air pollution. Environmental Health Perspectives, 123(4): 301–309

Kim J H (2009). Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap. Computational Statistics & Data Analysis, 53(11): 3735–3745

Kohavi R (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In: International Joint Conference on Artificial Intelligence (IJCAI), 14(2), 1137–1145

Li S, Zou B, Fang X, Lin Y (2019). Time series modeling of $PM_{2.5}$ concentrations with residual variance constraint in eastern mainland China during 2013–2017. Science of the Total Environment, doi: 10.1016/j.scitotenv.2019.135755

Liu W, Li X, Chen Z, Zeng G, León T, Liang J, Huang G, Gao Z, Jiao S, He X, Lai M (2015). Land use regression models coupled with meteorology to model spatial and temporal variability of NO_2 and PM_{10} in Changsha, China. Atmospheric Environment, 116: 272–280

Liu Z, Xie M, Tian K, Gao P (2017). GIS-based analysis of population exposure to $PM_{2.5}$ air pollution: A case study of Beijing. Journal of Environmental Sciences (China), 59: 48–53

Ma X, Longley I, Gao J, Kachhara A, Salmond J (2019). A site-optimised multi-scale GIS based land use regression model for simulating local scale patterns in air pollution. Science of the Total Environment, 685: 134–149

Marcon A, de Hoogh K, Gulliver J, Beelen R, Hansell A L (2015). Development and transferability of a nitrogen dioxide land use regression model within the Veneto region of Italy. Atmospheric Environment, 122: 696–704

Masiol M, Zíková N, Chalupa D C, Rich D Q, Ferro A R, Hopke P K (2018). Hourly land-use regression models based on low-cost PM monitor data. Environmental Research, 167: 7–14

Meng X, Chen L, Cai J, Zou B, Wu C F, Fu Q, Zhang Y, Liu Y, Kan H (2015). A land use regression model for estimating the NO_2 concentration in Shanghai, China. Environmental Research, 137: 308–315

Miskell G, Salmond J, Longley I, Dirks K N (2015). A novel approach in quantifying the effect of urban design features on local-scale air pollution in central urban areas. Environmental Science & Technology, 49(15): 9004–9011

Miskell G, Salmond J A, Williams D E (2018). Use of a handheld low-cost sensor to explore the effect of urban design features on local-scale spatial and temporal air quality variability. Science of the Total Environment, 619-620: 480–490

Morley D W, Gulliver J (2018). A land use regression variable generation, modelling and prediction tool for air pollution exposure assessment. Environmental Modelling & Software, 105: 17–23

Muttoo S, Ramsay L, Brunekreef B, Beelen R, Meliefste K, Naidoo R N (2018). Land use regression modelling estimating nitrogen oxides exposure in industrial south Durban, South Africa. Science of the Total Environment, 610-611: 1439–1447

Open Source Geospatial Foundation (2008). GDAL-OGR: Geospatial Data Abstraction Library/Simple Features Library Software, Available online at https://www.gdal.org/

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12: 2825–2830

Sanner M F (1999). Python: A programming language for software integration and development. Journal of Molecular Graphics & Modelling, 17(1): 57–61

Saucy A, Röösli M, Künzli N, Tsai M Y, Sieber C, Olaniyan T, Baatjies R, Jeebhay M, Davey M, Flückiger B, Naidoo R, Dalvie M, Badpa M, de Hoogh K (2018). Land use regression modelling of outdoor NO_2 and $PM_{2.5}$ concentrations in three low income areas in the Western Cape Province, South Africa. International Journal of Environmental Research and Public Health, 15(7): 1452-1465

Seabold S, Perktold J (2010). Statsmodels: Econometric and statistical modeling with python. In: Proceedings of the 9th Python in Science Conference, 57, 61

Weissert L F, Salmond J A, Miskell G, Alavi-Shoshtari M, Williams D E (2018). Development of a microscale land use regression model for predicting NO_2 concentrations at a heavy trafficked suburban area in Auckland, NZ. Science of the Total Environment, 619-620: 112–119

Westra E (2013). Python geospatial development. Birmingham: Packt Publishing Ltd.

Wu J, Li J, Peng J, Li W, Xu G, Dong C (2015). Applying land use regression model to estimate spatial variation of $PM_{2.5}$ in Beijing, China. Environmental Science and Pollution Research International, 22(9): 7045–7061

Xu H, Bechle M J, Wang M, Szpiro A A, Vedal S, Bai Y, Marshall J D (2019a). National $PM_{2.5}$ and NO_2 exposure models for China based on land use regression, satellite measurements, and universal kriging. Science of the Total Environment, 655: 423–433

Xu S, Zou B, Lin Y, Zhao X, Li S, Hu C (2019b). Strategies of method selection for fine-scale $PM_{2.5}$ mapping in an intra-urban area using crowdsourced monitoring. Atmospheric Measurement Techniques. 28; 12(5):2933–48

Zhai L, Zou B, Fang X, Luo Y, Wan N, Li S (2016). Land use regression modeling of $PM_{2.5}$ concentrations at optimized spatial scales. Atmosphere, 8(1): 1–15

Zou B, Pu Q, Bilal M, Weng Q, Zhai L, Nichol J E (2016). High-resolution satellite map- ping of fine particulates based on geographically weighted regression. IEEE Geoscience and Remote Sensing Letters, 13(4): 495–499