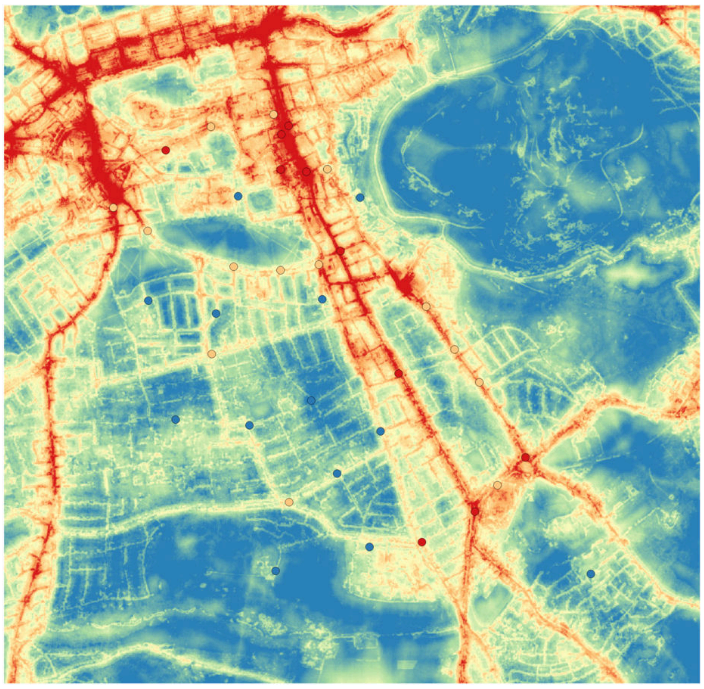
Doctoral School in Environmental Engineering

A spatial decision support system to assess personal exposure to air pollution integrating sensor measurements

Pietro Zambelli



UNIVERSITÀ DEGLI STUDI DI TRENTO

Dipartimento di Ingegneria Civile e Ambientale

March 2015

Doctoral thesis in **Environmental Engineering**, 26th **cycle**

Faculty of Engineering, **University of Trento**

Academic year 2014/2015

Supervisor: **Prof. Marco Ciolli, Prof. Marco Ragazzi**

University of Trento

Trento, Italy

31st January 2015

To Irene and Lorenzo,
with love.

Facing it, always facing it,
that’s the way to get through.
Face it.

Joseph Conrad

Acknowledgments

This Ph.D. thesis contains the result of research undertaken at the Department of Environmental Engineering of the University of Trento. Without the financial support of Dolphin S.r.l. this project would never have started; therefore my gratitude goes to Mauro Chelodi.

I would have never reached the point of finishing my dissertation without the help and support of others. First and foremost, I offer my gratitude to my supervisors: Marco Ciolli, who has supported and guided me through a path full of pitfalls such as the Ph.D. experience is, and Marco Ragazzi, who has believed in the project idea.

I thank Stafan Reis for his valuable help in revising and commenting the work and the material presented in this thesis; his hints and critiques were always precious. Massimo Vieno was the person that welcomed me at CEH and gave me the opportunity to present my work at the "EMEP - group", (I know that the name has changed, but in my heart it will remain the "EMEP - group"). And of course a special thanks to the whole group that patiently listened to my presentations on image classification of urban micro-environments. Without the work and support that have been provided by Mathew Heal and Lin Chun and their colleagues to organize and collect the data from the Passive Diffusive Tubes, this thesis would hae had a different shape, so thank you. A special thanks to a warm and wonderful person, the best presenter of the University of Edinburgh, Andrea Moring, thanks for the long chats that we had on the bus.

In addition, I thank Markus Neteler for his hints and ideas and SÃűren Gebbert, who supervised me during my Google Summer of Code (2012), for his kindness and availability.

These four years have been a challenging trip, with both ups and downs. Fortunately, I was not alone on this road, but accompanied by Irene, my wife, who patiently endured me all those times when, instead of going out to enjoy our free time, I was thinking, reading, and coding for my thesis; by Lorenzo, my son, who forced me to hurry up the end of this manuscript, and by my family, who always encouraged me.

Finally a special thanks to all the open source communities that supported me during these long years, in particular the GRASS-developers and Python communities, they inspire and stimulate me with new ideas and tools.

Contents

List of Figures

List of Tables

Preface

All of the work presented henceforth was conducted between the Department of Civil and Environmental and Mechanical Engineering at the University of Trento, the Centre for Ecology and Hydrology (CEH) of Edinburgh and the European Research Academy (EURAC) of Bolzano. The first three years of the PhD. were funded by Dolphin S.R.L. three months were covered by CEH and the last year by EURAC. All projects and associated methods were discussed with and approved by my supervisors: Marco Ciolli and Marco Ragazzi.

The work is split in seven main chapters sequenced to logically represent the of investigation I followed during my study. They can be read sequentially or separately.

Chapter 1 introduces the problem of the spatial variability of air pollutant characterization, providing an overview of the state of the art in terms of technology and modelling;

Chapter 2 focuses on emerging low-cost sensors for outdoor quality monitoring and describes the result of a two-week campaign for Carbon Monoxide (CO), which we carried out in the city of Trento (Italy). It explains the limits and potential of this technique;

Chapter 3 introduces PyGRASS, an object-oriented python library for GRASS GIS, which I developed specifically to facilitate the following phases of data analysis and classification;

Chapter 4 describes in detail the methodology used to classify urban micro-environments combining Geo-Object-Based Image Analysis (GEOBIA) and machine-learning algorithms;

Chapter 5 ranks and compares the correlation between geographical factors and measured data of a six-week campaign collecting Nitrogen Dioxide (NO_2) concentrations, using passive sample, in the city of Edinburgh. It explains the methodology followed to assess NO_2 concentrations combining machine-learning regression methods and geographical features;

Chapter 6 takes advantage of the results of the modelling, proposing three different scenarios for improving the air quality in the neighbourhood of a kindergarten in Edinburgh;

Chapter 7 describes the main advantages and disadvantages of the presented methodology and the limits of its applicability in other contexts.

A version of Chapter two has been published in: *Rada E.C., Chisté A., Schiavon M., Ciolli M., Zambelli P., Brini M., Marmo L., Chelodi M. 2012 "A contribution to the development of a public – private integrated network of wireless sensors for an enhanced management of air quality" in Sustainable Technology for Environmental Protection, Milano, 2012, p. [1-4]. - ISBN: 9788890355721. Proceedings: SIDISA2012, Milano, 26-29 June 2012.* and *Rada E.C., Ragazzi M., Brini M., Marmo L., Zambelli P., Chelodi M., Ciolli M. 2012 "Perspectives of low-cost sensors adoption for air quality monitoring" in Scientific Bulletin - "Polithenica" University of Bucharest. Series D, Mechanical Engineering, v. 74, n.2, p 241-250.*, I was responsible for the data analysis and comparison between data coming from the official measurement stations and low-cost sensors.

A version of Chapter three has been published in: *Zambelli P., Gebbert S., Ciolli M. 2013 "Pygrass: an object-oriented Python API for GRASS GIS". ISPRS International Journal of Geo-Information 2, p. 201–219. doi:10.3390/ijgi2010201.* This part of the work has been funded by Google within the Google Summer of Code 2012. I wrote the code and built the main benchmarks, S. Gebbert was the supervisor author of the code and M. Ciolli the supervisor for the manuscript composition.

A version of Chapter four has been presented at *Zambelli, P., Vieno, M., Reis, S., Ragazzi, M., Ciolli, M., 2013. "Application of machine-learning techniques for the characterization of urban micro-environments to assess spatial variability of air pollutants". Urban Environmental Pollution 2013 Asian Edition (UEP2013), Creating Healthy, Liveable Cities, 17-20 November 2013, Beijing, China* and *Reis, S., Zambelli, P., Vieno, M. and Reis, S. 2013 "Developing a virtual observatory for personal exposure assessment", 20th International Congress on Modelling and Simulation(MODSIM2013), Adelaide, South Australia, from Sunday 1 to Friday 6 December 2013.* At the moment a version of the chapter is under review in *Zambelli, P., Vieno, M., Reis, S., Neteler, M., Ragazzi, M., Ciolli, M. XXXX. "Machine learning classification of urban micro-environment: the case of Edinburgh, Scotland" under revision in Transaction in GIS.* Concerning the characterization and extraction of urban micro-environments I was the lead investigator, responsible for all major areas of concept formation, data collection and analysis, as well as the bulk of the manuscript composition. Marco Ciolli provided the contact with CEH researchers M. Vieno and S. Reis who were involved in the early stages of concept formation and contributed to manuscript edits. M. Neteler provided ideas for data analysis and M. Ciolli and M. Ragazzi were the supervisory authors.

I was the lead investigator for the projects found in Chapters five and six, where I was responsible for all major areas of concept formation, data collection and analysis, as well as a majority of the manuscript composition. This part of the work has not been published yet, but a version of Chapter 4 is under development in: *Zambelli, P., Vieno, M., Reis, S., Heal, M.R., Lin, C., Ragazzi, M., and Ciolli, M. XXXX. "Characterization of urban micro-environment for NO2: the case of Edinburgh, Scotland"*. Heal and Lin were responsible for the design of the measurement campaign with Passive Samplers and data collection. I participated in one of the six weeks of data collection. I was responsible for the data and GIS analysis as well as a majority of the manuscript composition.

Marco Ciolli and Marco Ragazzi were the supervisory authors on this project and were involved throughout the project in both concept formation and manuscript edits.

Summary

Recent epidemiological studies have reinforced the link between short and long-term exposure to air pollutants and adverse effects on public health especially over the weaker part of the population, like children and older adults. The creation of simple tools to locate sensible areas as well as of dedicated Spatial Decision Support System (SDSS) to improve the management of pollution risk areas system is strongly advised.

The aim of this work is to develop a SDSS methodology, based on easy to find data and usable by decision makers, to assess and reduce the impact of air pollutants in a urban context. To achieve this goals I tested the exploitability of a set of low-cost sensors for outdoor air quality monitoring, I characterized the urban micro-environments and the spatial variability of air pollutants using remote sensing compared to field data and eventually I developed a SDSS to improve the public health designing and comparing different scenarios.

The city centre of Edinburgh has been used as study case for the purposed methodology. To test the reliability and applicability of low cost sensors as proxies for remote sensed data, we conducted a measurements campaign to compare the observed data between an official measurements station (OMS) in Trento (Italy) and electrochemical and thick film sensors respectively of Carbon Monoxide (CO) and Ozone (O_3). Due to data quality and availability we decided to characterize the urban micro-environments of Edinburgh (Scotland, UK) in eight main classes (water, grass, vegetation, road, car, bus, buildings and shadow) combining the Geographic Object-Based Image Analysis (GEOBIA) with Machine Learning algorithms to process the high resolution (0.25m x 0.25m) RGB aerial ortho-rectified images. This land-use characterization combined with other geographical informations, like the classification of the roads and the urban morphology, were compared with 37 Nitrogen Dioxide (NO2) concentration data, collected using passive tubes during a six week campaign of measurements conducted by the school of Chemistry of the University of Edinburgh. I developed a new open-source GIS python library (PyGRASS), integrated in the stable release of GRASS GIS, to speed-up the prototyping phase and to create and test new GIS tools and methodologies. Different studies on SDSS were carried out to implement procedures and models. Based on these models and data all the factors (land-use, roads and geo-morphological features) were ranked to identify which are driving forces for urban air quality and to help decision makers to develop new policies.

The sensor tested in Trento revealed an evident drift in measurement residues for CO, furthermore the measurements were also quite sensitive to external factors such as temperature and humidity. Since these sensors required frequent recalibration in order to obtain reliable results, their use was not as low-cost as expected. The characterization of urban land-use in Edinburgh with GEOBIA and machine learning provided an overall accuracy of 93.71% with a Cohen's k of 0.916 using a train/test dataset of 9301 objects. The NO_2 data confirm the assumption that air concentration is strongly dependent on geographical position and it is strongly influenced by the position of the pollutant's source. Using the results of the tests and remote sensing analysis, I developed an SDSS. Starting from the current situation, I designed three scenarios to assess the effect that different policies and actions could have on improving air quality at on the local and district level.

The outcomes of this work can be used to define and compare different scenarios and develop effective policies to reduce the impact of air pollutants in an urban context using simple and easy to find data. The GIS-based tool can help to identify critical areas before deploying sensors and splitting the study area in homogeneous micro-environments clusters. The model is easy to expand following different procedures.

Chapter 1

Introduction

1.1 Status quo, context and motivations

An increasing body of literature underlines how air pollution has a negative influence on human health (Curtis et al., 2006; Janssen et al., 2013) in particular for weaker parts of the population such as foetuses, children and older adults. The effects of pollutants are more evident and temporally much closer to children due to their higher metabolism than adults (Coneus and Spiess, 2012). Recent studies highlight that maternal exposure to pollutants increases adverse pregnancy outcomes (Dadvand et al., 2011; Gomez-Mejiba et al., 2009), hence there is great interest from the science community to improve knowledge of the consequences of air pollution exposure on child health and deepen the environmental injustice implications (Jephcote and Chen, 2012; Meijer et al., 2012; Schoolman and Ma, 2012; Stuart and Zeager, 2011).

The World Health Organisation (WHO) assesses that around 7 million of premature deaths are due to air quality, in particular around 3.7 million are due to outdoor concentrations of air pollutants (World Health Organization (WHO), 2014a). The pollutants of major public health concern include particular matter (PM_{10}, $PM_{2.5}$, $PM_{0.1}$), carbon monoxide (CO), ozone (O_3), nitrogen dioxide (NO_2) and sulphur dioxide (SO_2). Reducing emissions from stationary sources (e.g. power plants); reducing emissions of road traffic sources; improving waste management; reducing energy consumption and increasing production from renewable energies are all examples of successful emission control policies (World Health Organization (WHO), 2014b). However, exceeding air quality limit values, in particular in urban areas and near busy roads, is still an open issue in many parts of the world.

1.1.1 Monitoring sensor network

The main technologies available to characterize the space and time variabilities of air pollutants in an urban context are: Official Measurements Stations (OMS), passive samplers, low-cost sensors, moving sensors and numerical models. All these solutions are different in terms of: cost, accuracy, spatial and time resolution.

The OMS are characterized by a very high accuracy and time resolution providing data in real-time, but also by high investments and maintenance costs, limiting the number of observation points and therefore limiting the spatial resolution. As evidenced by Sheppard et al. (2005) using central site monitor data will affect health effect estimates. Traditional technologies provide poor information on the spatial concentration variability of air pollutants especially in an urban context.

Passive samplers (Kardel et al., 2012; Sally Liu et al., 2012), and bio-indicators (Ram et al., 2012; Salo et al., 2012) such as lichens (Käffer et al., 2012; Llop et al., 2012; Salo et al., 2012; Stevens et al., 2012) increase the number of measurement points improving the spatial characterization of air pollutants, but generally with a lower accuracy and time resolution and requiring more time than other systems to collect and analyse the samplers.

Recent studies have started to investigate the use of low-cost sensors for air quality monitoring (De Vito et al., 2011; Hu et al., 2011). As highlighted in Rada et al. (2012) combining the information coming from the Wireless Sensor Network (WSN) can improve characterization of air pollutant concentrations in the micro-environment in both spatial and temporal variability. A good Sensor Network is necessary (Borrell, 2011) for assessing an emerging scientific concept defined by Wild (2005) as *exposome*. *Exposome* is a term used to represent all kind of exposure (including exposure to diet, lifestyle, and endogenous sources) and aims to identify the combined effects of genetic and environmental factors on chronic diseases (Gasiewicz, 2010; Hamzelou, 2011).

An emerging monitoring technique is to use moving sensors; this system couples sensors characterized by a high degree of accuracy and time resolution with a moving platform (e.g. taxi, tram, bus, etc.) to be able to increase the spatial resolution of the study area. What it is challenging in this approach is handling and interpolating data values that are changing in space and time.

1.1.2 Model spatial variability

Steinle et al. (2013) highlight that the spatio-temporal variability of personal exposures in urban micro-environments is very high. Fixed monitoring stations are not sufficiently informative to characterize this variability, and recent studies are moving from static monitoring to spatio-temporally resolved personal exposure assessment, collecting data of air pollutants concentrations while people are conducting their daily activities. However it is not possible to measure everything everywhere, therefore we need to identify a methodology to model personal exposure in highly variable urban concentration fields. Many studies confirm that the spatial temporal variability of ambient air pollution could affect the assessment of health effects on the population (EPA, 2010) and try to characterize the variability of micro-environments (Can et al., 2011; Dons et al., 2011; Mölter et al., 2012; Richmond-Bryant et al., 2011).

In most of the used cases, measuring concentrations of air pollutants is not enough to assess the exposure of a population in a certain area. To transform the information points from points to a surface, several methods are available; the most used in air quality literature are: the *Kriging* interpolation method (Baume et al., 2011; Beelen et al., 2009; Briggs et al., 2000; Guo et al., 2007; Janssen et al., 2008; Marchetti et al., 2011; Motaghian and Mohammadi, 2011; Pearce et al., 2009; Shad et al., 2009; Singh et al., 2011; Vienneau et al., 2009); the *Land Use Regression* (LUR) method (Arain et al., 2007; Beelen et al., 2010; Crouse et al., 2009; Hoek et al., 2008; Johnson et al., 2010; Mölter et al., 2010; Mukerjee et al., 2009; Ross et al., 2007; Su et al., 2008b); and *numerical dispersion modelling* (Allwine et al., 2006; Berkowicz et al., 2008; Carruthers et al., 2000; Kesarkar et al., 2007).

The first two methodologies are based on geo-statistical analysis, the third one resolves numerically the physical equations of air pollutants dispersion. If enough information is available concerning emissions and meteorology, dispersion models are well suited to short and long term exposure modelling. Many different dispersion models have been developed, but in broad terms, few of them are able to provide a detailed map over a large study area. The most used dispersion models in literature are: *ADMS-Urban* (Carruthers et al., 2000), *AERMOD* (Kesarkar et al., 2007), *DUSTRAN* (Allwine et al., 2006) and *OSPM* (Berkowicz et al., 2008). As highlighted by Gulliver and Briggs (2011) three main limiting factors characterize the applicability of these models: they require a huge amount of data (e.g. detailed data on source emissions and boundary-layer meteorology), they are often expensive, they do not simultaneously deal with large numbers of emission sources and generally tens of model runs are required to map air pollutants at high spatial resolution ($< 50m$).

The Land Use Regression method demonstrates good results on pollutant interpolations particularly to model the spatial variation of annual average concentrations of Particular Matter (*PM*) (Eeftens et al., 2012), Black Carbon (*BC*) (Dons et al., 2013), Nitrogen Dioxide (NO_2) (Mukerjee et al., 2009).

1.1.3 Decision support system and air quality

From a literature review, it seems that not many Decision Support Systems (DSS) have been developed to help urban dwellers, planners and policy-makers to test and compare different scenarios. Most of these DSSs focus on the national/regional scale with a spatial resolution that is not suitable/adaptable to an urban scale (Carnevale et al., 2012a,b; Elbir et al., 2010; Finardi et al., 2008; Gidhagen et al., 2013; Vedrenne et al., 2014). Vlachokostas et al. (2011) developed a multi-criteria methodological approach to managing air pollution at an urban scale but the spatial variable is not taken into account in the analysis. Mavroulidou et al. (2004) developed a DSS using an interaction matrix to assess qualitatively the concentration of air pollutants at an urban scale and provide a vulnerability map, but this final map still had quite low spatial resolution. González et al. (2013) presented a more general approach to developing a DSS for a sustainable urban metabolism that considers the problem in broader terms (e.g. social quality of life, economic performance, environmental protection).

1.2 Aims and objectives

This dissertation aims to develop a method to derive very high-resolution urban concentrations for personal exposure assessment, reflecting the spatio-temporal variability of urban pollution fields. The research questions addressed in this dissertation are the following:

- Are low-cost sensors ready for outdoor air quality monitoring?
- How can we characterize different urban micro-environments for air pollutants? Which geographical factors are more important?
- Can we enhance the Land Use Regression (LUR) method to take into account and integrate information on micro-environments and other factors that are drivers of air pollutant concentrations in an urban context?
- Can we develop a Spatial Decision Support System (SDSS) to assess and compare the impact of different policies and scenarios of urban management?

1.3 Methodology

During the PhD I have contributed to select areas that are suitable for the design of low-carbon settlements using a Spatial Multi Criteria Analysis (SMCA) (Vettorato et al., 2011) and developed a Spatial Decision Support System (SDSS) to assess the forestry biomass potential for energy exploitation (Sacchelli et al., 2014; Sacchelli et al., 2013;

Zambelli et al., 2012). Starting from this experience I developed an SDSS to assess the impact on air pollutant concentrations of different policies and scenarios. But in order to develop an SDSS, several intermediate steps needed to be addressed.

- Low-cost sensors were compared with measurements from the Official Measurement Station (OMS) to test reliability of this emerging technology.
- The urban micro-environments were extracted from high resolution aerial images.
- The correlation between geographical factors and pollutants concentration was verified using measured data of NO_2 using passive samples.
- The measured and geographical data were used to regress the concentration where the measurements were not available.
- Finally, different scenarios were made, changing the urban land use to mimic the effect of different policies and management planning decisions.

1.4 Framework

The GIS used to develop this work is the Geographic Resources Analysis Support System (GRASS), an open source software that supports creation, modification and processing of 2D and 3D raster and vector layers. It provides a topological vector model and true three dimensional coordinates for vector features analysis. GRASS is characterized by stability, an efficient application programming interface (API) written in C, and a large number of GIS functions and modules (Neteler et al., 2012). Its capabilities of processing geographical information have been evaluated and validated by many research and technical papers (Ciolli et al., 2004; Hofierka and Zlocha, 2012; Li et al., 2010; Okabe et al., 2009; Preatoni et al., 2012; Sacchelli et al., 2013; Tattoni et al., 2010, 2012; Vettorato et al., 2011; Zambelli et al., 2010, 2012).

I developed a new python interface (PyGRASS) to expand GRASS capabilities implementing a tool that gives the freedom to approach the GIS problem from a different perspective, opening the software developer's approach to GIS users and trying to maintain relative simplicity. As shown in Zambelli et al. (2013), the new Object-Oriented Python programming API introduces an abstract layer that opens the possibility for users who are not familiar with C or with GRASS C-API to use and access transparently the efficient C functions of GRASS.

The PyGRASS library has been designed to facilitate management of complex problems that need the integration of large data sets and geographical data, such as air quality, data monitoring and modelling, and to integrate/develop new methods and functions extending the GRASS functionalities. PyGRASS supplies a new interface to prototype complex scientific algorithms and simplify the interoperability with all related geospatial software and tools provided by a Python interface. The development of PyGRASS has been sponsored by Google's Summer of Code program (2012), and the code produced has been integrated in the current stable release: GRASS7.

To characterize urban micro-environments, I extracted information from high-resolution aerial photography, and I tested and used some Machine Learning algorithms for image classification and for the regression used in the LUR. The libraries used for this analysis are *sklearn* and *mlpy*. Both libraries provide a wide range of state-of-the-art machine learning methods for supervised, and unsupervised problems for regression and classification (Albanese and Visintainer, 2012; Pedregosa et al., 2011).

Chapter 2

Low-cost sensors: a new paradigm for environmental monitoring

low-cost sensors open new paradigms for air quality monitoring. Their performance allows a new strategy closer to the population and its health. Critical situations that cannot be seen with conventional approaches can emerge through a low-cost sensors network. In this chapter a comparison between electrochemical sensors and official measurement stations (OMS) data is reported. Low-cost sensors can not be used straight-forwardly in outdoor environments to measure environmental concentrations of air pollutants. An accurate characterization of sensor behaviour in local conditions during the design phase of the network and the post-processing of the collected data is required. A low-cost sensor network can help detect dangerous peaks in concentration and provide an early warning system. This system can be used to drive and develop local strategies to reduce the adverse effects of air pollutants.

2.1 Introduction

In recent years the European Union regulation of air quality management has reached important results in term of exposure and health implications of organic and inorganic pollutants, and in terms of environment protection. However many actions remain to be developed mainly in urban areas, the general trend is towards an average improvement of air quality, with positive consequences on population health. Generally, adopted regulations for air quality management are based on the concept of protecting the environment without facing critical situations at micro-scale, where human exposure to atmospheric pollutants can reach unacceptable concentration levels.

Traditionally, air quality measurement networks take into account two instruments: the first is characterized by very high accuracy and resolution, providing data in real-time, but on the other hand, requiring high investment and maintenance costs (Sheppard et al., 2005); the second technique is based on passive samplers (Kardel et al., 2011; Sally Liu et al., 2012), bio-indicators (Ram et al., 2012) such as lichens (Llop et al., 2012; Stevens et al., 2012) that increase the number of measurement points but only provide information on average concentrations. Therefore traditional monitoring and environmental warning systems are often insufficient for planning detailed corrective actions and quickly identifying critical situations that are potentially harmful to public health.

Many studies point out how the spatial and temporal variability of ambient air pollution could affect the assessment of health effects on the population (USEPA - U.S. Environmental Protection Agency, 2010). As highlighted by Dons et al. (2011) and Richmond-Bryant et al. (2011) some micro-environments can be characterized by significant variability.

The existing regulatory monitoring network provides punctual information on the spatial and temporal concentration of air pollutants. The main factors driving space-time variability are: pollutant sources, meteorological conditions, building and vegetation topography and different land-use. Due to the higher number of factors some local temporary anomalies in air pollution cannot be detected by the conventional regulatory approach; new sensor networks can help to characterize this spatial variability.

New developments in wireless communications and micro-electro-mechanical technology have enabled the development of low-cost and multifunctional sensors for application not only in the industrial sector (Flammini et al., 2009) but also in the agricultural and environmental sphere (Alemdar and Ersoy, 2010; Burgess et al., 2010; Cao et al., 2008; Hu et al., 2011; Rawi and Al-Anbuky, 2011; Wang et al., 2006).

In this context, Wireless Sensor Networks (WSNs) can effectively improve characterization of the variability of air pollutants in space and time and consequently improve the life hazard assessment. This is why recent studies have started to investigate the use of WSNs for air quality monitoring (Hu et al., 2011; Merbitz et al., 2012). However, protocols and sensors are extremely new, and much research remains to be done to integrate and test these technologies in the outdoor environment.

This chapter analyses the opportunities and challenges of low-cost monitoring networks for improving control of the human health risk from atmospheric pollutants. The chapter presents the results of some preliminary tests to build an outdoor air quality monitoring system based on WSNs technology aimed at detecting hotspots and peaks of pollutant concentrations (NO_2, CO and O_3). The proposed network is not designed to substitute traditional monitoring systems but aims to detect peak concentrations of pollutants, to provide a new instrument for public decision makers, and to minimize the impact of air pollution on the population.

Critical case	Selected pollutants	Notes
Kindergartens	NO_2	CO could be added
Street canyon	NO_2	CO could be added
Proximity to high traffic roads	NO_2	
Proximity to urban tunnels	NO_2	
Proximity to urban canyons	NO_2	
Proximity to Industrial plants	CO, NO_2	Depending on the process
Summertime	O_3	

Table 2.1: Selected pollutants for each critical case.

2.2 Materials and methods

A few micro-scale critical situations were selected pointing out peak values of O_3, NO_2, and CO, that could be reached and the potential effects on health, in order to develop strategies and policies to improve the status of air quality and comply with National and European legislations. Generally critical situations could be found as follows: in the yards of kindergartens and schools (when an important road is present in the proximity); in street canyons (when the flux of traffic is critical); in residential areas close to highways, tunnels or above trenched roads, and in the proximity of large industrial units.

Some low-cost sensors were selected to check their viability to act as sentinels where the conventional approach of air quality monitoring cannot guarantee a high spatial density of measurements.

The tests were conducted during two weeks from 14/12/2011 until 27/12/2011 in a suburban area of Trento (via Bolzano, Gardolo).

The implementation of the tested WSNs was provided by ENVEVE SA and we used electro-chemical sensors as a tool to evaluate and characterize micro-environment concentrations in both spatial and temporal variability. A mixed network composed of two different sensors characterized by different accuracy/sensitivity for CO, O_3 and NO_2, and optional sensors for the measurement of such physical variables as temperature, humidity, wind speed and direction, have been designed for this research.

During the campaign the low-cost sensors were coupled with an official measurement station, and we tested only CO and O_3.

2.3 Results and discussion

For each critical case, different pollutants were selected: NO_2, CO and O_3. Other pollutants were excluded, due to their power consumption that seemed ill-suited for a wireless sensor network (e.g. PM_{10}). The NO pollutant was not taken into account as non-toxic. Large industrial plants are required to analyse the production process of air

Company law limits	*NO_2* *0.1@1h & 0.02@8760h*			*CO* *8.7@8h*			*O_3* *0.06@8h*		
	min	max	resolution	min	max	resolution	min	max	resolution
e2v	0.05	5	n.a.	0	500	1	n.a.	n.a.	n.a.
alphasense	0.02	50	0.015	0	2000	0.02	0	10	0.01
Nemoto	0	20	0.2	0	1000	5	n.a.	n.a.	n.a.
Xcell Sensors	0	50	0.1	0	1999	1	n.a.	n.a.	n.a.
SGX sensortech	0	20	0.1	0	500	1	n.a.	n.a.	n.a.
Dräger	0	50	0.1	0	2000	1	0	10	0.01
veris.com	0	10	0.1	0	200	1	n.a.	n.a.	n.a.
City-tech	0	50	0.1	0	500	1	0	2	0.02
Euro-gas	0	20	0.1	0	200	n.a.	0	3	n.a.

Table 2.2: Comparison of different sensors concerning range and resolution for air pollutants: NO_2, CO, O_3, all units are in ppm.

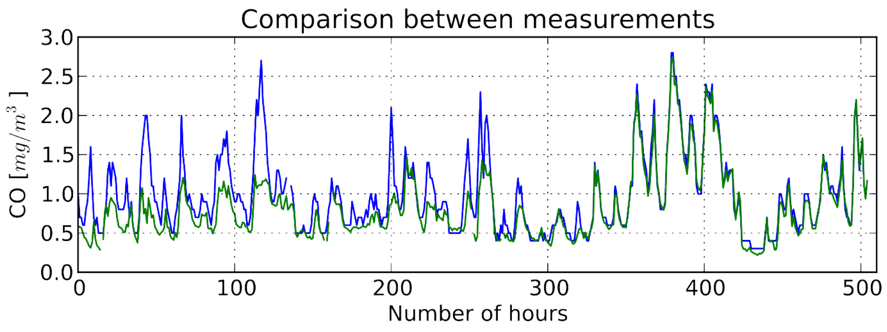


Figure 2.1: Comparison of official measurement station (blue) and low-cost sensor data (green) multiplied by a factor of 1.25.

pollutants and the way of they are released into the atmosphere, because both factors can change significantly the concentration mix and level of air pollutants. The selected pollutants for each critical case are shown in Table 2.1.

Looking at low-cost sensor available on the market (see Table 2.2), it is evident that to measure outdoor concentration of NO_2 and O_3 is challenging, due to their low resolution and because they are working always close to their detection limit.

The comparison between measurements of O_3 from the low-cost sensor and the OMS data highlight a malfunctioning of the sensor which makes the low-cost sensor data unsuitable for environmental monitoring. Thus, only data analysis for the CO will be presented.

Figure 2.1 shows the comparison between concentration data for CO collected using the official measurement station and the low-cost sensor data multiplied by a factor of 1.25 to reduce measurement shift.

To analyse the behaviour of the electrochemical sensor in Figure 2.2 the residuals between the two instruments are reported. The figure clearly shows a strong drift in the data collected by the low-cost sensor; furthermore the electrochemical sensor requires a quite long spin-up time (~ 1 week) with a significant standard deviation of $\sim 0.29\,mg$

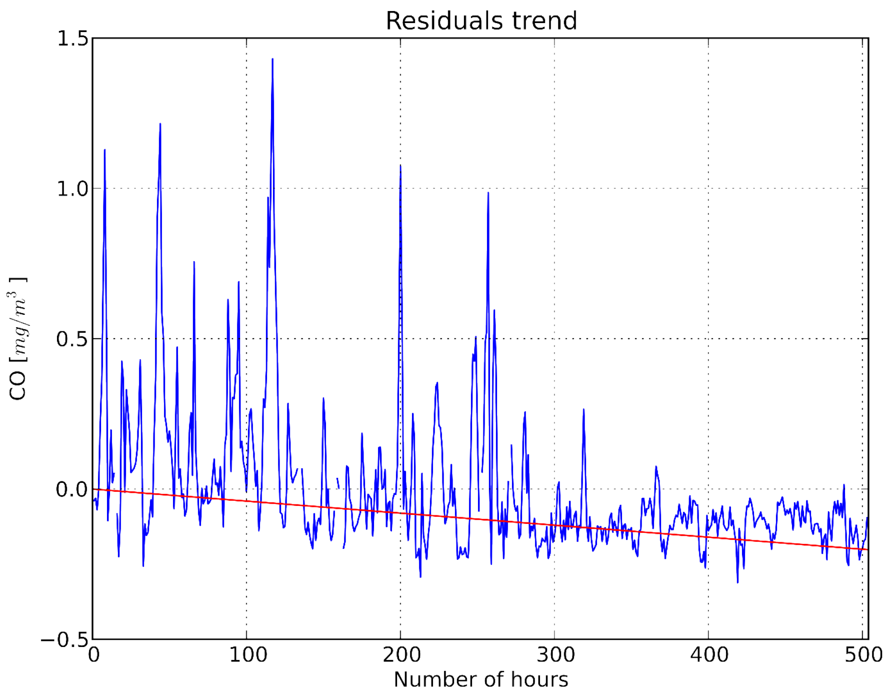
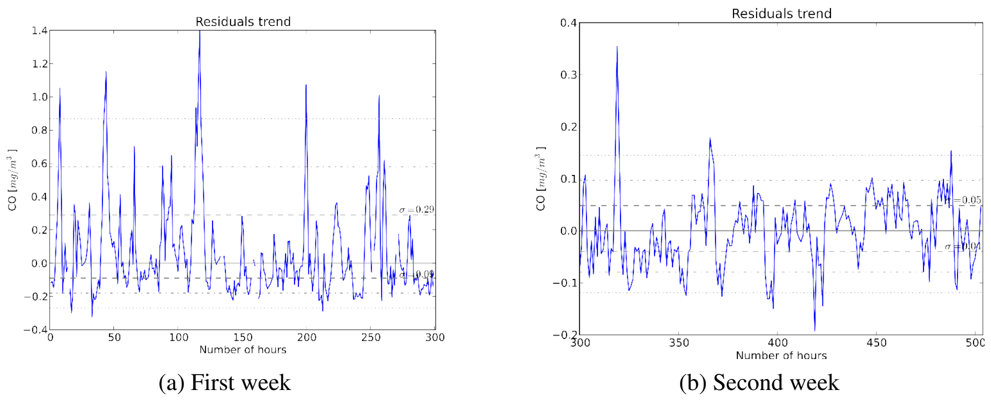


Figure 2.2: *CO* residuals between OMS measurements and low-cost sensor.

that became $\sim 0.05\,mg$ in the second week (see Figure 2.3).



(a) First week (b) Second week

Figure 2.3: Comparison of the residuals with removed drift for the first and second week of measurements.

2.4 Conclusions

The emerging technology of low-cost sensors can contribute to improving the characterization of spatial variability of air pollutants in a complex context (e.g. city centre). However, current technologies available on the market are designed to monitor the concentration of air pollutants in industrial and indoor environments. Further tests are required to prove the exploitability of these technologies for outdoor air quality monitoring. Low-cost sensors for NO_2, CO and O_3 can be used to provide qualitative information about air quality and to detect peaks, whereas low-cost sensors for PM_{10} are still not available (Rada et al., 2012).

The tests conducted in this work highlight a high cross-sensitivity of the low-cost sensor to other pollutants and micro-climate parameters (e.g humidity) and a low signal-to-noise ratio (S/N). Therefore, sensor network design and the post-processing phase requires accurate testing at local conditions.

As highlighted by Ragazzi et al. (2012), in spite of all the difficulties and challenges, low-cost sensors could lead to a new monitoring paradigm where private citizens can contribute toward integrating air quality monitoring systems. For example: enterprises could be interested in monitoring the air quality around plants to certify their activity to the community, while private citizens, grouping in committee, could support the air quality monitoring of their neighbourhood. Offering a higher-spatial resolution with respect to fixed air quality stations could be considered a useful tool in assessing population exposure near relevant pollutant sources (e.g. trafficked roads, domestic wood burning plants, filling stations placed in populated areas) located in regions with complex landforms or simply far away from traditional monitoring stations. This scenario, however, also raises some new issues that must be taken into account, such as: how can we rely on data collected from entrepreneurs or citizens? How can we validate or exclude these data? How can we homogenize information coming from different networks with heterogeneous sensors? How can we collect and organize these data coming from an increasing number of stakeholders and institutions?

To be able to address the above questions, the solution could be to mix and integrate different technologies such as: low-cost sensor networks, traditional air quality monitoring stations and protocols such as the Sensor Observation Service (SOS) defined by the Open Geospatial Consortium (OGC) to collect/filter and query all these data. An integrated system, that implements the above mentioned standards, promises to reach several advantages compared to traditional air quality monitoring systems and institutions: allows reaching higher spatial accuracy; reduces the redundancy of measures by different network systems; improves the localization of critical pollutant concentrations; reduces the cost of improving spatial resolution of the data; and allows the creation of a real time alert system for dangerous pollutants.

Chapter 3

PyGrass: a high level GIS library for research and rapid prototyping

PyGRASS is an object-oriented Python Application Programming Interface (API) for Geographic Resources Analysis Support System (GRASS) Geographic Information System (GIS), a powerful open source GIS widely used in academia, commercial settings and governmental agencies. We present the architecture of the PyGRASS library, covering interfaces to GRASS modules, vector and raster data, with a focus on the new capabilities that it provides to GRASS users and developers. Our design concept of the module interface allows the direct linking of inputs and outputs of GRASS modules to create process chains, including compatibility checks, process control and error handling. The module interface was designed to be easily extended to work with remote processing services (Web Processing Service (WPS), Web Service Definition Language (WSDL)/Simple Object Access Protocol (SOAP)). The new object-oriented Python programming API introduces an abstract layer that opens the possibility of using and accessing transparently the efficient raster and vector functions of GRASS that are implemented in C. The design goal was to provide an easy to use but powerful Python interface for users and developers who are not familiar with the programming language C and with the GRASS C API. We demonstrate the capabilities, scalability and performance of PyGRASS with several dedicated tests and benchmarks. We compare and discuss the results of the benchmarks with dedicated C implementations.

3.1 Introduction

Geographic Information Systems (GIS) have the capability to integrate heterogeneous digital data, giving the opportunity to public administration, industry and research to provide basic and advanced data analysis and modeling for a wide range of disciplines (Foody, 2008). The Geographic Resources Analysis Support System (GRASS) supports the creation, modification and processing of 2D and 3D raster and vector layers. It provides a topological vector model and true three dimensional coordinates for vector features analysis. GRASS is characterized by stability, an efficient application programming interface (API) written in C, and a large number of GIS functions and modules (Neteler et al., 2012). GRASS provides a large number of models and algorithms which, after substantial testing and trouble shooting, have proven to be very reliable. Its capabilities of processing geographical information have been attested by many research and technical papers (Ciolli et al., 2004; Hofierka and Zlocha, 2012; Li et al., 2010; Okabe et al., 2009; Preatoni et al., 2012; Sacchelli et al., 2014; Sacchelli et al., 2013; Tattoni et al., 2010, 2012; Vettorato and Zambelli, 2009; Vettorato et al., 2011; Zambelli et al., 2010, 2012).

GRASS GIS has a modular design. The core functionalities are implemented in shared libraries using the programming language C and can be accessed via the GRASS C API. This API provides read and write access to raster, 3D raster and vector data, as well as the handling of projection information, spatial and attribute database management, spline interpolation, mathematical and numerical functionalities and visualization functionalities (see Table 3.1). Spatial algorithms and models are implemented as small stand-alone programs, called modules, that make use of the C API. The implementation of GRASS modules follows the UNIX concept. Hence, each module in GRASS has a dedicated purpose and is efficiently implemented. Modules can be combined, similar to the UNIX tool concept. Since the early days of GRASS in the 80s, the UNIX shell was used to combine GRASS modules and UNIX tools to script repetitive tasks and to implement complex spatial analysis and processing algorithms. This concept results in a large amount of over 400 modules. Most of them are implemented in C. A sufficient amount is implemented as scripts using either POSIX (Portable Operating System Interface; defines a standard operating system interface and environment, including a command interpreter (or "shell"), and common utility programs to support applications portability at the source code level. Scripts are POSIX-based until version 6 of GRASS GIS or Python, as in the latest stable release of GRASS.

Many GIS software packages have chosen Python as a main language for users (see Table 3.2), because it is available on many platforms, and it seems to be a good compromise between simplicity (syntax, low learning curve), flexibility (multi-paradigm programming) and power (due to rich scientific libraries). GRASS developers have chosen Python to replace POSIX for scripting modules (Neteler et al., 2012). For this purpose, a Python scripting library was implemented by the GRASS development team, but it does not provide any further improvement to the POSIX approach than managing process chains using the standard Python library (`subprocess`).

ctypes	n. funcs	n. structs	n. vars
gis	501	20	175
raster	372	26	37
vector	344	66	147
dbmi	333	19	100
ogsf	331	33	151
raster3d	245	20	52
gmath	130	1	21
display	120	15	21
imagery	100	15	16
nviz	82	8	30
date	63	1	17
vedit	22	49	43
cluster	19	3	10
stats	19	0	10
proj	17	9	16
arraystats	0	1	10
	2,698	286	856

Table 3.1: GRASS C API consists of 2,698 C functions that are available through the ctypes library, divided into 10 different fields. This data is derived from the official GRASS source code (Zambelli et al., 2013).

Software	write	script	license	use	OS
ArcGIS	C++	python	proprietary	desktop	Windows
AutoCAD Map	C/C++	AutoLisp	proprietary	desktop	Windows
Geoserver	java	python(dev)	GPL	server	Windows, Mac, Linux
GRASS7	C	python	GPL	desktop	Windows, Mac, Linux
gvSIG	Java	jython	GPL	desktop	Windows, Mac, Linux
IDRISI	COM	python	proprietary	desktop	Windows
ILWIS	C++	python	GPL	desktop	Windows
Geomedia	C/C++	python	proprietary	desktop	Windows
MapInfo	C/Basic	MapBasic, python	proprietary	desktop	Windows
Mapserver	C/C++	python	X/MIT	server	Windows, Mac, Linux
QGIS	C++	python	GPL	desktop	Windows, Mac, Linux
Saga-GIS	C++	python	GPL	desktop	Windows, Mac, Linux
Udig	Java	groovy	LGPL	desktop	Windows, Mac, Linux

Table 3.2: Comparison of the most used GIS software (Zambelli et al., 2013).

Most GIS software and tools provide a large number of high-level algorithms to cover different GIS processing needs. Few GIS open capabilities to users to access the lower functionalities, such as iterating between the geometry features of a vector map, or iterating row by row to a raster map using a higher-level language.

GRASS modules must be implemented in C to access the low level functionality. To overcome this limitation and to reach a broader development community, a ctypes interface was introduced to GRASS version 7. This interface allows access to the low level GRASS C API in Python. However, the creation of new modules written in C or using the C API with Python through the ctypes interface is not a trivial task and is generally a very time-consuming activity. This happens because the writer must be a competent C programmer (manage the computer memory, work with pointers, *etc*.)

and because of how the GRASS library works internally. Hence, an intensive study of the large GRASS C API is required.

The goal of this work is to implement an intuitive and easy to use object-oriented layer around the GRASS C API, hiding its complexity, but providing a more abstract/powerful development environment for solving complex GIS data analysis and model problems. An additional task is the replacement of parts from the existing Python script API with more efficient and powerful object-oriented approaches. In this way, we can provide access to the capability of the C API of GRASS for power users and geo-scientists who are not familiar with C and the C API of GRASS.

The idea of PyGRASS was born from the experience of the authors who wished to expand GRASS capabilities by implementing a tool that gives the freedom to approach the GIS problem from a different perspective, opening a software developer's approach to GIS users and trying to maintain relative simplicity. The PyGRASS library provides a simple, object-oriented higher level interface that transforms each GRASS module into an object by interpreting its XML interface description, trying to simplify the syntax and enforcing the script activity. The object-oriented layer around the GRASS C API, PyGRASS, implements several classes to access vector and raster data, covering several complex features that are only available in the GRASS C API, like support for the vector topology or the use of the raster cache for fast random read and write access. In addition, PyGRASS simplifies interoperability with all related geospatial software and tools provided by a Python interface.

The development of PyGRASS has been sponsored by Google's Summer of Code program (2012), and the code produced has been integrated in the latest stable release of GRASS.

3.2 Methodology

The PyGRASS library is written in Python and makes use of modules from the Python standard library (van Rossum, 1995), like: `sys`, `fnmatch`, `collections`, `sqlite3`, as well as from the third party Python library, NumPy (Jones et al., 2013). NumPy is a package for scientific computing and it is already a dependence of GRASS. An optional library is `psycopg2` (Varrazzo and Psycopg Community, 2013), which is used to interface PyGRASS vector attribute handling with the PostgreSQL database.

The PyGRASS library was developed taking into account four main aspects:

- *consistency* – the library shall adhere to norms and architecture commonly found in both Python and GRASS, in order to avoid confusion for users who are only familiar with one of the above;
- *simplicity* – the library must be simple and intuitive, without hiding access to

lower-level functionality, indeed, providing a seamless user experience between the low level C API of GRASS with a higher-level object-oriented Python approach;

- *flexibility* – the library must be flexible, both allowing the use of existing GRASS modules and giving to each user the freedom to implement his own logic, using more detailed and fine-grained programming tools;
- *performance* – the library must be fast, considering both the development and the CPU time. GRASS C API functions are heavily used by *PyGRASS* every time that it is possible.

The library is split in two parts: the first is more related to script activity and the GRASS modules; the second is focused on programming aspects and the C API of GRASS.

To improve the existing script API of GRASS, PyGRASS considers each GRASS module as an object with input parameters, output parameters and flags. When the object is "instantiate", the `Module` class parses the XML interface description generated from the GRASS modules through the `--interface-description` flag to know which parameters and flags are defined. For each parameter, the metadata is analyzed. The metadata specifies if a parameter is required or optional, if it is an input or an output, what type it is (raster, vector, string, float, *etc.*) and many more. This information allows the class to check the correctness of the parameters and provides the capability to suggest the correct ones. The identification of inputs and outputs allows the implementation of process chains. The interface design of this class was chosen to support the implementation of local and remote process execution services, which may be added in future. To implement an interface to a Web Processing Service (WPS), the definition of complex inputs, complex outputs and literals must be known to generate the XML execute request. The same is true for remote process execution services based on WSDL/SOAP. The `Module` class provides all required module-specific information by design.

The current Python script API defines several functions to manage the GRASS module: the `make_command` returns a list of strings with the command options from a dictionary of keys and values. The `start_command` is a GRASS-oriented interface to `subprocess.Popen` (a module process creation and management provided by the Python standard library), that internally uses the `make_command` function. All the other `run/pipe/feed/read/write/parse` command functions are specialized wrappers of the `start_command`.

The `Module` class of PyGRASS gathers all these features in a single object, connecting directly the inputs and outputs of GRASS modules; see Listing 8.1 in the

Appendix.

The PyGRASS `Module` class simplifies the Python syntax as much as possible to be competitive with the POSIX module interface. It supports backward compatibility syntax and enhances the API to provide a tool that manages user errors and returns a list of valid options. Moreover, the PyGRASS library gives the capability to pass text to a command as input (*stdin*), to catch the text output (*stdout*) and the error message (*stderr*) of a command. Finally, PyGRASS allows users to manage (*i.e.*, terminate, kill, or wait) the process.

The PyGRASS library introduces an Object-Oriented (OO) Python API to GRASS, which implements for each GIS/GRASS entity one or more classes. The classes use C structures and functions through the existing *ctypes* interface. *Ctypes* is a Python library; it provides C-compatible data types and allows calling functions in DLLs or shared libraries. It can be used to wrap these libraries in pure Python. Our higher level Python interface uses *ctypes* to integrate the underlying GRASS C API structures and functions in an object-oriented framework, but at the same time, trying to respect the GRASS work-flow and nomenclature to conform with the C API. The object-specific *ctypes* pointer to the underlying C structures are available under the attribute name that starts with `c_*`. This allows the user to access the lower level GRASS C API structures directly using the *ctypes* interface. These classes allow facing the problem to be confronted in a more abstract way. A high-level object-oriented approach can help users face the problem, even if they are not familiar with the implementation details of the C API level, speeding up the design, writing, prototyping and debugging phases.

Architecture of the Library

The PyGRASS library follows the main GRASS structure and is divided in four parts. Each part implements a set of dedicated classes. See Figures 3.1 and 3.2 for a general overview of the library.

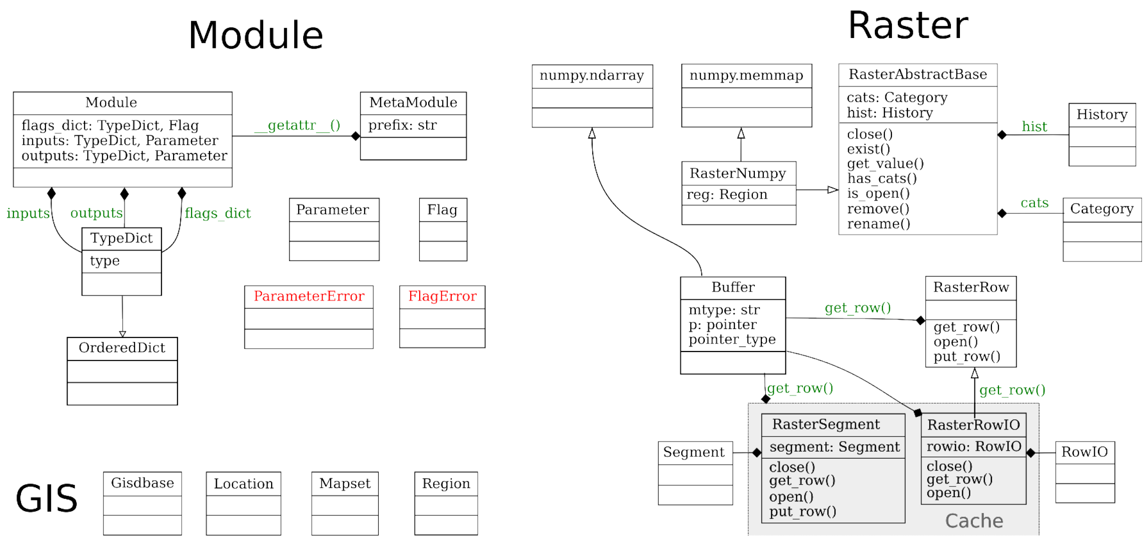


Figure 3.1: Module, Raster and GIS classes.

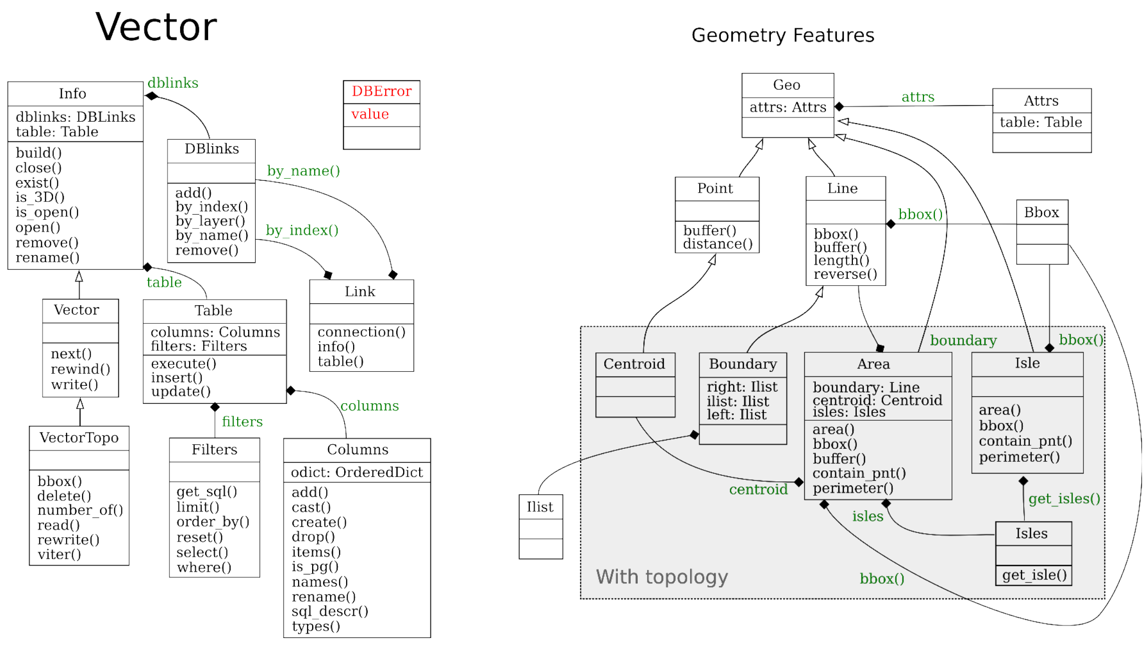


Figure 3.2: Vector class.

- `modules` contains the classes `Module`, `MetaModule` and `Parameter`. These classes are designed to substitute the previous POSIX-based scripting approach (see Listing 8.2) and replace parts of the existing Python script API (see Listing 8.3).

The object-oriented architecture of the PyGRASS library allows users to interact with GRASS modules as Python objects. These objects allow direct access to module attributes like: `name`, `description` and `keywords`. The `input` and `output` options are implemented using a dedicated parameter class. Instances of this class are stored either in an input dictionary or an output dictionary within the module object (Listing 8.4). Inputs and outputs can be referenced by their name in the dictionaries or as attributes of the dictionary objects. They can be connected to each other to create process chains, (see Listing 8.1). The type check system of the parameter class assures that output options can only be connected with input options of different modules when they have the same type. Hence, the PyGRASS module library will raise an error in case the output of a vector module was connected with a raster input of a second module.

The PyGRASS module library introduces special parameters to allow fine-grain control over the GRASS processes. These special parameters end with the '_' character to avoid a mix-up with option names. The first two special parameters – `run_` and `finish_` – are used to manage the process. The parameter definition `run_=True` will execute the process immediately, and `finish_=True` will wait until the process terminates, (see Listing 8.5). Other special parameters that were added are: `stdin_`, `stdout_` and `stderr_`. The parameters `stdin_` and `stdout_` are used to connect the textual inputs and outputs from different modules to create a process pipeline; `stdin_` is used to pass the textual output (`stdout_`) from one process to another, (see Listing 8.6).

- `vector` contains the classes `Vector` without the GRASS topology and `VectorTopo` with the GRASS topology.

  The `Vector` class allows the user to access the non-topological geometry features of a vector map in sequential order; see Listing 8.7. The class `VectorTopo` was designed to access topological and non-topological geometry features of a vector map in random order. This class allows for iterating among specific feature types, (see Listings 8.8 and 8.9). Writing is supported in both classes in sequential order. However, already written features can be updated in the topological access class.

  The following classes are designed to represent vector features: `Point`, `Line`, `Centroid`, `Boundary`, `Isle` and `Area`. Instances of these classes are usually created when features are read from vector maps by the `Vector` and `VectorTopo` classes. To manage multiple connections with vector attribute SQL databases, the classes `DBLinks` and `Link` were designed. Attribute tables can be created, accessed and modified with the `Table` class. The `Filter` class provides several methods for working with data without the need to know SQL. The `Attrs` class was designed to access the content of the attributes table from a geometry feature.

- `raster` contains the classes (`RasterRow`, `RasterRowIO`, and `RasterSegment`).

  Each class uses a different GRASS C-library to grant a specific kind of access to raster maps. All the raster classes share common methods to open a map, read raster values or raster rows, get raster information and write metadata, such as categories and history. As with vectors, a similar syntax has been used to instantiate, open and close a raster object.

  The `RasterRow` class reads the contents of the raster map row by row and writes it in a sequential mode, row after row, (see Listings 8.10 and 8.11).

  The `RasterRowIO` class implements a row cache that allows users to read raster rows randomly by keeping a number of rows in the main memory. This caching mechanism avoids heavy I/O (input/output) hard-disk usage in specific tasks, such as moving window operations, or cell neighborhood analysis. Similarly to the `RasterRowIO`, the `RasterSegment` class provides access to a tile cache. The tile cache is an uncompressed representation of a raster map that will be created at the point of initialization. The access to the uncompressed file is based on tiles that are cached in the main memory for fast random read and write access through the `Segment` class. With the `RasterSegment` class, it is possible to read and write the pixel value randomly at the same time in the same map.

- `gis` contains GRASS management classes, like `Gisdbase`, `Location` and `Mapset`, that help users interact with the GRASS environment, (see Listing 8.12). The `Region` class manages the computational region of GRASS that directly affects 2D and 3D raster processing, as well as several vector processing algorithms, (see Listing 8.13).

The PyGRASS library assures that memory management is fully handled by Python. All structures from the GRASS C API that are used by PyGRASS are *ctypes* objects or get deleted in the class destructor's and, therefore, are handled by the Python garbage collector. The user must not take care of memory allocation and deletion directly.

3.3 Results

In this section, we compare different solutions of simple GIS tasks using standard GRASS tools and PyGRASS. The machine used for the benchmark was a laptop with an Intel Core i7 3610QM processor with 2.30 GHz and 6 Mb L3 Cache. The system has 24 Gb DDR3@1333Mhz of RAM and a solid state disk (SSD) of 250 Gb as the system driver. The installed operating system (OS) is GNU/Linux 3.7.5 ($\times$86_64) on the SSD. The GRASS 7 development version used for the benchmark has the revision number `r54812`. The GRASS data are stored on a secondary hard disk of 750 Gb at 7,200 rpm.

Concerning script activity, PyGRASS improves mainly the syntax and changes how users can interact with GRASS modules. We measured small performance loss when executing GRASS modules using the PyGRASS module interface compared to the POSIX approach, which go from 1% up to 12%, due to the average load of the system. We did not expect a large performance difference, since Python and POSIX are basically using the same OS (operating system) functions to spawn processes.

On the contrary, we needed to test the new API added by PyGRASS to identify its strengths, weaknesses and scalability (all the benchmark tests used in this chapter are available at https://github.com/zarch/pygrass-benchmark). Each test, excluding the biggest region (with 10^{10} cells), has been repeated five times. There are only small differences between each measured run time, resulting in a small standard deviation. Hence, we think that the final results of our benchmark are representative.

The first test compares two simple procedures, one written using PyGRASS (`RasterRow` and `VectorTopo`) (see Listing 8.14) and the other using the programming language C (see Listing 8.15 and for the results, see Table 3.3). The test takes as inputs a vector point map and a raster map. It creates a new vector point map that includes all vector points from the input map. A new attribute table is created and linked with the vector map, which contains a column with the sampled values of the raster map. The procedure is applied to five different random vector point maps, to be independent from the spatial distribution of the vector points. Moreover, the tests have been executed using different region extents and numbers of points, to test the scalability of the different solutions. Both procedures are conceptually identical and share most of the GRASS C API functions. The only difference is database access: PyGRASS uses the Python driver instead of the C API of GRASS.

number of cells	10^2	10^4	10^6	10^8	10^{10}
number of points	10	10^2	10^3	10^4	10^5
Vector and Raster					
sample (PyGRASS)	2.21	4.23	23.87	218.63	12670.27
v.sample2 (C API)	3.03	5.48	31.67	266.54	13304.67
Raster					
RasterRow	0.046	0.431	4.53	74.46	4303.24
r.mapcalc	0.078	0.525	5.83	170.43	5347.95

Table 3.3: Table with benchmark results, reported in seconds, using different computational extents and a different number of vector points (Zambelli et al., 2013).

One indication that our approach is easier to handle than the C implementation is that the PyGRASS version is considerably shorter (48 lines) than the C version (102 lines). With a PyGRASS library, it is noted that there is a marginal advantage in speed compared to it's C counterpart. The speed gain over the C version is probably due to the slower driver adopted by the GRASS C API of the vector attribute database.

In a further test we compared performance of the PyGRASS `RasterRow` implementation (see Listing 8.16), with `r.mapcalc` using a simple raster map algorithm (see Listing 8.17). The algorithm stores only those pixels in a new raster map that have a value that it is greater than 50. Again, the PyGRASS version is slightly faster than the GRASS module. The good performance of PyGRASS is caused by our design approach that uses NumPy for row computation tasks. The performance will drop dramatically in case we would implement the same algorithm in pure Python comparing cell by cell without using the optimized NumPy approach.

3.4 Discussion and Benchmarks

The PyGRASS `Module` class adds some useful features that were not available with the previous Python script API; these features have a time cost, because they require exporting the GRASS module in XML, parsing the XML and instantiating the object, checking that all parameters are correct and then executing. The time cost for these operations is around 0.2 s, but generally, the execution time of a GRASS module requires much more time; therefore, in most of the cases, we can neglect this time loss.

Concerning the new approach introduced by PyGRASS, the performance depends mainly on the features that are used. For example, updating the column attribute with the value of area with PyGRASS requires almost the same time, around 0.24 s for PyGRASS and 0.26 s using the `v.to.db` module.

Using the `RasterRow` class to compute areas that satisfy a condition, with a region of 16,000 rows and 14,000 columns, it is slightly faster (27.42 s) than using `r.mapcalc` (35.49 s) if the row is used as a NumPY array. Using the PyGRASS `RasterRow` class without using the NumPy array makes the execution seven-times slower than using the GRASS `r.mapcalc` module (992.5 s vs 144.2 s).

The example above highlights that it is not convenient to replace an existing GRASS module with a new one written in PyGRASS, because the user has to write more code and because the GRASS native modules are generally faster. The big advantage of using the PyGRASS library is the object-oriented access to the GRASS C API functionality.

Without the need to extract information from the output string of the module, in this kind of operation, the PyGRASS library is faster compared with modules and with existing Python functions: for example, to get the list of the raster map contained in a `Mapset` with PyGRASS takes 608 ns.

Using the Python function `list_grouped` in the GRASS core takes 0.1273 s.

The same good results are obtained with the `Region` object; with PyGRASS, it takes 211 ns.

Using the Python function `region` in the GRASS core takes 0.1056 s.

The PyGRASS library can help to substitute all the commands in the GRASS Python `script` library that need to wrap and interpret the output of a GRASS module.

3.5 Conclusions

An increasing amount of GIS software uses the Python language to provide a powerful scripting interface. An easy to use, but powerful, Python interface can help to efficiently exploit the capabilities of GIS software. Such an interface can be effectively used to integrate different GIS, statistical, and geospatial tools and programming languages in a GIS to expand its overall capabilities.

The PyGRASS library tries to open a new perspective to power users and scientists that use GRASS GIS. It provides a Python interface that is able to compete with the simplicity of POSIX to write procedures with existing GRASS modules, as well as a powerful object-oriented interface to deal and experiment with GIS problems at a lower level.

The new `Module` class, introduced by PyGRASS, provides a single interface to all GRASS modules and can be extended to work with Web Processing Services (WPS), Web Services Description Language (WSDL) and Simple Object Access Protocol (SOAP) services or other remote execution services. The design concept of the `Module` class allows direct linking of inputs and outputs of GRASS modules to create process chains, including compatibility checks, process control and error handling.

The new Object-Oriented Python programming API introduces an abstract layer that opens the possibility for users who are not familiar with C and with GRASS C API, to use and access transparently the efficient C functions of GRASS. Our tests show that algorithms implemented with PyGRASS are comparable in terms of performance with an equivalent C implementation. Hence, our approach wraps the underlying GRASS C libraries efficiently. It needs much fewer lines of code to implement an algorithm in PyGRASS than in C. Moreover, it shows that specific Python strengths, for example, the database Python interface, can be used to gain a speed improvement over specific C-implementations in GRASS. The PyGRASS library has been designed to integrate new methods or to inherit from an existing class to extend GRASS functionalities, providing new tools for prototyping complex scientific algorithms.

Some of the functionalities provided by PyGRASS are also available in other software, such as Postgresql/Postgis, R, shapely, *etc*. However, switching to them requires changing the GIS working environment. That means installing, configuring, learning the new tools and converting from one format to another. The PyGRASS library does not force the users to learn and switch between different languages (C, SQL, R, Python, BASH, *etc*.) and tools to carry out their work.

The PyGRASS library allows GIS modelers and scientists to use the C API of GRASS, with a high level interface, providing a tool that gives the freedom to approach the GIS problem from a different perspective. In this way, users and scientists can combine the GRASS modules with the GRASS C API functions and algorithms. Therefore, PyGRASS is able to simplify the approach to develop a new GIS model, using one program (GRASS) and one language (Python) to cover the different GIS aspects, increasing the productivity and allowing geo-scientists to focus on studying the problem they have selected and not on studying the tools and languages used.

Moreover, the PyGRASS library can be used as a tool to facilitate use and integration with other GIS/statistical software and libraries (not only open source). The common language among different software and the object-oriented structure should make communication and procedure/data exchange easier.

The PyGRASS library, together with the GRASS GIS temporal framework, can provide a comprehensive high performance spatio-temporal GIS framework for GI-Scientists.

PyGRASS seems to be ideal applying in complex case studies, such as air quality monitoring from wireless sensor networks, and for building decision support systems to evaluate the assessment of sustainable forest energy.

Chapter 4

Machine learning classification of urban micro-environments

Urban micro-environments substantially affect the spatial variability and distribution of air pollutants, urban heat-islands (UHI) and urban micro-climate. A fine resolution land classification and urban morphology can help to address these topics and provide useful insights into this spatial variability for urban planners and local authorities. This work presents a new methodology to identify the urban micro-environments from colour airborne images (RGB) applying automatic algorithms in an open source software environment. A set of several machine-learning algorithms was used to obtain an object-based image supervised classification. Eight main categories (water, grass, vegetation, road, car, bus, building and shadow) were classified. Overall, 9,301 areas were used for training and evaluating the machine-learning results. The cross-validation accuracy of the six best algorithms is above 93% of the training data set using 5 folds. The best classifiers were Gradient Boost (93.82%), Random Trees (93.75%) and Extremely Randomized Tree classifier (93.73%).The described methodology yields reliable results for the classification of urban micro-environments using only RGB images and constitutes a promising approach. Where available, spectral data could further improve the results characterizing each object with additional information about the surface that can be used by the classifiers.

4.1 Introduction

A highly detailed spatial characterization of urban environments is of increasing interest for several research fields and communities. Fine resolution land classification and urban morphology can help address questions regarding the spatial variability of air pollutants (Merbitz et al., 2012), help to understand model phenomena like the Urban Heat Island (Busato et al., 2013), and urban micro-climates (Rahman et al., 2014), and also to identify the specific role of ecological functions and different ecosystem services (Behling et al., 2015).

Lehmann et al. (2014) stressed the importance of identifying the urban vegetation structure type (UVST) to link particular ecosystem services with physical parameters to explain temperature variability at the urban and district level. Jin et al. (2014) showed how the concentration of fine particulate matter ($PM_{2.5}$, particulate matter with an aerodynamic diameter $< 2.5\,\mu m$) is influenced, for instance, by the canopy density (CD) and the leaf area index (LAI). Other studies such as (Kaur et al., 2007; Pirjola et al., 2012; Rahman et al., 2014) demonstrated that the characterization of micro-climatic and urban micro-environment conditions also drive concentrations of various air pollutants.

With the term "urban micro-environments", we aim to identify not only environments with similar micro-climate conditions but also with a similar context such as: urban morphology, distance from main roads, and other human structures and activities. The characterization of urban micro-environments, especially if combined with indoor micro-environments (Steinle et al., 2013), can help to assess the overall impact of air pollutants on the population and to define new policies to reduce this impact (Rada, 2014; Rada et al., 2012; Schiavon et al., 2014).

We selected eight broad land cover classes, that are considered representative, to characterize urban environments (Chen et al., 2014; Dugord et al., 2014; Kotthaus and Grimmond, 2013; Lehmann et al., 2014; Llop et al., 2012). Then we extracted them from aerial images at high spatial resolution (0.25 m). The spatial characterization of these classes combined with other information, such as digital terrain and surface model, can provide useful information to assess the spatial variability of physical phenomena that occur at the urban scale (Liu and Shen, 2014; Mölter et al., 2010; Skelhorn et al., 2014). The classification of an image is a complex task due to the high number of elements (e.g., shapes, materials, textures, and relative position between the objects) that need to be interpreted. Recent publications suggest that an object-based image classification can achieve better results when combined with machine-learning algorithms, as compared to per-pixel classifications (Duro et al., 2012).

The Geographic Objects-Based Image Analysis (GEOBIA) is a process that splits image into segments (objects) based on analysis of the statistics of shape, texture and spectral response. A growing number of publications describe combining object-based image classification with machine-learning algorithms (Alioscha-Perez and Sahli, 2014; Clewley et al., 2014; Dronova et al., 2012; Duro et al., 2012; Novack et al., 2011; Senthilnath et al., 2012; Wieland and Pittore, 2014). The most employed classifiers are the Support Vector Machine (SVM), Random Forest (RF), Decision Tree (DT), and k-Nearest Neighbours (k-NN). All of these studies apply the GEOBIA to multi-

and hyper-spectral images.

The objective of this study was to develop a methodology along with an implementation as open-source tools for the characterization of urban micro-environments using only orthophoto images with the three spectral bands of red, green and blue (RGB). We chose to work with RGB images because they are generally available at lower cost compared to other multi-spectral data and are commonly available in aerial photography and in high resolution satellite imagery. An open-source tool is important as a guarantee of repeatability and reproducibility of the research (Steiniger and Hay, 2009).

The study areas in the City of Edinburgh, Scotland (UK), were chosen to represent a medium-size city of Northern Europe. With respect to other areas, it is particularly challenging because the roof colors are often similar to the color of nearby roads. Moreover, the study area is characterized by huge varieties of urban settlements that include streets and medieval alleys and buildings ranging from residential and sub-urban areas and modern and industrial buildings.

In this chapter, I present the use of ten different machine-learning algorithms. For each algorithm, I tested several set-ups to determine the options that perform best with our training dataset.

As reported by Clewley et al. (2014) there are currently a number of open source packages that can be used to perform various parts of the GEOBIA process (Orfeo Toolbox (Inglada and Christophe, 2009), TWOPAC (Huth et al., 2012), InterIMAGE (InterImage, 2014), GDAL-RSGISLib-RIOS-TuiView-KEA (Clewley et al., 2014)). None of them is integrated with a multi-purpose GIS software such as GRASS. The presented work aims to fill this gap opening the GEOBIA process to a wider number of users.

4.2 Material and methods

4.2.1 Study area and data

The study area covers most of the city center of Edinburgh, Scotland (UK), with a dimension of 6 km x 12 km (Figure 4.1). Edinburgh is the capital of Scotland and it is situated on the south shore of the Firth of Forth. It is the seventh most populated city in the United Kingdom and the second largest urban area in Scotland with a total population of 487,000 habitants in 2013 (National Records of Scotland, 2013).

The input dataset for the image classification task of the urban micro-environment of the city of Edinburgh is a set of RGB aerial ortho-rectified images (2009) provided by the Ordinary Survey (British Ordnance Survey, 2012) with a spatial resolution of 0.25 m by 0.25 m.

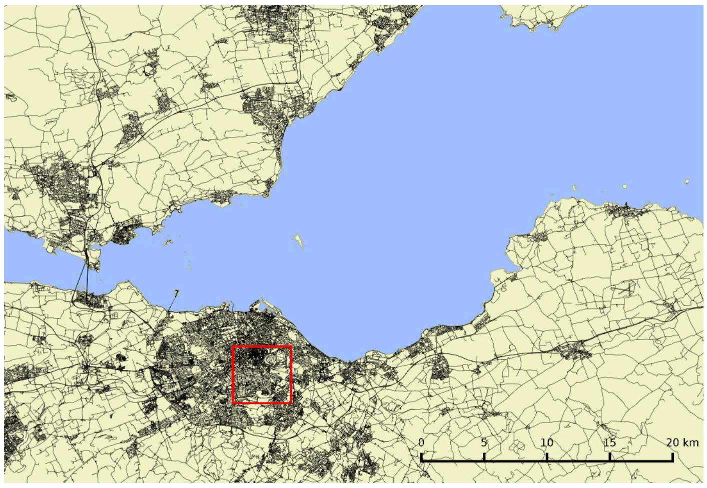


Figure 4.1: The red box illustrates the location and dimensions of the study area in the City of Edinburgh, Scotland/UK.

4.2.2 Tools

I used `GRASS` (Geographic Resources Analysis Support System), a free and open source Geographic Information System (GIS) (Neteler et al., 2012) to process geographical data through the `pyGRASS` library, and `scikit-learn`, which provided the main machine-learning algorithms for the classification of the objects. `Python` (van Rossum, 1995) was used as the main language to integrate all the tools together and to develop a new GRASS GIS 7 module `v.class.ml`.

- The GRASS GIS software suite has been under continuous development since 1982 (Neteler et al., 2012). Since 1999, it has been developed as free and open source software by an international development team. GRASS GIS has been used in several research projects, e.g., to develop holistic models to assess the biomass potential in alpine regions (Sacchelli et al., 2013; Vettorato et al., 2011; Zambelli et al., 2012).

- `pyGRASS` is a Python interface used to access the C API of GRASS GIS with a higher and more abstract interface (Zambelli et al., 2013) and was developed during the Google Summer of Code 2012 (GSoC 2012) as a tool to simplify the use of GRASS GIS from other Python libraries. The main features of `pyGRASS` are described in Chapter 3.

- Scikit-learn (Pedregosa et al., 2011) is a Python library that provides simple and efficient tools for data mining and data analysis built on NumPy (Walt et al., 2011), SciPy (Jones et al., 2001-2015; Millman and Aivazis, 2011; Oliphant, 2007), and matplotlib (Hunter, 2007). The main features of Scikit-learn are algorithms for classification (Support Vector Machine (SVM), nearest neighbors, and random forest), regression (Support Vector Regression (SVR), ridge regression, and Lasso), clustering (k-Means, spectral clustering, and mean-shift), dimensionality reduction (Principal Component Analysis (PCA), Isometric mapping, and non-negative matrix factorization), model selection (grid search, cross validation, and metrics), pre-processing and feature extraction.

- Python is a general purpose high-level programming language that supports multiple paradigms (e.g., object oriented, imperative, functional and procedural styles). It is an interpreted language with a dynamic type system and automatic memory management and is provided with a large and comprehensive standard library (van Rossum, 1995; Wikipedia, 2014b).

4.2.3 Methodology

For the classification methodology presented here, we used an RGB image of the city center of Edinburgh for the following eight broad classes: water, grass, tree, road, car, bus, building and shadow. The shadow class was introduced to avoid forcing the classifiers to assess the land-use of areas without enough information. We introduced the car and bus classes because, in many applications, it could be interesting to assess the traffic density on roads using relative low-cost RGB images. The remaining classes are basic land-use classifications that are generally associated with a certain material and use, which can help to assess the urban micro-climatic and/or environment's conditions.

I classified the RGB image set using five main steps: bands ratio, image segmentation, segment conversion, feature extraction, and classification. In detail, I processed the data as follows:

1. Bands ratio: as highlight by Gitelson and Merzlyak (1996) and Zhang and Hu (2012) to improve the identification of vegetation in a urban environment and to be less sensitive to the partial shadowing due to the closer vegetation and buildings, they combined the $NDVI$ defined as:

$$NDVI = \frac{NIR - R}{NIR + R} \tag{4.1}$$

   where NIR and R represent respectively the Near-Infrared and the Red image bands, with the Greeness Index (GI) expressed as:

$$GI = \frac{G}{R + G + B} \tag{4.2}$$

where R, G, B represent respectively the Red, Green and Blue image bands. Since the NIR band is not available on our dataset, we used the GI, which is an index representing the green percentage of the pixel. Zhang and Hu (2012) demonstrates that another useful band ratio that helps to distinguish between deciduous and conifer is GRI, defined as:

$$GRI = \frac{G-R}{G+R} \tag{4.3}$$

Following the same path, the indexes: GI, RI, BI, GRI GBI, RBI, visible in Figure 4.2, were computed and scaled to a value between $1-255$ to improve the results of the segmentation algorithm and help to identify and distinguish the image objects.

2. Image segmentation: The image segmentation was performed using a growing and merging region algorithm with the `i.segment` GRASS GIS 7 module. During the segmentation process, a unique segment ID was assigned to the pixel if the difference of the pixels similarity criteria was lower than the user defined threshold. The image was segmented hierarchically, which means that the output of the segmentation was used as input seeds for further segmentations. The image was segmented using a threshold of 0.01; the output was used as an input for the segment with a threshold of 0.02, and then for 0.03, etc. The thresholds computed were: 0.01, 0.02, 0.03, 0.04, 0.05, and 0.06. Another parameter used for the segmentation was to set a minimum number of pixels that a segment must have to be identified with a unique ID. I selected the best segmentation parameters from a visual comparison of the ones identified and objects that were fused together. Figure 4.3 shows the results of segmentation using different parameters.

3. Conversion of segmented raster data to a vector map: The segment map was converted from an integer raster map containing the ID of each segment, to a vector map using the `v.to.rast` GRASS GIS module. In GRASS GIS, each geometry feature of a vector map can be linked with one or more categories of an attribute table called a “layer”. During the classification process, I linked each geometric feature of the vector map with several layers: one containing the features extracted for each segment, one with the training classes, and one with the final classification results. I used `v.category` to assure that all of the geometric features are linked with the same category in all of the vector layers, otherwise, the first row in the attribute table with the extracted features and the first row with the classification results could not be referred to the same geometric feature (segment).

4. Segment characterization and features extraction: The segment characterization was performed by extracting several statistical parameters from the raster image for each band and feature shape, using `v.stats`. To improve characterization of the segment, I computed the Principal Component Analysis (PCA) of the RGB image (`i.pca`) and the first component of the PCA is used to analyse the texture index (`r.texture`). The `r.texture` module computed the first (see Figure 4.4) and the second order statistics (see Figure 4.5). The texture indexes considered on this work are six, with a moving window of three pixels: Entropy (ENT), Sum of Variance (SV), Sum of Average (SA), Angular Second Moment (ASM), Contrast (CON), and Measure of Correlation (MOC). The shape of each segment was characterized by extracting 13 parameters: the number of isles, the longitude and latitude extension, the perimeter, the area, the area of the boundary of the segment without considering internal isles, the ratio between the area of isles and the area of the boundary, the *compactness factor* and the *fractal dimension* defined as:

$$compactness\,factor = \frac{perimeter}{2\cdot\sqrt{\pi\cdot area}} \tag{4.4}$$

$$fractal\,dimension = 2\cdot\frac{log(perimeter)}{log(area)} \tag{4.5}$$

   The isles of segments were characterized by the same parameters: area, perimeter, compactness factor and fractal dimension and aggregate using the sum.

   From each raster (R, G, B, PCA1, PCA2, and PCA3) and for each segment, I extracted the following 20 zonal statistics: the number of pixels, the coefficient of variance, coefficient variance of the square, Kurtosis, Kurtosis of the square, maximum, mean, median, minimum, mode, number of occurrences, percentile (90%), range, skewedness, skewedness of the square, standard deviation, standard deviation of the square, first quartile, third quartile, variance and variance of the square. In summary, I used 15 bands to characterize each segment (3 RGB + 6 RGB indexes + 6 textures); for each band I extracted 20 parameters with zonal statistics for each segment, and the shape of the segment was characterize by 7 values. Therefore the number of features extracted for each segment was 307. All these statistics were grouped together into a new layer of the vector map.

5. Segment classification: The new module `v.class.ml`, developed during the third year of the Ph.D., implements several utilities: to perform pre-processing, to tune classification parameters and to conduct classification and post-processing

tasks. The module can be applied to any general vector map that has all of the attributes that are integer and/or float types, and there are no limitations on the number of columns (e.g., features) and rows (e.g., segments) except the dimension of the whole table that must be able to be allocated to the RAM of the computer used for the analysis. The classification process can be split into three main tasks: extraction of the training segments, testing of the different algorithms and set-ups, and finally, classification of all of the remaining segments.

The `v.class.ml` requires a vector map where the training areas for supervised classification are identified. For each supervised area, the module extracts all of the segments that are included and/or intersected and assigns the class of the area to them. All of the segments selected from the training vector map are saved in a new layer of the vector segment map. To train and test the different machine-learning algorithms, I manually classified more than 9,300 segments; a complete list of classes with the number of training segments per class is reported in Table 4.1

class	id	numb. of training segments	[%]
`water`	0	297	3.19
`grass`	1	2,810	30.21
`vegetation`	2	1,260	13.55
`road`	3	1,121	12.05
`car`	4	126	1.35
`bus`	5	94	1.01
`building`	6	3,279	35.25
`shadow`	7	314	3.38
total training		**9,301**	**1.34**
total		**691,795**	

Table 4.1: Table with the class name, numeric label and number of segments used to train all the different machine-learning algorithms

To identify the machine-learning algorithm that provides the best result, I used a cross-validation with a k-fold of five sub-samples. The sub-samples were generated by randomly shuffling the dataset samples but preserving the percentage of samples for each class. The machine-learning algorithm uses all of the sub-samples for training, excluding one that is used to compute the accuracy score as defined in Table 4.2, and the process is cycled through all of the sub-samples. Using this method, all sub-samples are used both to train and to test the machine-learning algorithm. The selection of the best algorithm is conducted by computing the average of five accurately computed during the cross validation process and selecting the algorithms with the higher mean value.

I tested the following ten machine-learning algorithms: the Gradient Boosting

Classifier (GBC) using 100 Decision tree estimators with a varying minleaf size of one, three, and five; the Support Vector Classifier (SVC) using different kernels: linear, Radial Basis Function (RBF), Sigmoid, and Polynomial and varying the C (from 10-2 up to 108) and the γ values (from 10-6 up to 106); Extra Trees Classifier (ET) with a different number of estimators (10 and 100) and with different minimum leaf sizes of one, three, and five; the Random Forest (RF) using different criteria (Gini and Entropy) and with a different value for the maximum feature (a percentage, the sqrt, and a log2); the k-Nearest Neighbor (kNN) using a different number of neighbors (2, 4, 8, 16) and different weights (uniform and distance); Nearest Centroid (NC) using different metrics (l1, l2, cityblock, manhattan, and eucldean); the Decision Tree (DT) using different criterion (Gini and Entropy); Stochastic Gradient Descendant (SGD) varying the loss parameter (hinge, huber, and log) and different penalties (l1, l2, and elasticnet); Gaussian Naive Bayes (GNB), and Ada Boosting Classifier (ABC) using 50 estimators with a learning rate of 1 and a minimum leaf of 3.

Because the Support Vector Classifier requires 2420 models ($4(kernel) \cdot 11(C) \cdot 11(\gamma) \cdot 5(cross - validation)$) to run, the training dataset were applied on a balanced sub-set of the training classes, with a number of segments for each class that is equal to the class with the least training segments. The class with the lowest number of training segments is the bus with only 94 segments. Therefore, the subset was composed of 94 segments per class, reducing the total number of the training dataset from more than 9300 to ($94 \cdot 8 =$) 752 segments. To select the number of trainings for each class that provides the best results, an instance of the SVC using the default parameters (kernel=RBF, $C = 1$, $\gamma = 0$) was tested 1000 times with 1000 different randomly balanced shuffled training datasets; the subset of the training segments that provided the highest accuracy was used to explore the domain of the best kernel, C and γ set of values that maximize the accuracy score.

Some of the above machine-learning algorithms tested during this work were ensemble methods. Ensemble methods, to improve robustness and generalizability of the classifier, combine the results of several estimators. The first approach is to return the average of several classifiers; this approach compared to a single estimator has a reduced variance (Pedregosa et al., 2011). RF and ET classifiers are the ones that use this approach. A second approach is to combine and organize several (weak) classifiers in a sequence to obtain a powerful ensemble (Pedregosa et al., 2011); the classifiers tested as part of this group are ABC and the GBC.

The classification task was repeated with a different configuration: using a scaled and decomposed (PCA) version of different features: RGB, PCA, and RGB and PCA statistics.

Based on the overall accuracy, the best 6 algorithms and pre-processing options were further analyzed by computing the confusion matrix to see which classes can be more problematic than others and by analyzing the Precision/Recall curve as defined in Table 4.2.

	Condition				
	$T_{pop} = CP + CN$ Total Population	$CP = TP + FN$ Condition Positive	$CN = FP + TN$ Condition Negative	$PRV = \frac{CP}{CN}$ Prevalence	
Outcomes	$OP = TP + FP$ Outcomes Positive	TP True Positive	FP False Positive	$PPV = \frac{TP}{OP}$ Positive Predictive Value (Precision)	$FDR = \frac{FP}{OP}$ False Discovery Rate
	$ON = FN + TN$ Outcomes Negative	FN False Negative	TN True Negative	$FOR = \frac{FN}{ON}$ False Omission Rate	$NPV = \frac{TN}{ON}$ Negative Predictive Value
	$LR_{+} = \frac{tpr}{fpr}$ Positive Likelihood Ratio	$TPR = \frac{TP}{CP}$ True Positive Rate (Sensitivity, Recall)	$FPR = \frac{FP}{CN}$ False Positive Rate (Fall-out)	$ACC = \frac{TP+TN}{T_{pop}}$ Accuracy	
	$LR_{-} = \frac{fnr}{tnr}$ Negative Likelihood Ratio	$FNR = \frac{FN}{CP}$ False Negative Rate	$TNR = \frac{TN}{CN}$ True negative rate (Specificity)		
	$DOR = \frac{LR_{+}}{LR_{-}}$ Diagnostic Odds Ratio				

Table 4.2: Definitions of the Accuracy, Precision and Recall parameters used to evaluate and compare the performance of different machine-learning classifiers (Wikipedia, 2014a).

4.3 Results and Discussion

The threshold used to segment the aerial images that provide the best results was 0.03. To identify the best threshold, we performed a visual comparison between the segmented images. Higher values of the threshold parameters generate segments that merge different classes together; in our case, some buildings were merged with streets, cars with roads, and trees with grass. However, a lower threshold value would only identify a small portion of the objects. Therefore, the segment threshold parameter needs to be set manually depending on what image objects we aim to classify. Another important parameter used for image segmentation was the minimum segment areas, which were set to 64 pixels ($0.25m \cdot 0.25m \cdot 64pix = 4m^2$). Setting a minimum number of pixels to identify the image objects helps to exclude all objects that are too small to be classified and reduces the overall number of segments. Another reason to set a minimum number of pixels was that the feature extraction of each segment is characterized by a statistical value, which can have some meaning only if it is computed over a significant number of pixels. The total number of segments identified

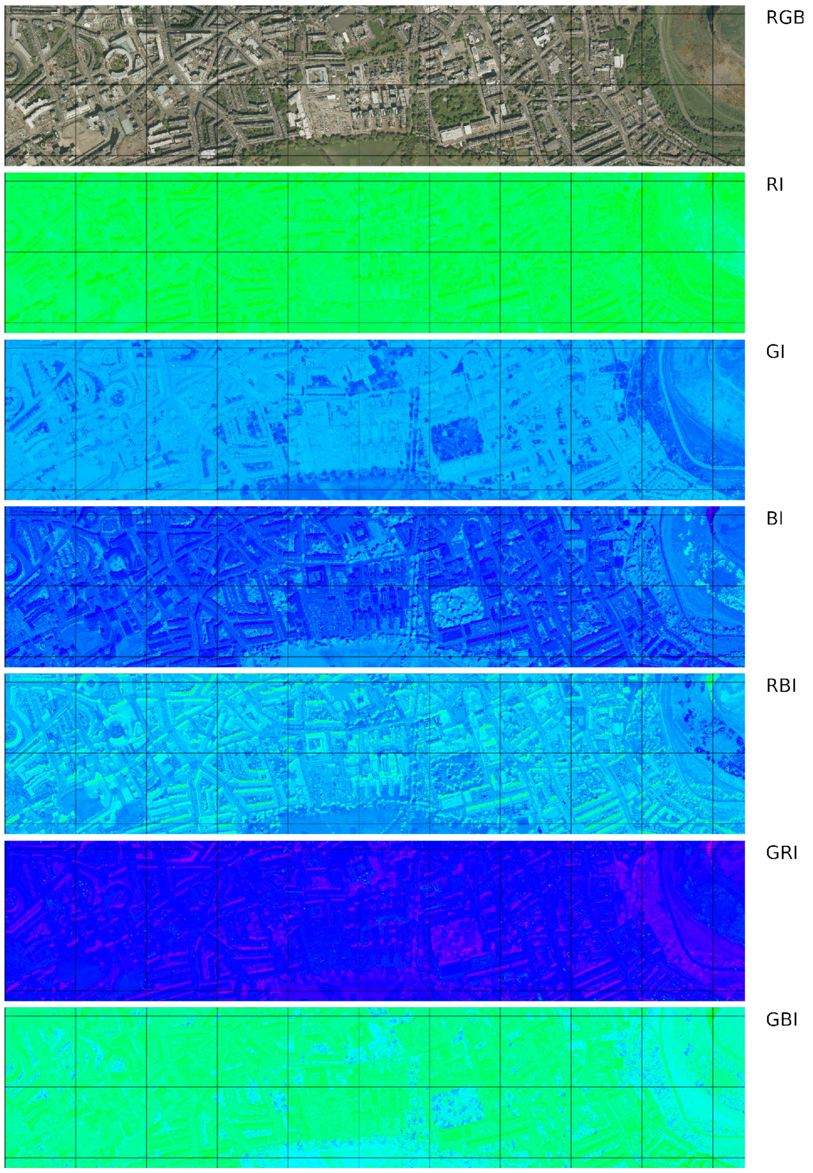


Figure 4.2: Bands ratio indexes used to highlight differences on aerial images, the grid size is 250 m. The indexes are defined as $RI = \frac{R}{R+G+B}$, $GI = \frac{G}{R+G+B}$, $BI = \frac{B}{R+G+B}$, $RBI = \frac{R-B}{R+B}$, $GRI = \frac{G-R}{G+R}$, $GBI = \frac{G-B}{G+B}$ with R, G, B respectively the Red, Green and Blue image bands.

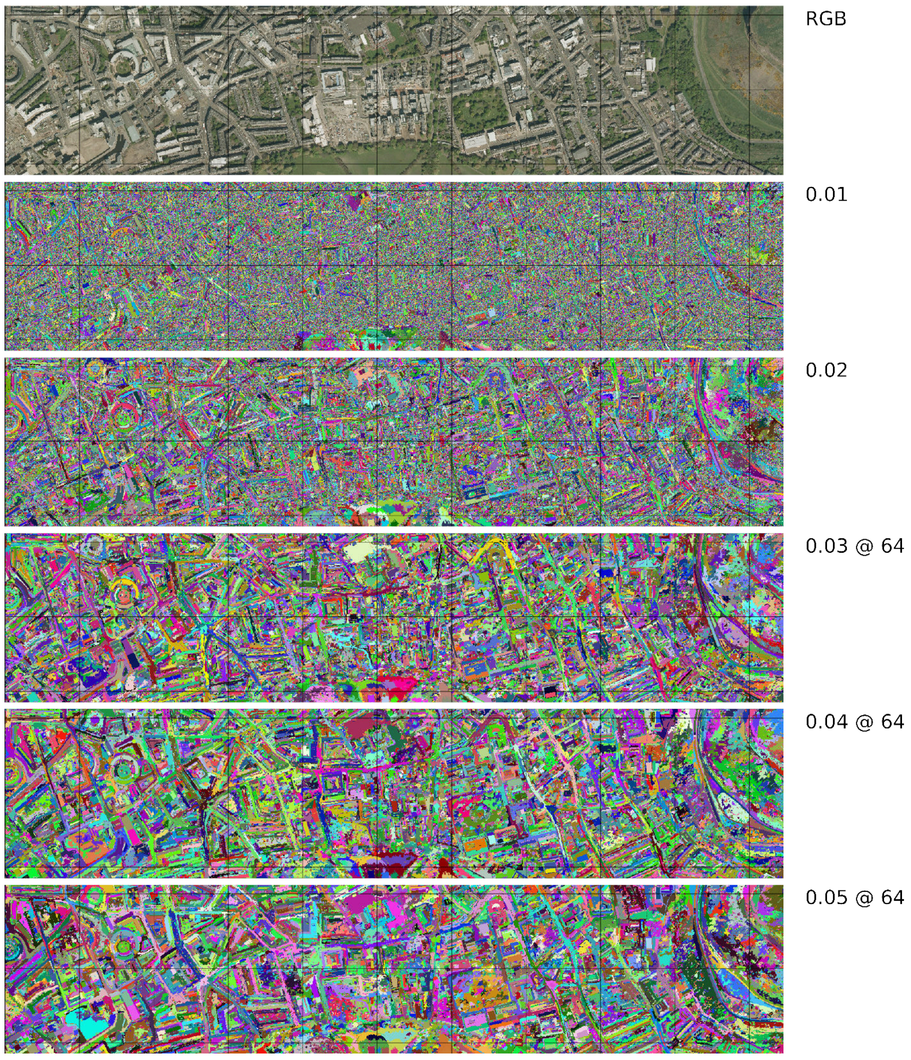


Figure 4.3: The results of different segment thresholds are shown, where indicated a minimum size of 64 pixels ($4\,m^2$) is used, the grid size is 250 m.

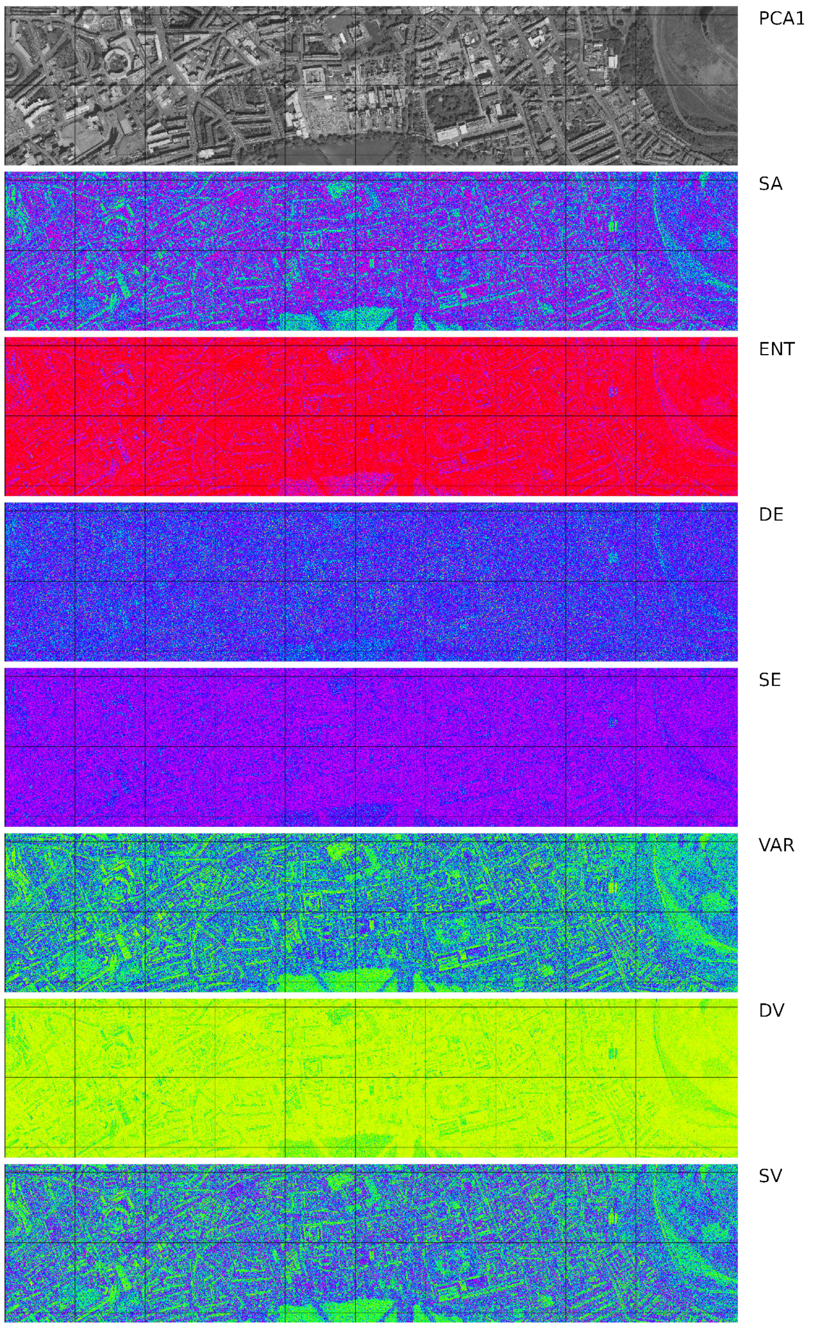


Figure 4.4: First order statistics using the first component of the Principal Component Analysis (PCA1): Sum Average (SA), Entropy (ENT), Difference Entropy (DE), Sum Entropy (SE), Variance (VAR), Difference Variance (DV), Sum Variance (SV), the grid size is 250 m.

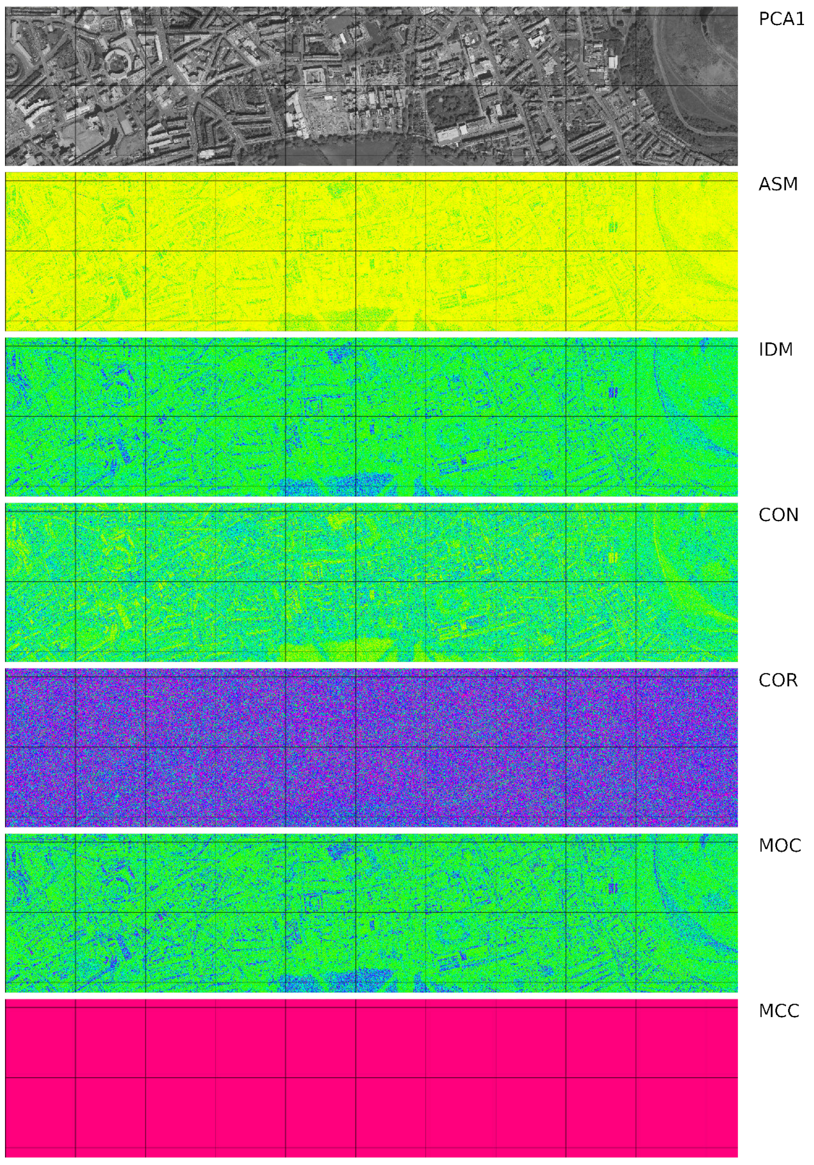
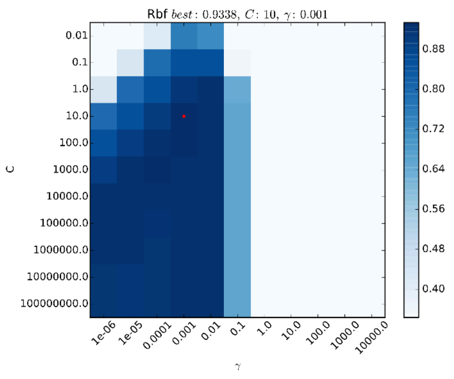
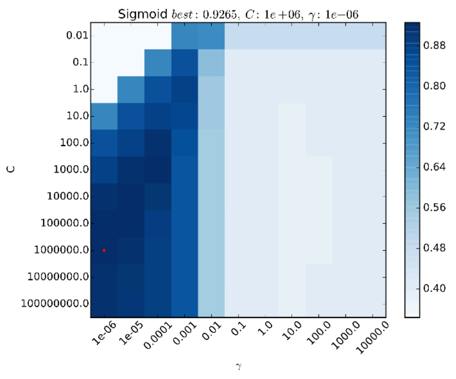


Figure 4.5: Second order statistics using the first component of the Principal Component Analysis (PCA1): Angular Second Moment (ASM), Inverse Difference Moment (IDM), Contrast (CON), Correlation (CORR), Measures of Correlation (MOC), Correlation Coefficient (MCC), the grid size is 250 m.



(a)



(b)

Figure 4.6: Domain exploration of the best set of parameters for each SVC kernel that provides a higher averaged accuracy score using a cross-validation with five k-folds and varying the C and γ values, using a subset of 94 segments per class.

using a threshold of 0.03 and a minimum number of pixels (64) were 691,795.

As described in the methodology, the Support Vector Machine algorithm has two main parameters (C, γ) that have to be set properly to reach good classification results, for this reason a domain exploration of the best parameters was performed. To speed-up the process, the classification was applied on a balanced sub-set of the training classes; Figure 4.6 shows the results.

I ranked the features used to classify the segment, using the `Extra Trees Classifier` with 500 estimators (see results in Figure 4.7 and Table 4.3). The values not reported in the previous Figure and Table are the importances of shape features that are [%]: numb. of isles 0.0506, x extent 0.2155, y extent 0.2692, perimeter of the isles 0.0450, area of the isles 0.0322, compact of the isles 0.0383, fractal dimension of the isles 0.0565, perimeter 0.3075, area 0.1728, bound of the area 0.1531, aratio 0.0313, compact 0.3656, fractal dimension 0.1720.

More than 80 different algorithm set-ups were tested to find the best set of parameters. The 6 best classifiers providing a higher accuracy using 5 shuffled folds are reported in Table 4.5 and visible in Figure 4.8 and are: with an accuracy of 93.82% `Gradient Boosting Classifier` using 100 estimators and a min samples leaf of 3 (`gradient_boost_500_meanleaf3`); with 93.75% the `Random Forest Classifier` using 500 estimators, the entropy function to assess the quality of a split and considering only 50% of the features at each split (`rand_tree_entropy_0p50_500`); with almost the same accuracy 93.73% the `Extra Trees Classifier` using 500 estimators and a min samples leaf of 1 (`extra_tree_500_1`); with 93.59% the `Support Vector Classifier` (SVC), the Gaussian radial basis function kernel (RBF) and $C = 10, \gamma = 0.001$ (`SVC_rbf`); with 93.10% the SVC using a linear kernel with a $C = 1$

	B	GBI	BI	R	G	RBI	GI	RI	SV	Contr	SA	MOC1	GRI	Entr	ASM
third_quart	0.9147	0.9374	1.0932	0.5535	0.6327	0.7541	0.5311	0.4619	0.5453	0.3243	0.3280	0.2042	0.3241	0.3977	0.2108
stddev2	0.8721	1.4561	1.0688	0.4626	0.6478	0.7142	0.4881	0.4377	0.4138	0.3259	0.2591	0.2615	0.2711	0.1624	0.1969
mean	0.7295	0.9598	0.9627	0.5277	0.4729	0.8004	0.7374	0.5152	0.3871	0.3995	0.2591	0.4647	0.3126	0.2793	0.1679
first_quart	0.6767	1.0549	1.1937	0.5292	0.4270	0.8294	0.4254	0.4672	0.2226	0.2849	0.2543	0.4729	0.3113	0.2958	0.3306
median	0.7838	1.2277	0.8959	0.4847	0.4899	0.6453	0.5324	0.4303	0.2591	0.2836	0.3133	0.4225	0.2893	0.3306	0.2377
variance2	0.6420	0.8236	1.0394	0.4218	0.4036	0.4529	0.5369	0.3662	0.2695	0.3256	0.2758	0.2715	0.1714	0.2389	0.1320
stddev	0.9716	0.4381	0.5204	0.8621	0.8772	0.3729	0.2690	0.1884	0.3328	0.2897	0.2187	0.1725	0.1333	0.1454	0.1361
max	0.9791	0.5068	0.7431	0.8448	0.6921	0.4397	0.3627	0.1939	0.2423	0.2875	0.2015	0.0960	0.1976	0.0338	0.1002
mode	0.8039	0.6410	0.4792	0.4945	0.4822	0.7001	0.3362	0.2490	0.2507	0.2808	0.3405	0.0634	0.2184	0.1526	0.1090
range	0.8136	0.3559	0.3883	0.8381	0.7304	0.3284	0.2607	0.2587	0.1288	0.1708	0.0900	0.1011	0.2606	0.1222	0.1055
perc_90	0.5463	0.7825	0.5617	0.4018	0.4213	0.5100	0.3031	0.2680	0.0537	0.0574	0.0546	0.0502	0.1977	0.0521	0.0507
coeff_var	0.5924	0.2290	0.4657	0.6681	0.6246	0.1250	0.2720	0.1977	0.1773	0.1486	0.1850	0.2078	0.0751	0.1580	0.1276
min	0.3571	0.8956	0.4424	0.3348	0.2590	0.5950	0.2762	0.3096	0.0941	0.0987	0.1005	0.0752	0.2688	0.1059	0.0294
variance	0.3180	0.2761	0.3135	0.3893	0.4114	0.3152	0.1904	0.1344	0.3662	0.2950	0.2291	0.2009	0.1327	0.1115	0.0743
skewness	0.6275	0.1655	0.1621	0.5197	0.4344	0.1384	0.1030	0.0834	0.2321	0.2619	0.0998	0.0923	0.0868	0.0663	0.0616
skewness2	0.2701	0.1790	0.2557	0.2120	0.2166	0.3431	0.2056	0.2188	0.2010	0.1324	0.1414	0.1133	0.2438	0.1046	0.0670
kurtosis2	0.2413	0.1593	0.1326	0.2235	0.2720	0.1788	0.1541	0.1344	0.1509	0.1308	0.1730	0.1723	0.1000	0.1953	0.0874
coeff_var2	0.1376	0.1417	0.1470	0.1426	0.1436	0.1034	0.1821	0.2152	0.1472	0.1772	0.2046	0.2276	0.0892	0.1215	0.1060
kurtosis	0.1035	0.0667	0.0765	0.1417	0.1271	0.0809	0.0698	0.0754	0.1385	0.0689	0.0981	0.0693	0.0784	0.0565	0.0569
occurrences	0.1197	0.0032	0.0000	0.1349	0.1501	0.0012	0.0000	0.0030	0.1131	0.1356	0.1480	0.0293	0.0000	0.0380	0.1046

Table 4.3: Features importance in percentage % of each couple raster-statistics.

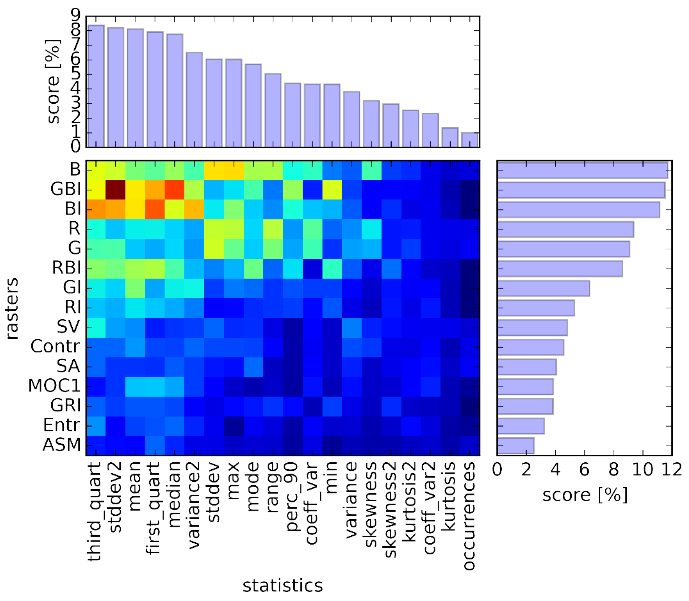


Figure 4.7: Score matrix with the percentage of the importance of each raster and zone statistics considered in the dissertation.

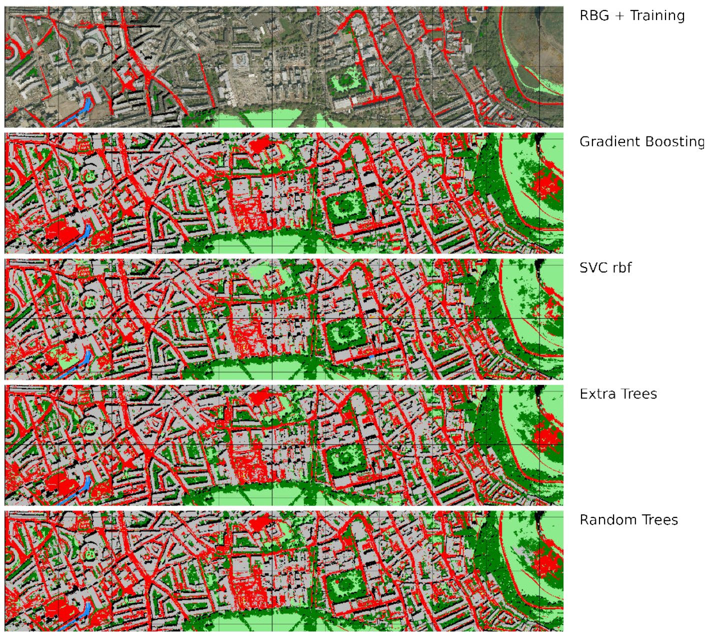


Figure 4.8: In the first plot the segments used for training the machine-learning algorithms are reported, in the other subplots the results of the classification with different algorithms are reported.

(SVC_linear); with 93.07% the SVC using the Sigmoid kernel and $C = 1e^{6}, \gamma = 1e^{-6}$ (SVC_sigmoid);

In Table 4.4 the confusion matrices of the first best six algorithms are reported.

The accuracy score of the first 6 best machine-learning algorithms were very similar (e.g., the best and the worst classifiers differ from each other by only 0.75% of the overall accuracy). This result was in part due to the extremely unbalanced training dataset. Analyzing the confusion matrix, all of the algorithms had particularly good performances to identify: water, grass, tree, shadow which, looking at Table 4.1, were the classes with an accuracy above 95% in most of the classifiers.

All of the algorithms had a lower identification performance for road class, with an accuracy around $\sim$ 84% of the training dataset. This low identification performance for the road class shows that the features extracted from the RGB channels did not characterize and differentiate this class enough from the others. All of the other classes were above 80%. The classes car and bus obtain accuracy that, depending on the classifiers, vary from 94.74% to 79.84% for the car and between 94.25% and 87.37% for the bus. The confusion matrix clearly shows that there were two classes that were more confused from each other: road and building. If available, we can integrate the classification result with the Digital Surface Model (DSM) to help the classifiers distinguish between these classes. A graphical overview of how the classifier was able to distinguish between the classes is provided in Figure 4.9, which illustrates the variation of the *Precision* and the *Recall*, defined in Table 4.2.

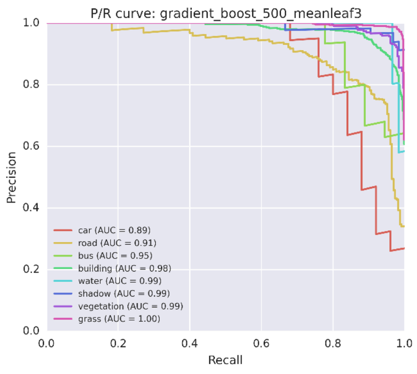
From our tests the Gradient Boost was the best classifier with an overall accuracy of 93.8% and a Cohen's k of 0.92. The overall accuracies from other studies (Duro et al., 2012; Li et al., 2014; Novack et al., 2011; Vieira et al., 2012) lie between 71% and 95%, and the Cohen's k coefficients lie between 0.63 and 0.91. Therefore, the current work seems to be in line with previous works, but it is notable that we were able to reach almost the same accuracy level using only the visible spectrum, while previous studies used at least an additional Near Infra Red (NIR) spectrum or more. The methodology described in this chapter can also be applied to images with other spectral bands and context, and it is not limited to the visible spectrum or the classification of urban land use, but can also be used for classification and interpretation of rural images.

4.4 Conclusions

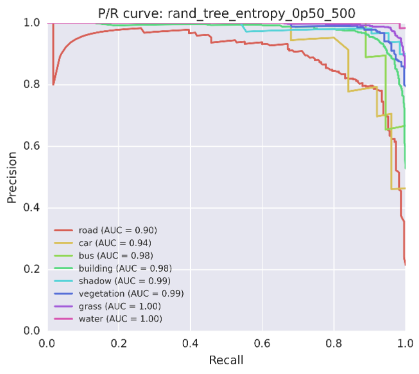
This chapter presents a methodology for the extraction, from RGB aerial ortho-rectified images, of urban micro-environments with a high spatial resolution by combining Geo-Objects Based Image Analysis (GEOBIA) and machine-learning algorithms to achieve

					gradient_boost_500_meanleaf3					
class	water	grass	vegetation	road	car	bus	building	shadow	Tot. Prod.	Prod. Acc. [%]
water	**285**	4			k_Cohen		5	3	*297*	*95.96*
grass	1	**2757**	25	9			18		*2810*	*98.11*
vegetation		39	**1202**	3	1		11	4	*1260*	*95.40*
road		18	8	**924**			170	1	*1121*	*82.43*
car			1		**96**		29		*126*	*76.19*
bus			4			**75**	15		*94*	*79.79*
building		14	8	153	7	5	**3085**	7	*3279*	*94.08*
shadow	2	1	2				7	**302**	*314*	*96.18*
Tot. User	*288*	*2833*	*1250*	*1089*	*104*	*80*	*3340*	*317*	**9301**	
User Acc. [%]	*98.96*	*97.32*	*96.16*	*84.85*	*92.31*	*93.75*	*92.37*	*95.27*		
Overall Acc. [%]	**93.82**									
k_{Cohen} [%]	**91.72**									
					rand_tree_entropy_0p50_500					
water	**289**	2					2	4	*297*	*97.31*
grass	1	**2742**	37	10			20		*2810*	*97.58*
vegetation		40	**1196**	4			17	3	*1260*	*94.92*
road		26	9	**927**			158	1	*1121*	*82.69*
car			1	1	**90**	1	33		*126*	*71.43*
bus		1	2			**82**	9		*94*	*87.23*
building	1	19	11	158	5	4	**3072**	9	*3279*	*93.69*
shadow	2		2				3	*307*	*314*	*97.77*
Tot. User	*293*	*2830*	*1258*	*1100*	*95*	*87*	*3314*	*324*	**9301**	
User Acc. [%]	*98.63*	*96.89*	*95.07*	*84.27*	*94.74*	*94.25*	*92.70*	*94.75*		
Overall Acc. [%]	**93.59**									
k_{Cohen} [%]	**91.43**									
					extra_tree_500_1					
water	**293**	1					2	1	*297*	*98.65*
grass		**2747**	38	10			15		*2810*	*97.76*
vegetation		35	**1211**	3	1		7	3	*1260*	*96.11*
road		26	9	**906**			179	1	*1121*	*80.82*
car			1		**96**	1	28		*126*	*76.19*
bus			1			**82**	11		*94*	*87.23*
building		18	16	151	5	4	**3078**	7	*3279*	*93.87*
shadow	1		2				6	**305**	*314*	*97.13*
Tot. User	*294*	*2827*	*1278*	*1070*	*102*	*87*	*3326*	*317*	**9301**	
User Acc. [%]	*99.66*	*97.17*	*94.76*	*84.67*	*94.12*	*94.25*	*92.54*	*96.21*		
Overall Acc. [%]	**93.73**									
k_{Cohen} [%]	**91.61**									
					SVC_rbf					
water	**290**	1					3	3	*297*	*97.64*
grass	1	**2754**	28	11	1		15		*2810*	*98.01*
vegetation		33	**1212**	3	3		5	4	*1260*	*96.19*
road		18	5	**913**			184	1	*1121*	*81.45*
car			3	1	**98**	1	23		*126*	*77.78*
bus					1	**83**	10		*94*	*88.30*
building	2	26	9	153	15	8	**3058**	8	*3279*	*93.26*
shadow	3		4		1		9	**297**	*314*	*94.59*
Tot. User	*296*	*2832*	*1261*	*1081*	*119*	*92*	*3307*	*313*	*9301*	
User Acc. [%]	*97.97*	*97.25*	*96.11*	*84.46*	*82.35*	*90.22*	*92.47*	*94.89*		
Overall Acc. [%]	**93.59**									
k_{Cohen} [%]	**91.43**									
					SVC_linear					
water	**290**	3					1	3	*297*	*97.64*
grass	4	**2748**	30	13			15		*2810*	*97.79*
vegetation		37	**1208**	1	3		6	5	*1260*	*95.87*
road	1	22	5	**906**			186	1	*1121*	*80.82*
car			3	1	**99**	1	22		*126*	*78.57*
bus					2	**83**	9		*94*	*88.30*
building	4	26	11	174	19	11	**3029**	5	*3279*	*92.38*
shadow	3	1	4	1	1		8	**296**	*314*	*94.27*
Tot. User	*302*	*2837*	*1261*	*1096*	*124*	*95*	*3276*	*310*	**9301**	
User Acc. [%]	*96.03*	*96.86*	*95.80*	*82.66*	*79.84*	*87.37*	*92.46*	*95.48*		
Overall Acc. [%]	**93.10**									
k_{Cohen} [%]	**90.78**									
					SVC_sigmoid					
water	**290**	3					1	3	*297*	*97.64*
grass	4	**2748**	30	13			15		*2810*	*97.79*
vegetation		37	**1208**	1	3		6	5	*1260*	*95.87*
road	1	22	5	**904**			188	1	*1121*	*80.64*
car			3	1	**99**	1	22		*126*	*78.57*
bus					2	**83**	9		*94*	*88.30*
building	4	26	11	175	19	11	**3028**	5	*3279*	*92.35*
shadow	3	1	4	1	1		8	**296**	*314*	*94.27*
Tot. User	*302*	*2837*	*1261*	*1095*	*124*	*95*	*3277*	*310*	*9301*	
User Acc. [%]	*96.03*	*96.86*	*95.80*	*82.56*	*79.84*	*87.37*	*92.40*	*95.48*		
Overall Acc. [%]	**93.07**									
k_{Cohen} [%]	**90.74**									

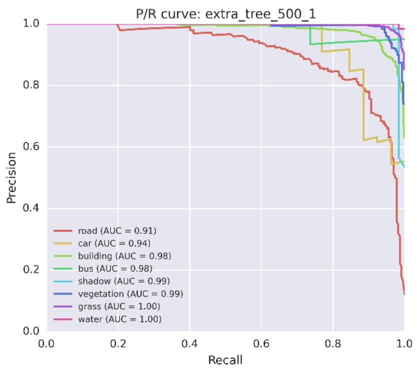
Table 4.4: Confusion matrix for the first best six classifiers.



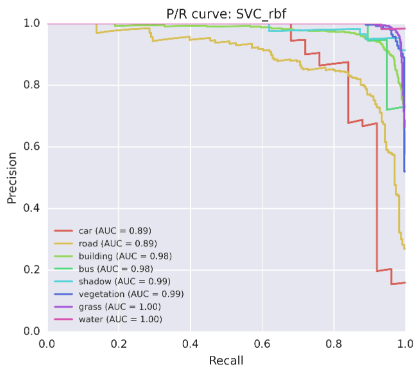
(a) Gradient Boosting Tree



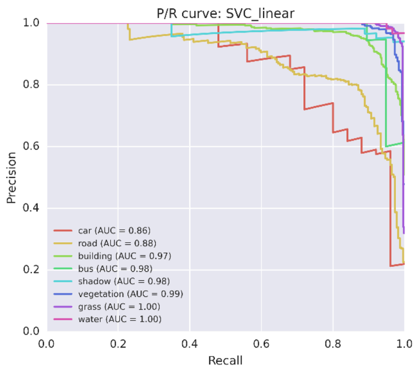
(b) Random Tree



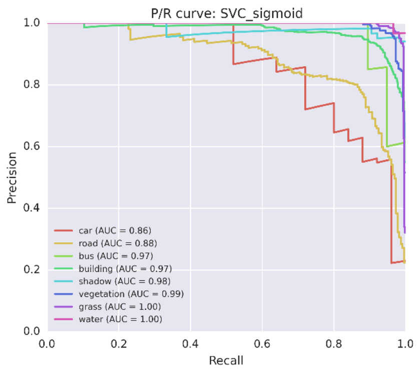
(c) Extremely Randomize Tree



(d) SVC radial basis function (RBF)



(e) SVC linear



(f) SVC sigmoid

Figure 4.9: The Precision/Recall (PR) curves provide an informative picture of algorithm performance. Ideally, if all of the segments were classified correctly without errors, both the value for Precision and Recall are 1, and the Area Under the Curve (AUC) reported in the legend will be 1. (a) Gradient Boosting Tree (GBC) (c) Extremely Randomize Tree (ET) (e) SVC Sigmoid (b) SVC linear (d) SVC radial function (RBF) basis (f) Random Forest (RF).

name	mean [%]	max [%]	min [%]	std [%]	time [s]
gradient_boost_500_meanleaf3	93.82	94.78	92.91	0.67	2939.2
rand_tree_entropy_0p50_500	93.75	94.41	92.96	0.56	2202.7
extra_tree_500_1	93.73	94.09	93.12	0.34	34.3
SVC_rbf	93.59	94.35	92.48	0.64	81.6
SVC_linear	93.10	93.98	91.99	0.67	92.2
SVC_sigmoid	93.07	93.82	91.99	0.63	90.9

Table 4.5: Total accuracy reach by each classifier, in the table are reported mean, maximum, minimum, standard deviation and the time needed for cross-validation using 5 k-folds.

a reliable accuracy. The entire methodology is based on open-source software. The minimum input data required is quite common and easy to retrieve, which makes this research easier to reproduce, modify and apply to other contexts. The methodology was verified with more than 9,300 training segments, testing several pre-processing and machine-learning algorithm set-ups. The classifiers achieve an overall accuracy greater than 93% and a Cohen coefficient above 0.92, and the results are comparable to those of other literature for the same classification task (Duro et al., 2012; Li et al., 2014; Moskal et al., 2011; Novack et al., 2011; Vieira et al., 2012).

The accuracy achieved with the presented methodology could be sufficient to characterize urban micro-environmental conditions. For example, limiting the analysis to micro-climatic conditions, even if the classifiers confuse some classes such as the building and road, these two classes have generally similar material property with regard to the emissivity and heat transmission and the same considerations are valid for grass and trees. Further improvement of the overall accuracy can be reached using extra information to characterize each segment, such as another spectrum band or the elevation difference between the digital terrain model (DTM) and the digital surface model (DSM).

The GRASS GIS module `v.class.ml` developed and used in this work is not limited to image and segment classification, but presents a more general tool for wider application that allows the classification of any kind of vector map using the attribute table as a data source for machine-learning classifiers. The segment classification was chosen as a test due to the relatively high number of segments (2,726,635) and features (307), but the module can also be applied in other contexts and research fields.

Chapter 5

Geographical factors and urban Nitrogen Dioxide micro-environments

In this chapter I examined the correlation between Nitrogen Dioxide (NO_2) concentrations measured using passive diffusive tubes (PDTs) and geographical factors. The study area is the city centre of Edinburgh. I considered nine urban micro-environment classes; for each of them I computed the distance and the Sky View Factor (SVF). For all of these maps I calculated the Sum Average texture using different window dimensions. The chapter presents which factors have a stronger correlation with the measured concentration of NO_2. These geographical factors were used as input to apply several machine-learning regression algorithms, to assess the spatial variability of NO_2 within an urban context.

5.1 Introduction

To examine the effects of air pollutants on human health, an accurate assessment of spatial variability of the pollutants is needed. For this reason an increasing amount of literature focuses on spatial variability within-city at an intra urban scale. At this scale the traditional central site monitors based on official measurement stations (OMS) are inadequate.

This chapter aims to understand if it is possible to treat the spatial variability of air pollutants within the domain of GIS modelling, neglecting, as first approximation, the meteorological conditions. Moreover I want to identify which geographical factors influence pollutant concentration within an urban context.

Levy et al. (2014) highlights that the space and time variability of multipollutant mix varies considerably throughout a city, and even if a single pollutant that acts as proxy measure for the entire mix under all circumstances does not exist, indicates Nitrogen Dioxide (NO_2) as "*the best available indicators of spatial variation in exposure to the outdoor urban air pollutant mixture*".

Similarly to other studies, passive diffusive tubes (PDTs) were used to measure and characterize the spatial variability of the NO_2 (Madsen et al., 2007); what is new with respect to previous studies is the high density of PDTs used and the relatively long time of the measurements campaign (6 consecutive weeks).

I tested the correlation between several geographical factors and the average of the measured concentrations of NO_2, and then I used these geographical factors to regress the NO_2 and assess the spatial variability of the air pollutant. The presented work extends the concept of the Land Use Regression method to base the regression not only on land-use categories but also on other geographical factors that could be significant to describe micro-environment conditions of the physical phenomena under investigation.

5.2 Materials and methods

The study area is the city centre of Edinburgh, Scotland (UK); see previous chapter for further information on the area object of this study. The methodology combines the urban micro-environments extracted by the RGB aerial ortho-rectified images with the elevation difference between the Digital Terrain Model (DTM) and the Digital Surface Model (DSM) to better distinguish between buildings and roads or trees and grass, improving the final characterization of the urban morphology.

Thirty-seven passive diffusion tubes (PDTs) of Nitrogen Dioxide were positioned by the School of Chemistry and Engineering of the University of Edinburgh under the supervision of Dr. M. Heal and Dr. C. Lin. The passive samplers were analysed weekly for a six-week campaign starting on 2 December 2013 and ending on 13 January 2014. The area covered by the PDTs was less than 9 sq.km with a density of about 0.250 sq.km per passive sample, and covered the south part of the city centre of Edinburgh and suburban areas ()see Figure 5.1 for more details).

Nine geographical factors were considered in this work: buildings, tree, grass, roads split in five main classes: very high, high, medium, low, very low traffic, and urban tunnels. Since during this work we had not the opportunity to analyse data coming from the traffic counter of the city council, the traffic classification of roads were based on the Open Street Map (OSM) project classification. The OSM project founded in 2004 by Steve Cost collects and organizes geo-referenced data provided and checked from a growing number of volunteers, often in the scientific literature we refer to this data as Volunteered Geographic Information (VGI). Roads vector data

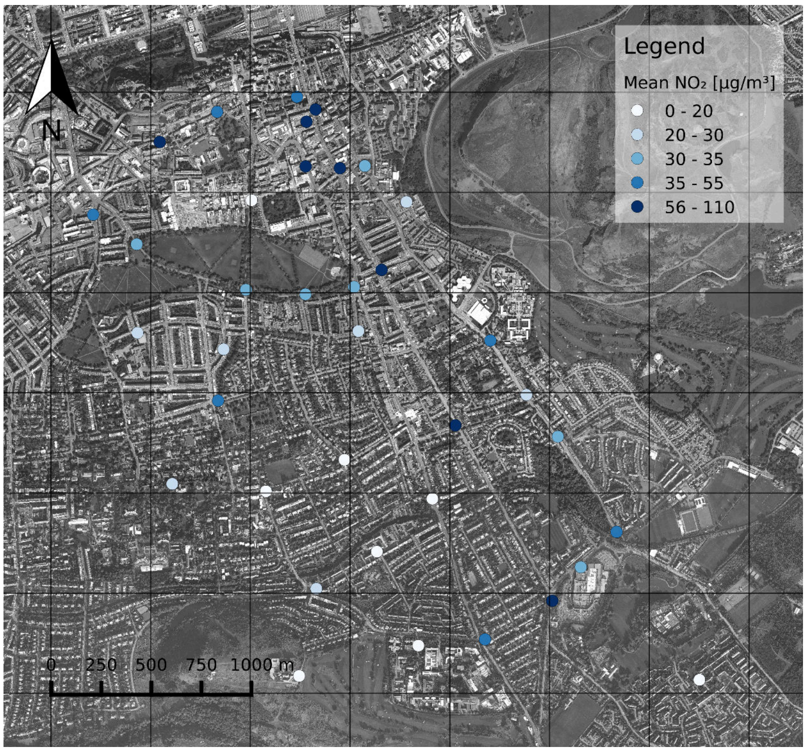


Figure 5.1: The white points represent the passive samplers used to monitor the concentration of Nitrogen Dioxide (NO_2) and to characterize the spatial variability of air pollutants in an urban context during the six weeks of the measurement campaign.

were transformed to raster using an increasing buffer around the road starting from 8m for very low until 18m for very high traffic roads and removing from the buffer all pixels that were already classified as buildings. The other geographical factors were extracted from the visible spectrum (RGB) of aerial ortho-rectified images provided by the Ordinary Survey (British Ordnance Survey, 2012), with a resolution of 0.25 m by 0.25m, combining a Geo-Object Image Analysis (GEOBIA) and machine-learning classifiers as explained in the previous chapter. The classified raster map was re-sampled from a resolution of 0.25m to a 5m using the statistic mode as value of the pixel. The spatial resolution has been reduced, because spatial variability of an average week's concentration of air pollutants is difficult to model and to measure at this scale, and to make the processing time faster. Moreover a spatial resolution of 0.25 m seems not coherent with deployment of the measurement campaign and probably not feasible with current technologies and costs. A lower resolution of 10 or 25m, on the other hand, hides the effect of important geographical factors such as urban grass and trees.

For each of these geographical factors three main features were extracted: the distance, the Sky View Factor (SVF), and the Sum Average texture. The SVF is defined as:

$$SVF = \frac{\int_0^{2\pi} cos(\phi(\lambda))d\lambda}{2\pi} \tag{5.1}$$

and represents the portion of sky visible from a specific point. For geographical features that haven't got an elevation such as grass, roads and tunnels a default value of 100m was assigned. The SVF was computed with an angle step of 5 degrees.

The Sum Average (SA) texture was used to "describe" the context of each point with a moving window of a different number of pixels (3, 5, 9, 15, 21, 27, 33, 39, 45), therefore considering a zone of influence that goes from 15m up to 225m, for each geographical factor.

For each of the 243 geographical factors $(9(class) \cdot 3(features) \cdot 9(texture)$ I computed the Spearman's correlation coefficient (ρ_s) in respect to the mean concentration of NO_2 measured during the winter campaign. All the geographical factors satisfying the following condition: $\rho_s > 0.4 \vee \rho_s < -0.4$ were manually selected considering the class type of the geographical features (e.g. grass, tree, roads, etc.), the feature type (e.g. distance, SVF, etc.) and the texture value if used.

A Recursive Feature Elimination (RFE) was used to reduce the number of features and select only those features providing a higher score. The RFE process consists of defining an estimator, in this work the Support Vector Regression (SVR) with $kernel = RBF, C = 1, \gamma = 1$ was used, and select features recursively considering only a small subset of the features available. Only features with the highest score are selected. To

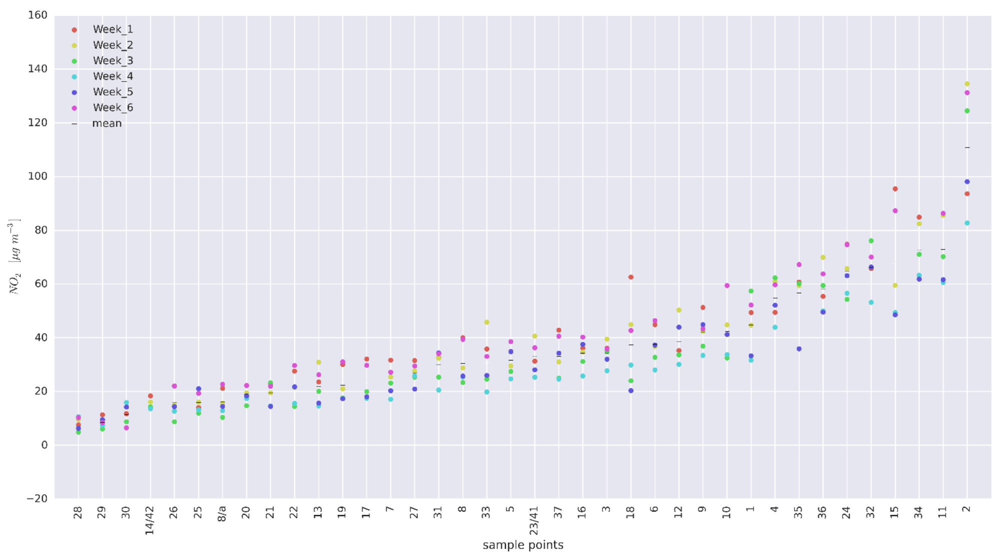


Figure 5.2: Concentration of Nitrogen Dioxide (NO_2) measured during the six weeks, sorted by the mean value of each point.

evaluate the performance avoiding, or limiting, the over-fit problem, we compute the Mean Square Error (MSE) of a cross-validated dataset using the left-one-out (LOO) method. I tested 18 different regression algorithms with several set-ups to find the best set of options for each regressor that maximize performance.

5.3 Results and discussion

The work is based on the assumption that the NO_2 concentrations are strictly dependent on the geographical position. To verify this assumption, the sample measurements were sorted using the average value during the six weeks for each point and sorting the points based on this value (see Figure 5.2). Since we were not interested in a concentration value itself, but in a spatial variability of the NO_2, we scaled the values to have $\mu = 0$ and $\sigma = 1$ (see Figure 5.3). Both graphs show that points characterized by a low concentration of NO_2 have low concentrations in all the weeks, accordingly the spatial variability of the NO_2 seems stable during the six weeks of the campaign. Figure 5.4 shows a summary of the Spearman's rank correlation coefficient (ρ_s) between weeks of the scaled dataset, and Figure 5.5 shows the detailed plot for each week combination. The Spearman's rank correlation coefficient assesses how well the relationship between two variables can be described using a monotonic function. If there are no repeated data values, a perfect Spearman correlation of +1 or -1 occurs when each of the variables is a perfect monotone function of the other. All couples of weeks had a strong Spearman correlation coefficient between 0.88 and 0.97. From this dataset

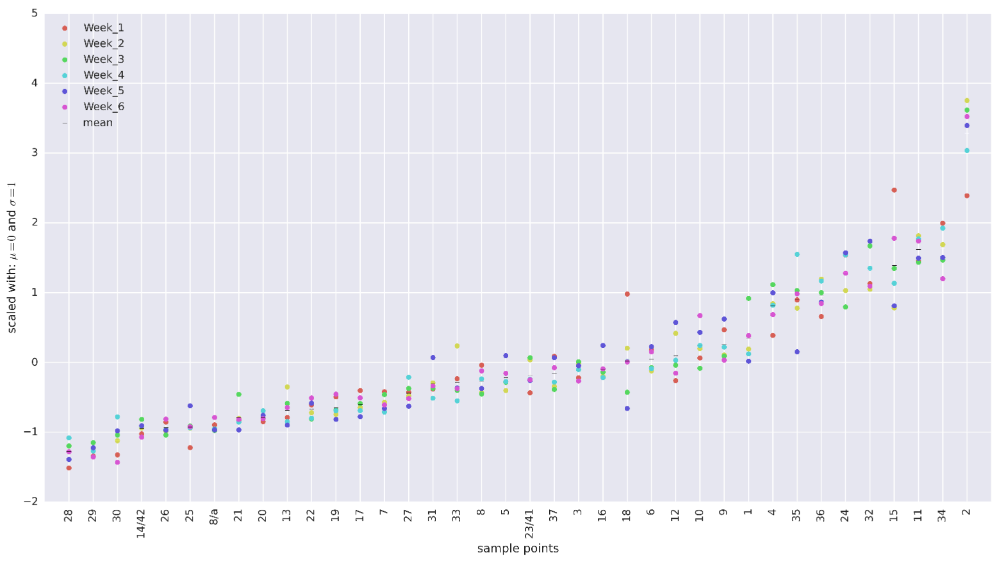


Figure 5.3: Concentration of Nitrogen Dioxide (NO_2) measured during the six weeks, sorted by the mean value of each point and scaled to have: $\mu = 0$ and $\sigma = 1$.

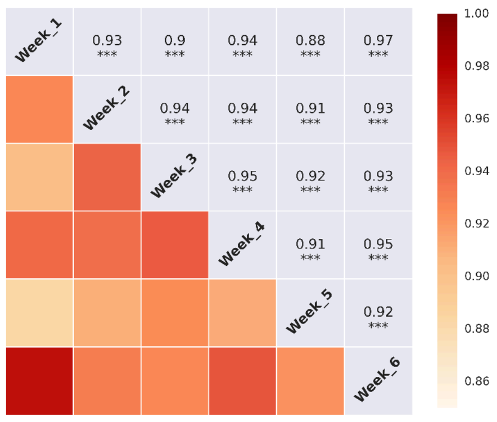
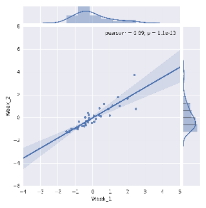
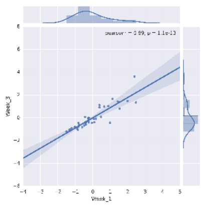
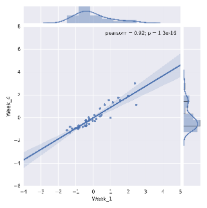
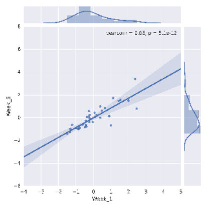


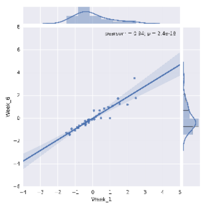
Figure 5.4: Spearman correlation coefficient between weeks measurements of NO_2 scaled with $\mu = 0$ and $\sigma = 1$.

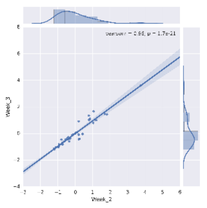

(a) Week_1 VS Week_2

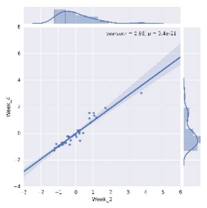

(b) Week_1 VS Week_3

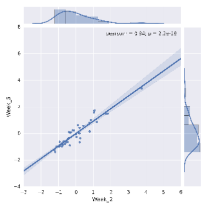

(c) Week_1 VS Week_4

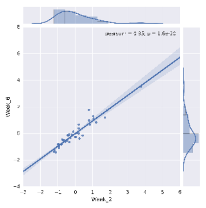

(d) Week_1 VS Week_5

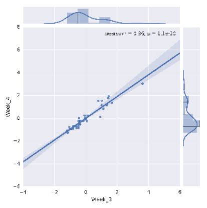

(e) Week_1 VS Week_6

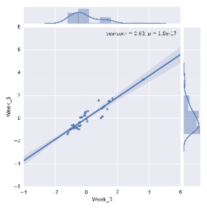

(f) Week_2 VS Week_3

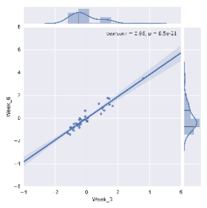

(g) Week_2 VS Week_4

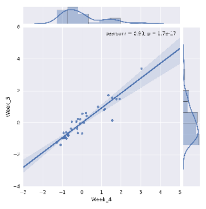

(h) Week_2 VS Week_5

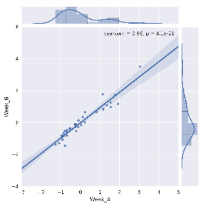

(i) Week_2 VS Week_6


(j) Week_3 VS Week_4


(k) Week_3 VS Week_5


(l) Week_3 VS Week_6


(m) Week_4 VS Week_5


(n) Week_4 VS Week_6

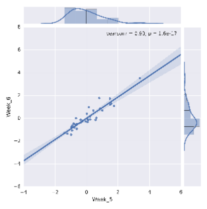

(o) Week_5 VS Week_6

Figure 5.5: Detail of the correlation between weeks, the values are scaled with $\mu = 0$ and $\sigma = 1$.

and graphs the geographical position seems to be the main driver or proxy of the NO_2 concentrations.

The Spearman's rank correlation coefficient was computed for each geographical factor. The geographical factors with $\rho_s > 0.4$ are reported in Figure 5.6, and in Figure 5.7 are reported all of them with $\rho_s < -0.4$. The subset of geographical factors with the higher Spearman's coefficient were manually selected avoiding repetition about: class, feature, and texture type. The selected geographical factors with a positive correlation coefficient are shown in Figure 5.8, while Figure 5.9 shows the geographical factors with a negative correlation's coefficient. The details and values of the Spearman's coefficient are shown in Table 5.1.

These graphs highlight a strong correlation between NO_2 concentrations and grass distance ($\rho_s = 0.60$), and in particular considering the grass presence within an area of 45m ($\rho_s = 0.66$) another linked indicator is the grass' SVF ($\rho_s = 0.51$). A strong correlation exists with the presence of very high traffic roads in a radius of 195m; this confirms the role of traffic as main driver of the NO_2 concentrations. The strong correlation with the grass class could be due to the fact that where we have grass we haven't got NO_2 sources (e.g. roads). A correlation also exists between NO_2 concentrations and the SVF of urban tunnels in a surrounding area of 135m ($\rho_s = 0.59$). It is not clear from these data if the tunnels influence the NO_2 as a geomorphological factor or because they indicate a certain urban context in Edinburgh. Similar considerations are valid for the presence of buildings in a radius of 195m ($\rho_s = 0.46$). A strong negative correlation exists between NO_2 concentrations and the SVF ($\rho_s = -0.67$) and distance ($\rho_s = 0.66$) of very high and high traffic roads and also considering only very high traffic roads with a ρ_s respectively of 0.59 and 0.56 for SVF and distance, a similar correlation was found with the urban tunnels with distance ($\rho_s = -0.56$) and SVF ($\rho_s = -0.55$). We found a negative correlation ($\rho_s = -0.49$) with the presence of the tree within a surrounding area of 25 m, which means that a higher presence of trees generally correspond to a lower concentration of NO_2.

As highlighted by Cape (2009), the PDTs could be sensitive to wind speed, temperature and humidity, and these meteorological conditions could have influenced the campaign, perhaps the correlation with grass and trees is partially due to the microclimatic conditions influenced by urban vegetation. However the average humidity condition of Edinburgh is quite high, and therefore we can assume a very low spatial variability of this parameter.

Table 5.2 reports the values of the Root Mean Square Error (RMSE) of the regression estimators. The correlation of the best regressed values and the measured concentrations of NO_2 are shown in Figure 5.10 with a Spearman's correlation coefficient of 0.78. A direct comparison between the estimate concentrations and measured data is shown in Figure 5.11. Figure 5.12 shows the raster map with the assessed concentrations of NO_2.

5.4 Conclusions

The work highlight how the concentration of Nitrogen Dioxide (NO_2) is strictly dependent on geographical position. Therefore it is possible to explore the spatial vari-

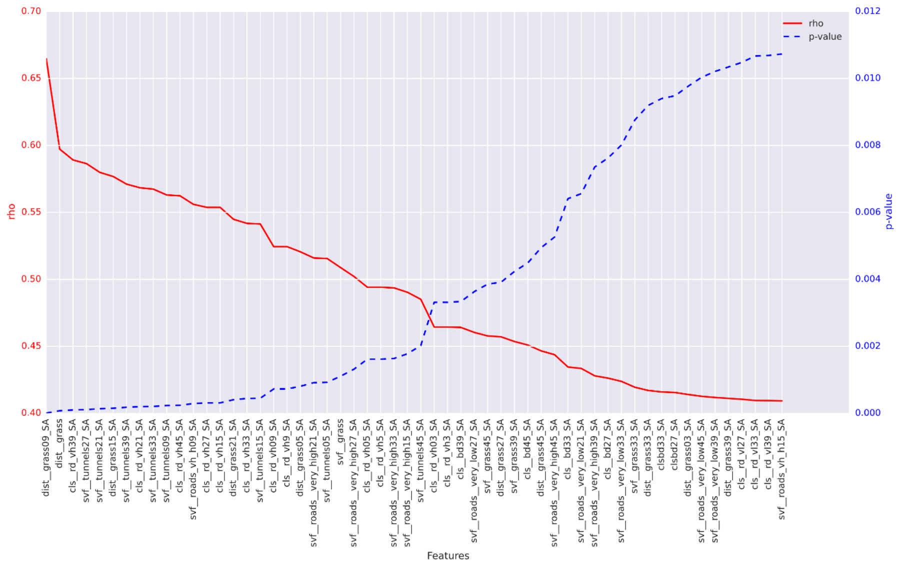


Figure 5.6: Positive Spearman's correlation coefficient between Geographical features and measurements of NO_2 concentrations.

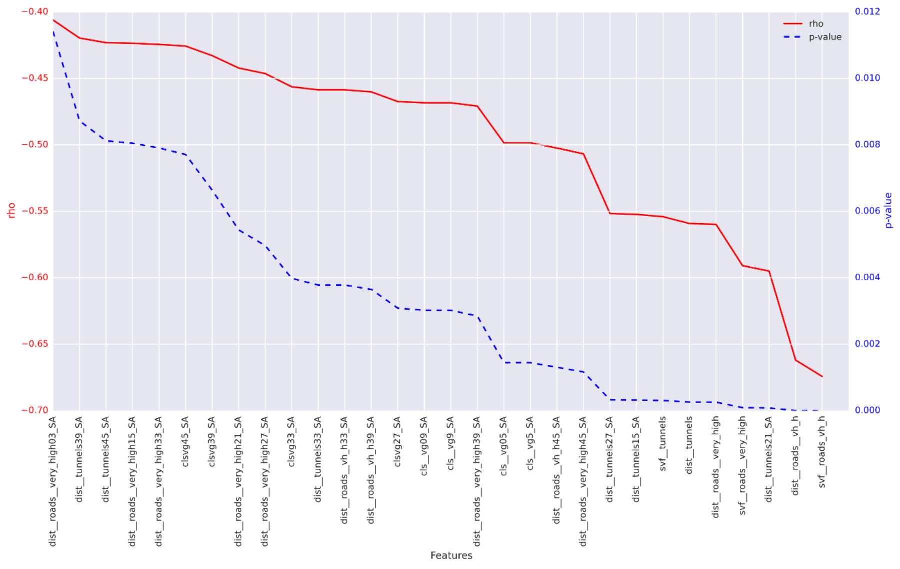


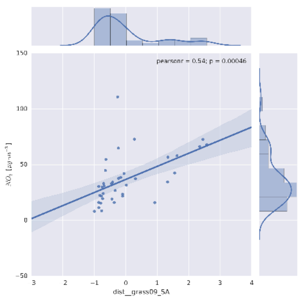
Figure 5.7: Positive Spearman's correlation coefficient between Geographical features and measurements of NO_2 concentrations.

label	class type	feature type	texture [pixels]	texture [m]	ρ_s	P-value
dist__grass09_SA	grass	distance	9	45	0.6648	0.000005
dist__grass	grass	distance			0.5972	0.000075
cls__rd_vh39_SA	very high traffic road	class	39	195	0.5891	0.000100
svf__tunnels27_SA	tunnel	SVF	27	135	0.5864	0.000109
svf__roads_vh_h09_SA	very high + high traffic road	SVF	9	45	0.5560	0.000290
svf__grass	grass	SVF			0.5088	0.001107
cls__bd39_SA	building	class	39	195	0.4641	0.003337
svf__roads_vh_h	very high + high traffic road	SVF			-0.6742	0.000003
dist__roads__vh_h	very high + high traffic road	distance			-0.6619	0.000006
dist__tunnels21_SA	tunnel	distance	21	105	-0.5950	0.000082
svf__roads__very_high	very high traffic road	SVF			-0.5909	0.000094
dist__roads__very_high	very high traffic road	distance			-0.5598	0.000258
dist__tunnels	tunnel	distance			-0.5592	0.000263
svf__tunnels	tunnel	SVF			-0.5540	0.000308
dist__tunnels15_SA	tunnel	distance	15	75	-0.5523	0.000324
dist__roads__very_high45_SA	very high traffic road	distance	45	225	-0.5068	0.001168
dist__roads__vh_h45_SA	very high + high traffic road	distance	45	225	-0.5024	0.001309
cls__vg5_SA	tree	class	5	25	-0.4985	0.001447

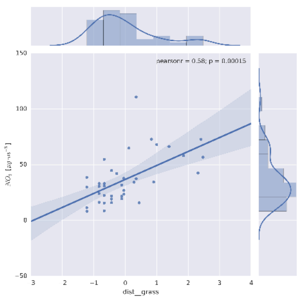
Table 5.1: Spearman's correlation coefficient (ρ_s) computed between the average value of the concentration of NO_2 and different geographical factors, In this Table are reported selected geographical factors with a moderately or strong positive or negative Spearman's correlation coefficient $\rho_s > 0.4 \vee \rho_s < -0.4$.

model	parameters	mean [$\mu g \cdot m^{-3}$]	max [$\mu g \cdot m^{-3}$]	min [$\mu g \cdot m^{-3}$]	std [$\mu g \cdot m^{-3}$]	time [s]
Support Vector Regression	kernel=Sigmoid, $C = 10$, $\gamma = 0.01$	15.17	62.48	0.41	25.31	0.07
Support Vector Regression	kernel=linear, $C = 0.1$	15.54	63.24	0.47	25.56	0.06
Support Vector Regression	kernel=RBF, $C = 100$, $\gamma = 0.001$	15.89	60.42	0.46	24.61	0.08
Ridge	$\alpha = 100$	15.99	62.40	0.00	25.50	0.03
BayesianRidge		16.47	62.29	0.30	25.59	0.24
Elastic Network	$\alpha = 10$, $l1\_ratio = 0$	16.52	65.49	0.47	26.52	0.56
Gradient Boosting Regressor	loss=lad, learning_rate=0.5, numb. estimators=10	17.02	45.67	0.63	20.54	0.18
Extra Trees Regressor	numb. estimators=500, max_features=sqrt	17.09	62.75	0.33	25.45	9.35
Lars	numb. non-zero coefficient=5	17.27	67.72	0.20	27.37	0.05
Lasso Lars	α=1	17.63	69.65	0.54	28.12	0.05
Random Forest Regressor	numb. estimators=500, max_features=log2	17.64	65.03	1.33	26.40	18.79
Bagging Regressor	numb. estimators=500, max_features=20	17.77	65.03	0.83	26.33	33.02
Lasso	$\alpha = 1$	20.10	61.90	0.28	26.94	0.05
Perceptron	penality=l2, $\alpha = 100$	21.60	81.72	0.20	33.45	0.02
SGD Regressor	loss=squared_epsilon_insensitive, penality=l2	21.60	81.72	0.20	33.45	0.02
Logistic Regression	penality=l1, $C = 10$	22.53	48.00	1.00	26.67	0.73
ARD Regression		30.39	73.98	3.18	35.26	12.13
Passive Aggressive Regressor	$C = 10$, loss=epsilon	38.80	105.57	1.89	48.28	0.02

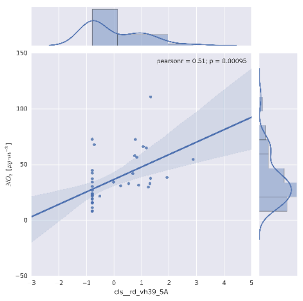
Table 5.2: Root Mean Square Error (RMSE) computed between the average value of the concentration of NO_2 and the regression value assess using a left-one-out cross-validation method for each machine learning algorithm. In this Table are reported the mean value the maximum, minimum, standard deviation and the computational time needed for cross-validation.



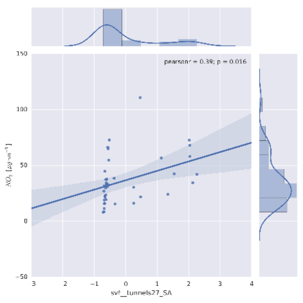
(a) $\rho_s = 0.66$, $p = 5.3e^{-6}$
grass, distance, texture 9 pixels



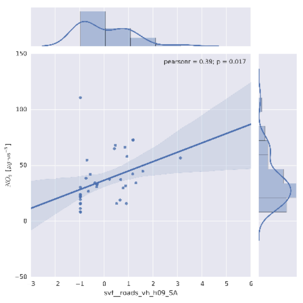
(b) $\rho_s = 0.60$, $p = 7.5e^{-5}$
grass, distance



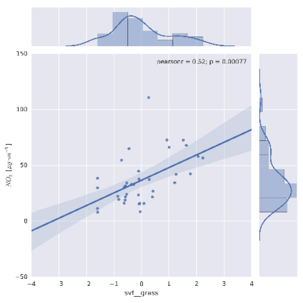
(c) $\rho_s = 0.59$, $p = 9.9e^{-5}$
very high traffic road, class, texture 39 pixels



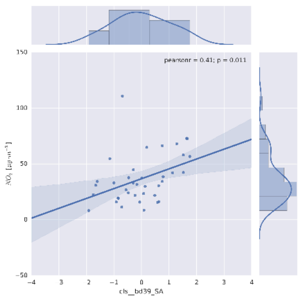
(d) $\rho_s = 0.59$, $p = 1.1e^{-4}$
tunnel, SVF, texture 27 pixels



(e) $\rho_s = 0.56$, $p = 2.8e^{-4}$
very high + high traffic road, SVF, texture 9 pixels



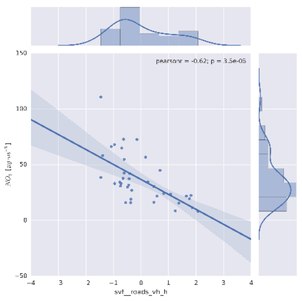
(f) $\rho_s = 0.51$, $p = 1.1e^{-3}$
grass, SVF



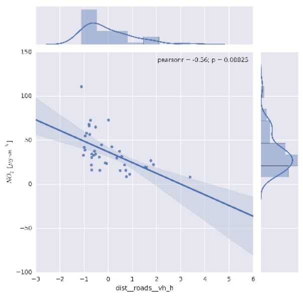
(g) $\rho_s = 0.46$, $p = 3.3e^{-3}$
building, class, texture 39 pixels

Figure 5.8: Correlation between the transformed geographical features ($\mu = 0$ and $\sigma = 1$) and NO_2 concentration. In each sub-plot are reported: the ρ_s and the relative p, the feature land-use class, the feature type and, if a texture is applied, the number of pixels used to compute the texture.

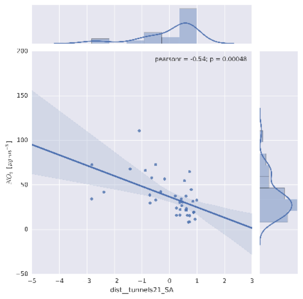
ability of the NO_2 taking into account the geographical factors. In this chapter several factors were taken into account: 9 land use classes were used (very high/high/medium/low/very low traffic roads, trees, grass, building, tunnels) two different features were computed (distance and SVF) and for all these factors 9 different texture distances were considered. For each of these 243 geographical factors we checked the



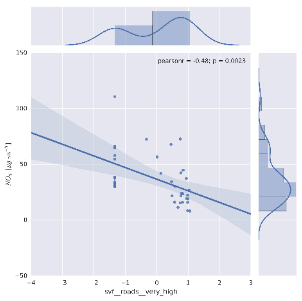
(a) $\rho_s = -0.67$, $p = 3.4e^{-6}$ very high + high traffic road, SVF



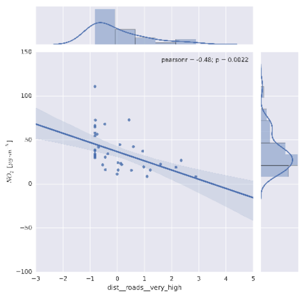
(b) $\rho_s = -0.66$, $p = 6.0e^{-6}$ very high + high traffic road, distance



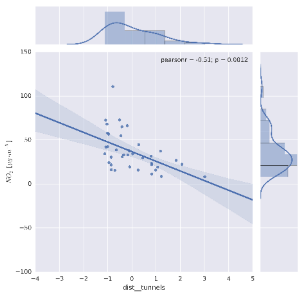
(c) $\rho_s = -0.59$, $p = 8.1e^{-5}$ tunnel, distance, texture 21 pixels



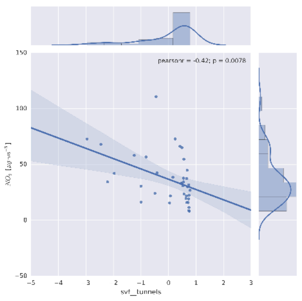
(d) $\rho_s = -0.59$, $p = 9.4e^{-5}$ very high traffic road, SVF



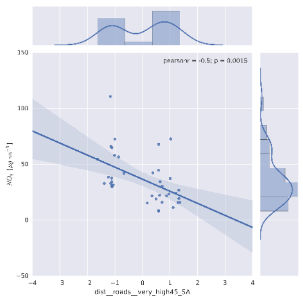
(e) $\rho_s = -0.56$, $p = 2.6e^{-4}$ very high traffic road, distance



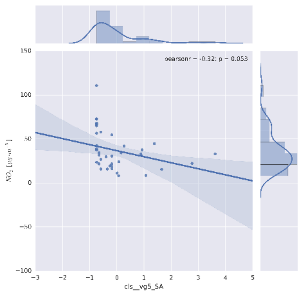
(f) $\rho_s = -0.56$, $p = 2.6e^{-4}$ tunnel, distance



(g) $\rho_s = -0.55$, $p = 3.1e^{-4}$ tunnel, SVF



(h) $\rho_s = -0.51$, $p = 1.2e^{-3}$ very high traffic road, distance, texture 45 pixels



(i) $\rho_s = -0.50$, $p = 1.4e^{-3}$ tree, class, texture 5 pixels

Figure 5.9: Correlation between the transformed geographical features ($\mu = 0$ and $\sigma = 1$) and NO_2 concentration. In each sub-plot are reported: the ρ_s and the relative p, the feature land-use class, the feature type and, if a texture is applied, the number of pixels used to compute the texture.

correlation with the measured average mean concentration of NO_2.

A positive correlation was found with the distance and the SVF of the grass, the presence of very high trafficked roads in the surrounding area, the presence of tunnel and building. Negative correlation was found for the SVF and distance of the roads with high and very high traffic level, distance from the urban tunnels and the presence

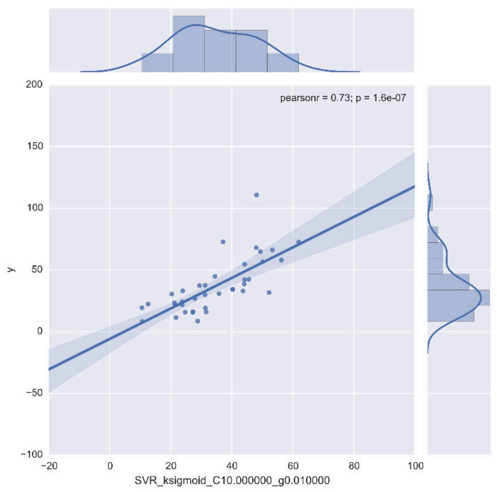


Figure 5.10: Relation between the regressed data using the Support Vector Regression method and measurements of NO_2 concentrations.

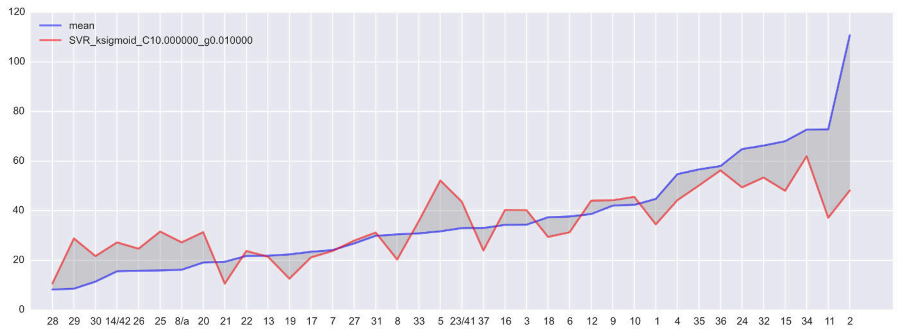


Figure 5.11: Comparison between regressed data and the average of the NO_2 measured concentrations.

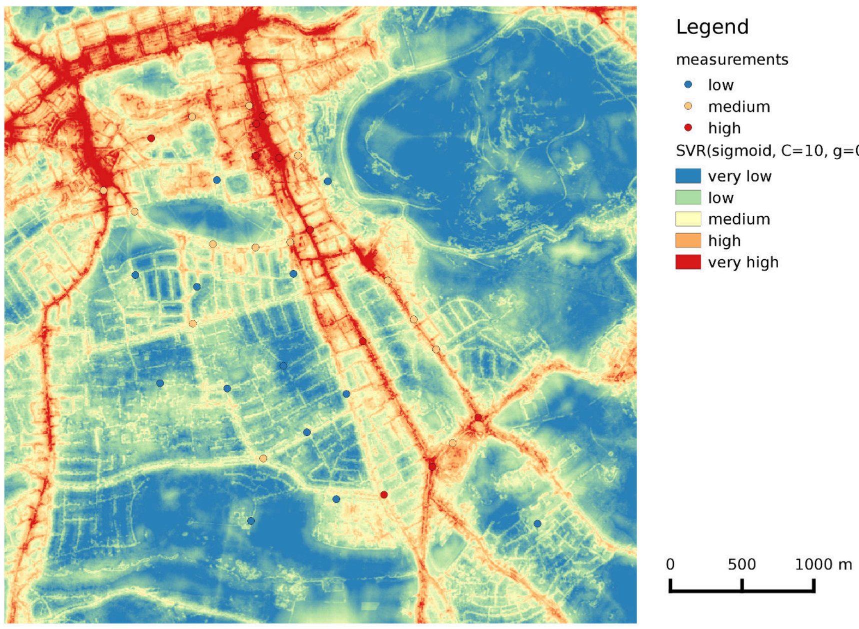


Figure 5.12: Map with assessed NO_2 concentrations using the Support Vector Regression method trained with the average of the measured concentration

of trees. Since correlation does not imply causation, we can not say that grass or vegetation play an active role in the reduction of NO_2 concentrations or simply indicates the absence of NO_2 sources, however these correlations can be used to assess the spatial variability of NO_2 within the urban context.

Geographical factors were reduced using a recursive feature selection and then used as input data to train and test 18 different machine learning algorithms to test the algorithm that provides the best regressed values of the NO_2. The Support Vector Regression algorithm using a Sigmoid kernel provided a mean RMSE of $15.17 \mu g \cdot m^{-3}$ with a strong correlation between measured and estimate concentration data ($\rho_s = 0.78$).

This chapter presents a new methodology that can be easily extended to consider other land-use classes, different features and texture types. Combining these geographical factors with machine-learning algorithms proved to be reliable in assessing the spatial variability of physical phenomena within the urban context.

Chapter 6

Decision support system to assess and compare the effect of different action on the urban air quality

Chapter 7

Regressed scenarios comparison: a methodology for urban planners

Many studies focus their research on characterizing the intra-urban variability of air pollutants; few of them try to provide a methodology to assist policy and decision makers to maximize air quality enhancement within the urban context. This chapter presents a methodology that aims to compare different scenarios, combining the measurement field data and regression modelling to assess the Nitrogen Dioxide (NO_2) concentrations before and after a certain urban change. 37 passive diffusive tubes (PDTs) sampler points were used to train a regression model and link the measured concentration with some geographical factors. Then three different scenarios were developed to reduce the average concentrations of NO_2 in the area surrounding a nursery in the city of Edinburgh (UK). The first sets some trees in front of the nursery building, the second changes the traffic level of the road introducing a new tram line, and the last one couples the tram line with some green spaces using grass and trees along the road. The resulting concentration in all three scenarios is computed and compared to one another.

7.1 Introduction

An increasing number of epidemiological studies have highlighted an association between air pollution exposure and adverse health effects (World Health Organization (WHO), 2014a). Many works focus their research developing and testing tools that predict and model concentration of intra-urban variation of air pollutants. Two main techniques are used in literature: Dispersion Modelling (DM) and Land Use Regression modelling (LUR). As highlighted by Hoogh et al. (2014) “DMs are based on detailed knowledge of the physical, chemical, and fluid dynamical processes in the atmosphere”. DMs require information related to meteorology to model transport and pollutant transformations, as well as a detailed model of the physical, chemical and fluid dynamical processes in the atmosphere. These requirements come at a price of difficulties in collecting or modelling all these input data. LUR, on the other hand, combines monitored data with Geographic Information System (GIS)-based predictor data to build a prediction model, in order to assess the average concentration of air pollutant in a certain urban context. LUR modelling requires a data field campaign to characterize the intra-urban variability and link the geographical factors with the concentration of air pollutants. LUR is good at assessing average values and is less used to estimate short time variations. In comparison with DMs, LUR is easier to use and faster to run, reaching comparable performances (Hoogh et al., 2014).

This chapter presents a LUR modelling to assess the effect of different actions and scenarios on the air quality level at an intra-urban scale. The idea is to provide a methodology that can be applied by urban dwellers, planners and policy-makers to create healthier cities, allowing them to compare the effects of different policies, landscape and urban design. The methodology not only provides an estimate of the air quality level that will be reached after a certain action, but furnishes information about how much and where this reduction occurs and identifies the area affected by a certain urban change. This data-driven methodology can be used to compare different urban and policy options.

7.2 Materials and methods

The geographical features of urban micro-environments were used in combination with machine-learning algorithms for the regression of the average concentration of Nitrogen Dioxide (NO_2) measured, during a winter campaign, using passive diffusive tubes (PDTs) to monitor the city centre of Edinburgh (UK). For more details on the study area, the measurement campaign, the geographical analysis and the regression technique please refer to previous chapters. Levy et al. (2014) highlight that NO_2 seems to be the best proxy available for the other air pollutants, therefore high levels of NO_2 are generally linked with high levels of others air pollutants, providing information about the air quality of the area.

Old adults and babies are more sensitive than others to air pollutant concentrations. To preserve their health it could be useful to identify some of the places where they spent a great portion of the day time. We identified some of these places and we extracted them from the Open Street Map (OSM) project; in particular we considered: nurseries, kindergartens, schools and clinics.

I use a nursery in the south of the city centre of Edinburgh, close to trafficked roads, as a test case to assess the effects that different private and urban decisions and

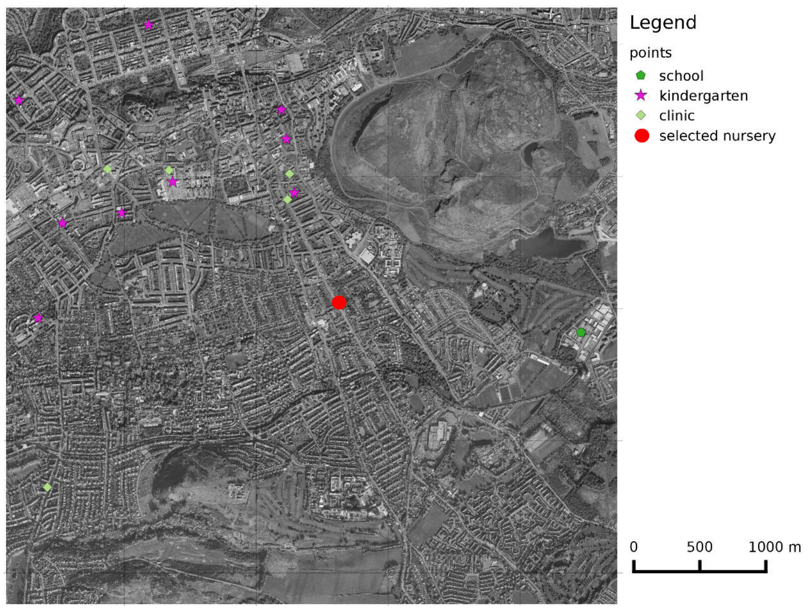


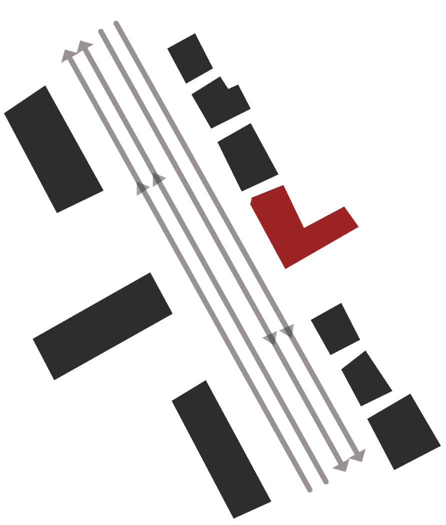
Figure 7.1: The main places, in the city centre of Edinburgh extracted by the OpenStreetMap project, where the weaker part of the population (elderly and young people) spend a relevant part of their day-time.

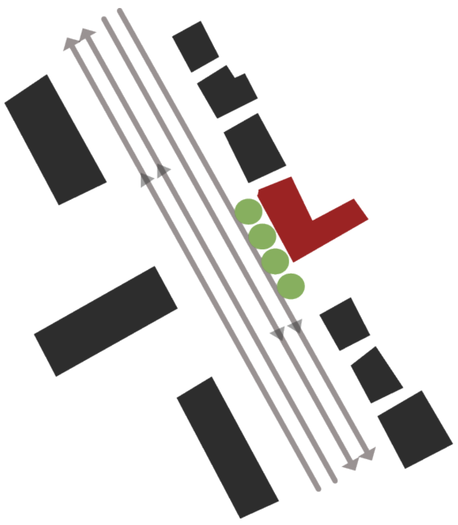
actions could have on air quality and specifically on the NO_2 concentrations on that area.

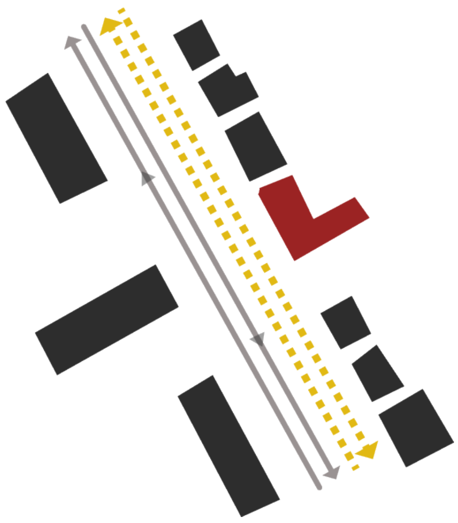
Three different scenarios were developed and compared. In the first scenario we tried to improve the air quality, planting some trees to make a sort of "vegetation wall" in front of the building. In the second scenario we decided to reduce the traffic of the road, introducing an electric tram that lowers the traffic level from a very high to a high trafficked road. In this scenario the road lines were dropped from the current 4 road lines to 2 roads and 2 tram lines. The third scenario hypothesized substituting one tram line with a green area with trees and grass and covering the remaining tram line with grass.

In all these scenarios we basically changed the land-use and the urban micro-environments (see Figure 7.3), so we needed to update all the geographical factors that are influenced by the change. Then we used the previous geographical features and the average NO_2 concentration to train the best regression algorithm, which in our case is the Support Vector Regression (SVR), and applied the regression algorithm using the changed geographical features of the scenarios to assess the impact of different planning choices. Figure 7.4 shows the SVF of grass and tree for the current context and the context in the third scenario.

I computed the difference between the current NO_2 concentrations and all the scenarios to highlight the impact of different actions. Finally, actions have an effect not


(a) Current


(b) Add trees


(c) Add tram

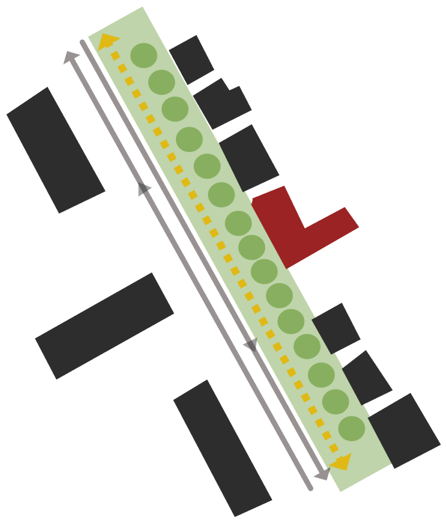

(d) Add trees and grass

Figure 7.2: In Figure 7.2a is reported the current situation; in Figure 7.2b some trees are added in front of the building to make a sort of green wall to protect from pollutant concentrations; Figure 7.2c introduces an electric tram line reducing the traffic road load from very high to high; Figure 7.2d leaves only one tram line introducing grass and trees along the road.

only at the local scale (the nursery) but also at the district and city level.

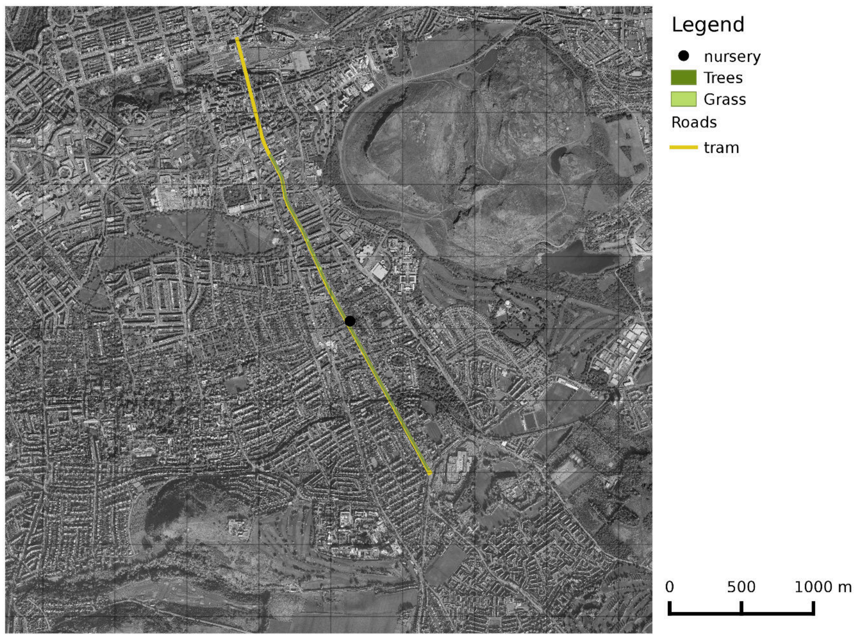
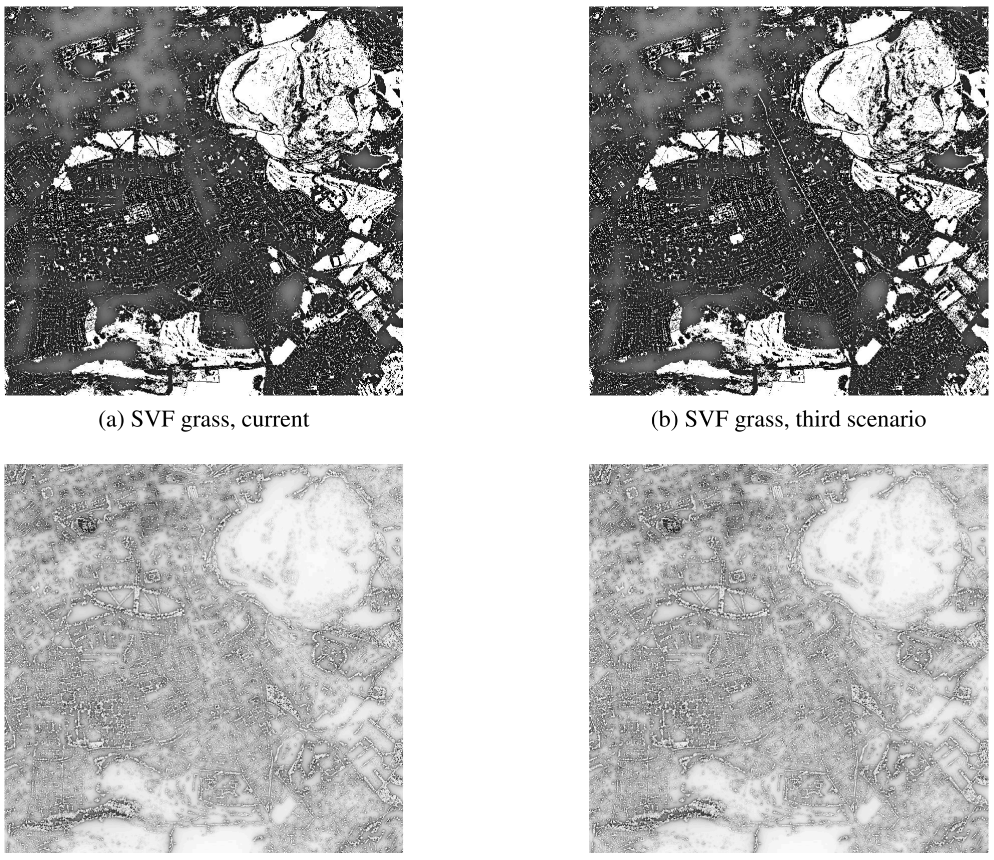


Figure 7.3: Land-use changes concerning the second scenario that introduced a new tram line and the third that added trees and grass areas.

7.3 Results and discussion

In the first scenario, adding trees in front of the nursery building had a negligible effect on air pollutant concentrations (see Figure 7.5b). The maximum difference between the existing configuration and the configuration with the trees was $0.5 \mu g m^{-3}$ (see Figure 7.6a). The second scenario assesses the impact on the air quality if a tram line was used to reduce the traffic load of the road in front of the nursery building. In this case the traffic load was reduced from a very high trafficked road class to high. Introducing this change had an important effect on the nursery area (see Figure 7.5c), with an assessed reduction of NO_2 concentration of about $15 \mu g m^{-3}$ (see Figure 7.6b). The third scenario combines the tram with the introduction of grass and trees, with an assessed reduction of $22 \mu g m^{-3}$; the resulting concentration is shown in Figure 7.5d, and the NO_2 concentration differences are reported in Figure 7.6c. Furthermore, as shown in Figure 7.7, the methodology presented in this chapter is able to assess NO_2 concentrations changes at an urban scale. Since the tram line involves a large part of the city-centre of Edinburgh the effect of this change involves also other parts of the city and not only the surrounding areas of the nursery.


(a) SVF grass, current

(b) SVF grass, third scenario

(c) SVF tree, current

(d) SVF tree, third scenario

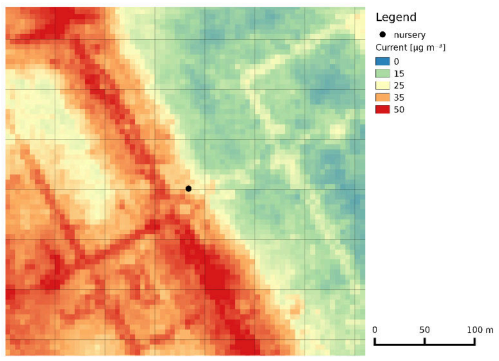
Figure 7.4: The Sky View Factor (SVF) as it is now and as it is in the third scenario with the introduction of the grass and trees element along the trafficked road.

7.4 Conclusions

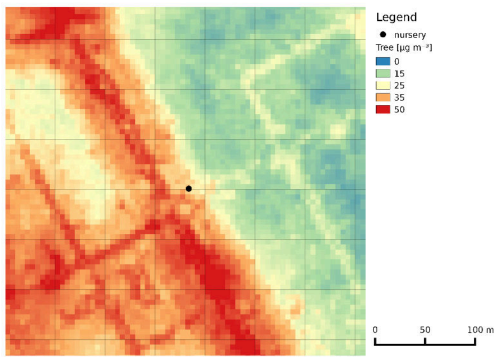
The presented chapter shows a new data-driven methodology to assess the impact that different actions have on average air quality within an urban context. The methodology requires a set of geographical and measurement data to train the machine-learning regression algorithms and link the pollutant's concentrations with geographical factors. Once this link has been established, it is possible to build new scenarios changing the geographical factors and designing the urban landscape to enhance the average air quality level of the city.

Using a greater number of sampled points can increase the measurement density and therefore improve the characterization of the spatial variability of air pollutants. A higher number of measurements can help to make the link between geographical factors and measured concentrations of air pollutants clearer and stronger.

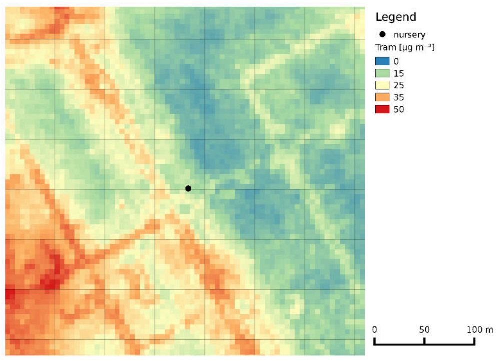
The described methodology is simple to implement and use, since it requires com-



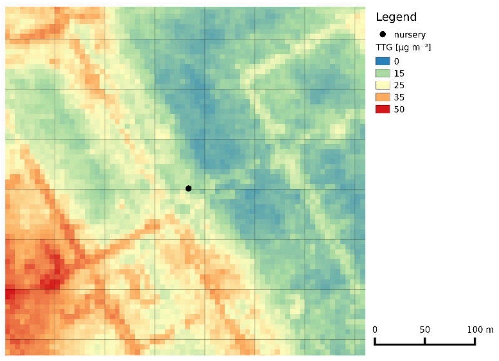
(a) Current



(b) Tree



(c) Tram



(d) TTG

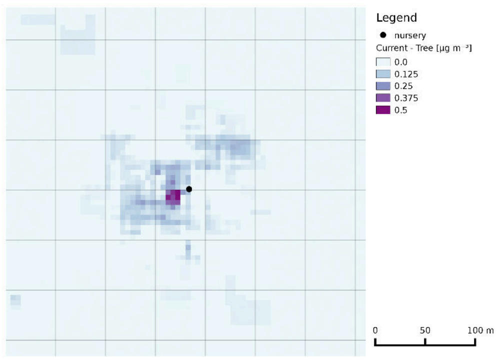
Figure 7.5: Variation of $N0_2$ concentrations [μgm^{-3}], in different scenarios, zoomed on the surrounding areas of the nursery. Figure 7.5a shows the current concentration of NO_2, Figure 7.5b illustrates the concentrations after the introduction of a tree in front the nursery. Figure 7.5c reports the concentration level after reducing the traffic level of the road by introducing a tram, and Figure 7.5d introduces the tram and covers the street with trees and grass (TTG).

mon geographical data as input and a measurement campaign using the Passive Diffusive Tubes (PDTs) to characterize the concentration of air pollutants. Moreover, the methodology is fast to run and allows testing for several urban configurations and comparing the effectiveness of different scenarios and actions to each other.

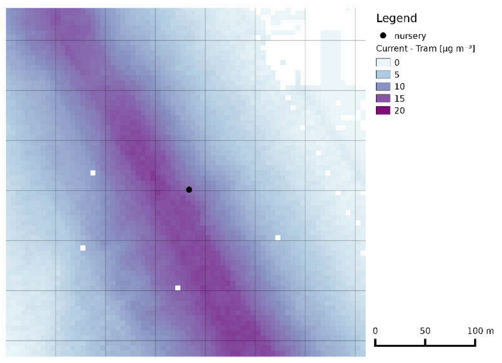
The used measurements provide a value that is representative only of the average concentration value of Nitrogen Dioxide (NO_2). The same methodology can be extended to work with field data characterized by a higher temporal resolution, also integrating meteorological factors into the model (Su et al., 2008a).

The methodology can be improved to consider other factors that can play a role in final air quality in a certain area (e.g. farms, industrial areas, etc.)

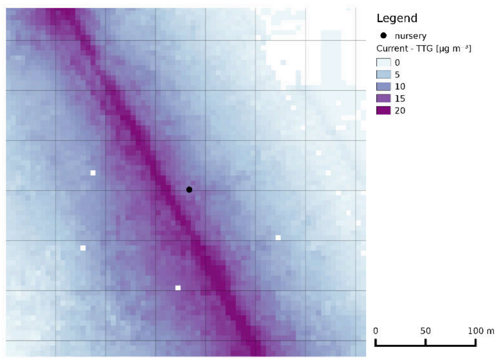
A future development could be to write a tool that, when geographical factors are changed, automatically updates all the derived maps and estimates the concentration



(a) Tree



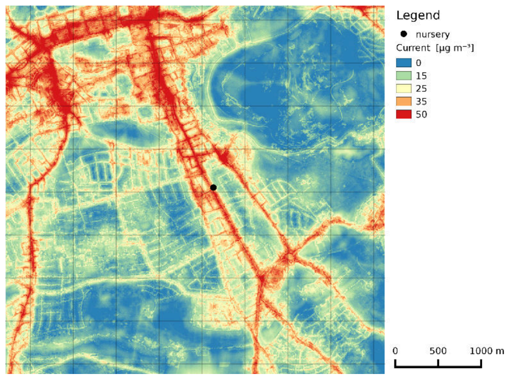
(b) Tram



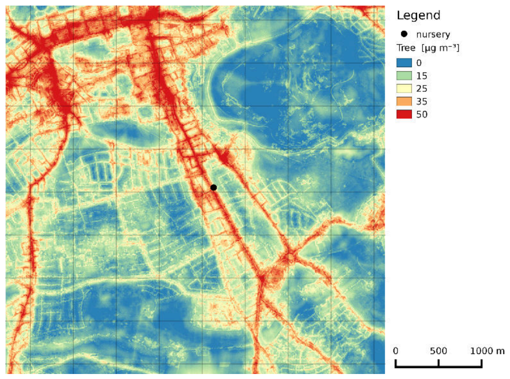
(c) TTG

Figure 7.6: Differences between the current $N0_2$ concentrations [$\mu g m^{-3}$] and the scenarios.

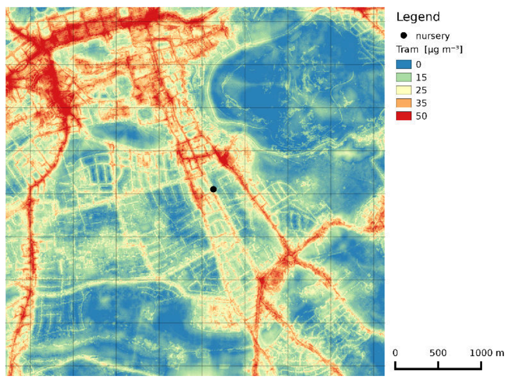
of NO_2 in the study area.



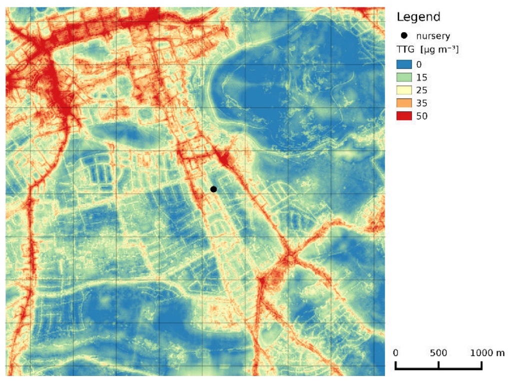
(a) Current



(b) Tree



(c) Tram



(d) TTG

Figure 7.7: Variation of $N0_2$ concentrations [μgm^{-3}], in different scenarios at city scale. Figure 7.7a shows the current concentration of NO_2, Figure 7.7b illustrates the concentrations after the introduction of the tree in front the nursery. Figure 7.7c shows the concentration level after reducing the traffic level of the road due to the introduction of a tram, and Figure 7.7d reports the situation after the introduction of the tram and when the street is covered with trees and grass (TTG).

Chapter 8

Conclusions

A new Geographic Information System model has been developed to characterize the effect of micro-environmental factors that play a role in driving spatial and time variability of air pollutants, using machine-learning algorithms to assess and compare the effects of different policies and scenarios on urban air quality. To achieve these results, we tested low-cost sensor technologies on outside conditions, and the measurements were coupled with Official Measurement Stations (OSM) datasets. The comparison highlights a strong sensitivity of the accuracy to external humidity and temperature conditions and a strong drift on the residues with time.

A new python library has been developed to enlarge the prototyping options to GIS modellers, creating a high-level interface to the C API of GRASS GIS that conveniently opens the GIS data with other scientific communities and libraries. The characterization of urban micro-environments is made through classifying land-use using an Object-Based Image Analysis (OBIA). To assign the proper land-use category, for each object several features were extracted, and these data were used to train, test and select the machine-learning algorithms that provide the higher accuracy.

The School of Chemistry of the University of Edinburgh conducted a six-week campaign using Passive Diffusive Tubes (PDTs) to characterize the spatial variability of Nitrogen Dioxide (NO_2) for the city centre of Edinburgh. I used these measurements to test the correlation between different geographical factors and air quality level. The analysis highlights that, even if with a certain variability, within the measurement weeks, the most polluted points are the most polluted in all the weeks; therefore we can consider the average value of NO_2 quite constant over time and spatially well-defined. The geographical factors were used as an input dataset, to train machine-learning regression algorithms to assess the average concentration value of NO_2. The final step was to design some urban changes and test and compare the impact these could have on average NO_2 concentration within the urban context.

8.1 Pro and cons of the proposed methodology

PROS:	CONS:
• Common and easy to found geographical input data;	• Needs a measurements campaign to collect air quality data;
• Extensible to consider other categories and factors;	• Provides a static average picture of pollutants level;
• Fast to compute and assess the average air quality level;	• Does not consider meteorological conditions;
• Easily quantifies the estimate error;	• The uncertainty increases as the differences between the current situation and the projected scenarios increase.

Table 8.1: Pros and Cons of the methodology described in this manuscript.

The main advantages and disadvantages of the methodology described in this chapter are summarized in Table 8.1. The proposed methodology requires geographical input data that are quite common; if not available at the desired resolution it is possible to extract them following the path as described in Chapter 4, from aerial or satellite images. The geographical factors that were used to assess air quality level are five different classes of roads, grass, trees and buildings, but new factors can be used depending on the context and the pollutants that we aim to assess. The methodology requires, for each change in the current land-use, to re-compute all the raster features used by the regression model before being able to assess the air quality level. The processing time required for these operations is short, especially if compared with dispersion models, since it is only extracting quite a basic feature, such as distance, Sky View Factor or raster texture. Another interesting point of this methodology is that it quantifies the Root Mean Square Error (RMSE) or coefficient of determination (R^2) and therefore provides a value to assess the quality of the final computed scenarios.

The proposed methodology requires the concentration data value of several measured points to test and train the regression model. Due to the low number of Official Measurement Stations (OMS) that are generally available within the urban context, the existing air quality monitoring network is not enough. Therefore a dedicated measurements campaign or network using passive samplers, bio-indicators or low-cost sensors, is required to apply the method. The data collected using passive samplers provides a weekly average value of NO_2 and the meteorological factors seem to play a minor role in influencing the average concentration value. Therefore we did not consider meteorological factors; however, if data with a higher time resolution are available, the methodology can be modified also to use these factors. Since the methodology is based on measured data and not resolving the physics of the phenomena as object of the study, then if the scenarios remain quite similar to the context where the data were collected the error is close to the regression error; if the scenarios apply radical changes, uncertainty on the assessed error will increase considerably. Therefore a dispersion model can be more suitable to assess radically different scenarios.

8.2 Future development

Concerning the Decision Support System, with the availability of data with a higher temporal resolution the methodology can be tested in short-term modelling. As reported by Gulliver and Briggs (2011), the LUR methods are able to model long term spatial variability but are not well-suited to deal with short-term modelling, mainly because the LUR methods do not consider the influence of meteorology. Combining LUR techniques with a dispersion model produces good predictions of monitored concentrations for different time periods and locations (Beelen et al., 2010; Gulliver and Briggs, 2011). Therefore increasing the temporal resolution by taking meteorology into account could help to develop a Decision Support System that can deal with scenarios that change the day's policies, such as, for example, closing a certain road to the traffic during peak hours, etc.

Concerning the libraries and tools developed during the Ph.D., PyGRASS is now included in the last stable release of GRASS GIS and it is being used by other researchers (Grohmann, 2015). The `v.class.ml` module has been also converted into a QGIS plugins (STEM), and it will be tested for processing biological images. The analysis of urban micro-environments and the regression is under testing to assess the Urban Heat Island (UHI) effect in the city centre of Bolzano, Italy within the SINFONIA (FP7) project of the European Research Academy (EURAC).

Appendix

Some small samples of code are provided below to show how modelers, scientists and developers could interact with the PyGRASS library. If the code starts with >>> , this indicates a python interactive section with the terminal. To use and test the PyGRASS extension, the reader needs to install the latest development version of GRASS7. Furthermore, all the following examples use the maps contained in the free available GRASS demonstration dataset *North Carolina* (`http://grass.osgeo.org/sampledata/north_carolina/nc_basic_spm_grass7.tar.gz`).

Modules

Listing 8.1: Direct inputs/outputs.

```
# -*- coding: utf-8 -*-
import subprocess as sub
from pygrass.modules import Module


colout = Module("r.colors.out", map="elevation", stdout_=sub.PIPE)

col = Module("r.colors")
col.inputs.map = "field"
col.inputs.stdin = colout.outputs.stdout
col.inputs.rules = '-'
col.run()
```

Listing 8.2: The syntax is similar to POSIX.

```
# -*- coding: utf-8 -*-
from grass.pygrass.modules import raster as r

# cmd: r.info map=elevation
r.info(map='elevation')

# cmd: r.slope.aspect elevation=elevation slope=slope aspect=aspect --overwrite
r.slope_aspect(elevation='elevation', slope='slope', aspect='aspect', overwrite=True)

# cmd: r.mapcalc 'slope_gt100 = if(slope > 100, slope)' --overwrite
r.mapcalc('slope_gt100 = if(slope > 100, slope)', overwrite=True)
```

Listing 8.3: Backward compatibility.

```
# -*- coding: utf-8 -*-
from grass.pygrass.modules import Module as run_command

run_command('r.info', map='elevation')
```

Listing 8.4: Module as an object.

```
>>> from grass.pygrass.modules import Module
>>> slp = Module('r.slope.aspect')                      # instantiate
>>> slp.name
'r.slope.aspect'
>>> slp.description
'Aspect is calculated counterclockwise from east.'
>>> slp.inputs['elevation']
Parameter <elevation> (required:yes, type:raster, multiple:no)
>>> slp.inputs.elevation = 'elevation'              # set parameter value
>>> slp.inputs.slope = 'slope'
>>> slp.run()                     # finally execute the 'slp' module
```

Listing 8.5: Run and finish.

```
>>> from grass.pygrass.modules import Module
>>> slp = Module('r.slope.aspect')
>>> slp(elevation='elevation', slope='slp', aspect='asp',
...   format='percent', overwrite=True, run_=False) # only set the parameters
>>> slp(elevation='elevation', slope='slp', aspect='asp',
...   format='percent', overwrite=True, finish_=False)   # start the process
>>> slp.popen.wait() # .kill() manage the process
```

Listing 8.6: Stdin.

```
# -*- coding: utf-8 -*-
from pygrass.modules import raster as r

rules = """0 red
1 green
2 blue
end"""

r.colors(map='rtest', rules='-', stdin_=rules)
```

Vectors

Listing 8.7: Vector class.

```
>>> from grass.pygrass.vector import Vector
>>> schools = Vector('schools')
>>> schools.open('r')
>>> schools.title    # Vector attributes: name, organization, person, map_date
'Wake County schools (points map)'
>>> school = schools.next()             # access to the geometry features
>>> school
Point(633649.285674, 221412.944348)
>>> for school in schools:        # or simply iterate through the vector map
...   print school
...
POINT(628787.129283, 223961.620521)
POINT(629900.710134, 222915.798505)
POINT(630686.456623, 224447.772161)
...
>>> schools.close()
```

Listing 8.8: VectorTopo class.

```
>>> from grass.pygrass.vector import VectorTopo
>>> geology = VectorTopo('geology')
>>> geology.open('r')
>>> geology.title    # Vector attributes: name, organization, person, map_date
'North Carolina geology map (polygon map)'
>>> for g in geology:                # or iterate through the vector map
...   print g
...
LINESTRING(...)
LINESTRING(...)
...
POINT(...)
POINT(...)
...
>>> big = [a for a in geology.viter('areas')    # iterate only a geometry type
...     if a.alive() and a.area() >= 10**8]
>>> geology.close()
```

Listing 8.9: Write a new vector map.

```
# -*- coding: utf-8 -*-
from pygrass.vector import VectorTopo

# instantiate and open the tunnels map
tunnels = VectorTopo('tunnels')
tunnels.open('r')

# instantiate a new map
new = VectorTopo('newvect')

# open a new vector map defining the column names and types, with:
new.open('w', tabcols=[(u'cat', int),
                       (u'tunnel', int),
                       (u'length', float)])

for tunnel in tunnels:
    # extract the first and the last point of the tunnel
    first, last = tunnel[0], tunnel[-1]
    # compute the tunnel length and divide the length for each point
    length = tunnel.length() / 2.
    #.write(geom_feature, attributes)
    new.write(first, (tunnel.cat, length))
    new.write(last, (tunnel.cat, length))
# then close all
new.close()
tunnels.close()
```

Rasters

Listing 8.10: RasterRow class.

```
>>> from grass.pygrass.raster import RasterRow
>>> elev = RasterRow('elevation')
>>> elev.exist()                           # check if the map exist
True
>>> elev.name                             # return the raster name
'elevation'
>>> elev.mapset                          # return the raster mapset
'PERMANENT'
>>> elev.open('r')                               # open in read mode
>>> elev.is_open()                        # check if the map is open
True
>>> elev.range                             # return the map range
(55.578792572021484, 156.32986450195312)
>>> for row in elev[:5]:                         # get the first 5 rows
...   print(row[:3])             # show the first 3 columns of each row
...
```

```
[ 141.99613953 141.27848816 141.37904358]
[ 142.90461731 142.39450073 142.68611145]
[ 143.81854248 143.54707336 143.83972168]
[ 144.56524658 144.58493042 144.86477661]
[ 144.99488831 145.22894287 145.57142639]
>>> elev.close()
```

Listing 8.11: Write a new raster map.

```
# -*- coding: utf-8 -*-
from grass.pygrass.raster import RasterRow

# instantiate raster objects
old = RasterRow('elevation')
new = RasterRow('mapname1')

# open the maps
old.open('r')
new.open('w', mtype=old.mtype, overwrite=True)

# start a cycle
for row in old:
    new.put_row(row > 100)                                          # write row

# close the maps
new.close()
old.close()
```

GIS/GRASS

Listing 8.12: Write a new raster map.

```
>>> from grass.pygrass.gis import Location, Mapset
>>> location = Location()
>>> location.mapsets()
['PERMANENT', 'user1']
>>> permanent = Mapset('PERMANENT')
>>> permanent.glist('rast', pattern='elev*')     # return a list with rasters
['elevation_shade', 'elevation']
```

Listing 8.13: Write a new raster map.

```
>>> from grass.pygrass.gis.region import Region
>>> region = Region()
>>> region.north
258500.0
>>> region.south
185000.0
>>> region.rows
7350
>>> region.nsres
10.0
>>> print region
projection: 99 (Lambert Conformal Conic)
zone:     0
```

```
datum:    nad83
ellipsoid: a=6378137 es=0.006694380022900787
north:    258500
south:    185000
west:     596670
east:     678330
nsres:    10
ewres:    10
rows:     7350
cols:     8166
cells:    60020100
```

Benchmark

Listing 8.14: Write a new vector points map adding the raster value in the attribute table using PyGRASS.

```
# -*- coding: utf-8 -*-
import numpy as np
from grass.pygrass.gis.region import Region
from grass.pygrass.vector import VectorTopo
from grass.pygrass.raster import RasterRow
from grass.pygrass.functions import coor2pixel


def sample(vect_in_name, rast_in_name):
    """sample('point00', 'field')"""
    # instantiate the object maps
    vect_in = VectorTopo(vect_in_name)
    rast_in = RasterRow(rast_in_name)
    vect_out = VectorTopo('test_' + vect_in_name)

    # define the columns of the attribute table of the new vector map
    columns = [(u'cat',       'INTEGER PRIMARY KEY'),
               (rast_in_name,  'DOUBLE')]

    # open the maps
    vect_in.open('r')
    rast_in.open('r')
    vect_out.open('w', tab_cols=columns, link_driver='sqlite')

    # get the current region
    region = Region()

    # initialize the counter
    counter = 0
    data = []
    for pnt in vect_in.viter('points'):
        counter += 1
        # transform the spatial coordinates in row and col value
        x, y = coor2pixel(pnt.coords(), region)
        value = rast_in[int(x)][int(y)]
        data.append((counter, None if np.isnan(value) else float(value)))
        # write the geometry features
        vect_out.write(pnt)
    # write the attributes
    vect_out.table.insert(data, many=True)
    vect_out.table.conn.commit()

    # close the maps
    vect_in.close()
    rast_in.close()
    vect_out.close()
```

Listing 8.15: Write a new vector points map adding the raster value in the attribute table using C.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

#include <grass/gis.h>
#include <grass/raster.h>
#include <grass/glocale.h>
#include <grass/dbmi.h>
#include <grass/vector.h>


static void sample(char *input, char *rast, char *column, char *output) {

        struct Cell_head window;
        struct Map_info In, Out;
        int fdrast;
        DCELL value;
        G_get_window(&window);
        int line;
        int type;
        struct field_info *Fi;
        dbDriver *Driver;
        char buf[2000];
        dbString sql;
        DCELL *dcell_buf;

        /* Open input */
        Vect_set_open_level(2);
        Vect_open_old2(&In, input, "", "1");
        fdrast = Rast_open_old(rast, "");

        /* Open output */
        Vect_open_new(&Out, output, 0);
        Vect_hist_copy(&In, &Out);
        Vect_hist_command(&Out);

        /* Create table */
        Fi = Vect_default_field_info(&Out, 1, NULL, GV_1TABLE);
        Vect_map_add_dblink(&Out, Fi->number, Fi->name, Fi->table, Fi->key,
                        Fi->database, Fi->driver);
        Driver = db_start_driver_open_database(Fi->driver,
                Vect_subst_var(Fi->database, &Out));
    sprintf(buf, "create table %s ( cat integer, rast_val double precision)",
                Fi->table);
    db_init_string(&sql);
    db_set_string(&sql, buf);
    db_execute_immediate(Driver, &sql);
    db_create_index2(Driver, Fi->table, Fi->key);
    db_grant_on_table(Driver, Fi->table, DB_PRIV_SELECT, DB_GROUP | DB_PUBLIC);

    /* Sample the raster map with vector points */
    struct line_pnts *Points = Vect_new_line_struct();
    struct line_cats *Cats = Vect_new_cats_struct();
    int nlines = Vect_get_num_lines(&In);
    int count = 0;

    dcell_buf = Rast_allocate_buf(DCELL_TYPE);

    db_begin_transaction(Driver);

    for (line = 1; line <= nlines; line++) {
            type = Vect_read_line(&In, Points, Cats, line);

            if (!(type & GV_POINT))
                continue;

            if(G_point_in_region(Points->x[0], Points->y[0]) == 0)
                    continue;

            if (Rast_is_d_null_value(&value))
                continue;

            int row = Rast_northing_to_row(Points->y[0], &window);
            int col = Rast_easting_to_col(Points->x[0], &window);

            Rast_get_d_row(fdrast, dcell_buf, row);
            value = dcell_buf[col];

            /* Write value into the vector table */
            count++;
            Vect_reset_cats(Cats);
            Vect_cat_set(Cats, 1, count);
            Vect_write_line(&Out, GV_POINT, Points, Cats);

            sprintf(buf, "INSERT INTO %s VALUES ( %d, %e ); ", Fi->table, count,
                        (double)value);
            db_set_string(&sql, buf);
            db_execute_immediate(Driver, &sql);
    }

db_commit_transaction(Driver);
    db_close_database_shutdown_driver(Driver);

    Rast_close(fdrast);
    Vect_close(&In);
    Vect_build(&Out);
```

```
        Vect_close(&Out);

        exit(EXIT_SUCCESS);
}
```

Listing 8.16: Compute using the RasterRow class.

```
# -*- coding: utf-8 -*-
from grass.pygrass.raster import RasterRow

def ifcondition(mapname0, mapname1):
    # instantiate raster objects
    old = RasterRow(mapname0)
    new = RasterRow(mapname1)
    # open the maps
    old.open('r')
    new.open('w', mtype=old.mtype, overwrite=True)
    # start a cycle
    for row in old:
        true = row > 50
        new.put_row(row * true)
    # close the maps
    new.close()
    old.close()
```

Listing 8.17: Call the *r.mapcalc* module from Python

```
sub.Popen("r.mapcalc expression='test_mapcalc=if(field>50,field,0)' --o", shell=True).wait()
```

Bibliography

Albanese, D. and R. Visintainer (2012). “mlpy: Machine Learning Python”. In: *arXiv preprint arXiv: . . .* Pp. 1–4.

Alemdar, H. and C. Ersoy (2010). “Wireless sensor networks for healthcare: A survey”. In: *Computer Networks* 54.15, pp. 2688–2710.

Alioscha-Perez, M. and H. Sahli (2014). “Efficient Learning of Spatial Patterns with Multi-Scale Conditional Random Fields for Region-Based Classification”. In: *Remote Sensing* 6.8, pp. 6727–6764. ISSN: 2072-4292.

Allwine, K., F. Rutz, W. Shaw, J. Rishel, B. Fritz, E. Chapman, B. Hoopes, and B. Seiple (2006). “DUSTRAN 1.0 user’s guide: a GIS-based atmospheric dust dispersion modelling system”. In: *Technical report* PNNL-16055.

Arain, M. A., R. Blair, N. Finkelstein, J. R. Brook, T. Sahsuvaroglu, B. Beckerman, L. Zhang, and M. Jerrett (2007). “The use of wind fields in a land use regression model to predict air pollution concentrations for health exposure studies”. In: *Atmospheric Environment* 41.16, pp. 3453–3464.

Baume, O., J. O. Skø ien, G. B. M. Heuvelink, E. J. Pebesma, and S. J. Melles (2011). “A geostatistical approach to data harmonization - Application to radioactivity exposure data”. In: *International Journal of Applied Earth Observation and Geoinformation* 13.3, pp. 409–419.

Beelen, R., G. Hoek, E. Pebesma, D. Vienneau, K. de Hoogh, and D. J. Briggs (2009). “Mapping of background air pollution at a fine spatial scale across the European Union”. In: *Science of The Total Environment* 407.6, pp. 1852–1867.

Beelen, R., M. Voogt, J. Duyzer, P. Zandveld, and G. Hoek (2010). “Comparison of the performances of land use regression modelling and dispersion modelling in estimating small-scale variations in long-term air pollution concentrations in a Dutch urban area”. In: *Atmospheric Environment* 44.36, pp. 4614–4621.

Behling, R., M. Bochow, S. Foerster, S. Roessner, and H. Kaufmann (2015). “Automated GIS-based derivation of urban ecological indicators using hyperspectral remote sensing and height information”. In: *Ecological Indicators* 48, pp. 218–234. ISSN: 1470160X.

Berkowicz, R., M. Ketzel, S. S. Jensen, M. Hvidberg, and O. Raaschou-Nielsen (2008). “Evaluation and application of OSPM for traffic pollution assessment for a large number of street locations”. In: *Environmental Modelling & Software* 23.3, pp. 296–303.

Borrell, B. (2011). “Epidemiology: Every bite you take”. In: *Nature* 470, pp. 320–322.

Briggs, D. J., C. de Hoogh, J. Gulliver, J. Wills, P. Elliott, S. Kingham, and K. Smallbone (2000). “A regression-based method for mapping traffic-related air pollution: application and testing in four contrasting urban environments”. In: *The Science of The Total Environment* 253.1-3, pp. 151–167.

British Ordnance Survey (2012). *Orthophoto RGB Edinmburgh.*

Burgess, S., M. Kranz, N. Turner, R. Cardell-Oliver, and T. Dawson (2010). “Harnessing wireless sensor technologies to advance forest ecology and agricultural research”. In: *Agricultural and Forest Meteorology* 150.1, pp. 30–37.

Busato, F., R. Lazzarin, and M. Noro (2013). "Three years of study of the Urban Heat Island in Padua: Experimental results". In: *Sustainable Cities and Society*, pp. 1–8. ISSN: 22106707.

Can, a, M Rademaker, T Van Renterghem, V Mishra, M Van Poppel, a Touhafi, J Theunis, B De Baets, and D Botteldooren (2011). "Correlation analysis of noise and ultrafine particle counts in a street canyon." In: *The Science of the total environment* 409.3, pp. 564–72.

Cao, X., J. Chen, Y. Zhang, and Y. Sun (2008). "Development of an integrated wireless sensor network micro-environmental monitoring system". In: *ISA Transactions* 47.3, pp. 247 –255.

Cape, J. N. (2009). "The Use of Passive Diffusion Tubes for Measuring Concentrations of Nitrogen Dioxide in Air". In: *Critical Reviews in Analytical Chemistry* 39.4, pp. 289–310. ISSN: 1040-8347.

Carnevale, C., G. Finzi, E. Pisoni, M. Volta, G. Guariso, R. Gianfreda, G. Maffeis, P. Thunis, L. White, and G. Triacchini (2012a). "An integrated assessment tool to define effective air quality policies at regional scale". In: *Environmental Modelling & Software* 38, pp. 306–315.

Carnevale, C., G. Finzi, E. Pisoni, M. Volta, and F. Wagner (2012b). "Defining a nonlinear control problem to reduce particulate matter population exposure". In: *Atmospheric Environment* 55, pp. 410–416. ISSN: 13522310.

Carruthers, D., H. Edmunds, A. Lester, C. McHugh, and R. Singles (2000). "Use and validation of ADMS-Urban in contrasting urban and industrial locations". In: *International Journal of Environment and Pollution* 14.1-6, pp. 364–374.

Chen, A., L. Yao, R. Sun, and L. Chen (2014). "How many metrics are required to identify the effects of the landscape pattern on land surface temperature?" In: *Ecological Indicators* 45, pp. 424–433. ISSN: 1470160X.

Ciolli, M, M De Franceschi, R Rea, A Vitti, D Zardi, and P Zatelli (2004). "Development and application of 2D and 3D GRASS modules for simulation of thermally driven slope winds". In: *TRANSACTIONS IN GIS* 8.2, pp. 191–209.

Clewley, D., P. Bunting, J. Shepherd, S. Gillingham, N. Flood, J. Dymond, R. Lucas, J. Armston, and M. Moghaddam (2014). "A Python-Based Open Source System for Geographic Object-Based Image Analysis (GEOBIA) Utilizing Raster Attribute Tables". In: *Remote Sensing* 6.7, pp. 6111–6135. ISSN: 2072-4292.

Coneus, K. and C. K. Spiess (2012). "Pollution exposure and child health: Evidence for infants and toddlers in Germany". In: *Journal of Health Economics* 31.1, pp. 180–196.

Crouse, D. L., M. S. Goldberg, and N. a. Ross (2009). "A prediction-based approach to modelling temporal and spatial variability of traffic-related air pollution in Montreal, Canada". In: *Atmospheric Environment* 43.32, pp. 5075–5084.

Curtis, L., W. Rea, P. Smith-Willis, E. Fenyves, and Y. Pan (2006). "Adverse health effects of outdoor air pollutants". In: *Environment International* 32.6, pp. 815–830. ISSN: 0160-4120.

Dadvand, P., J. Rankin, S. Rushton, and T. Pless-Mulloli (2011). "Ambient air pollution and congenital heart disease: A register-based study". In: *Environmental Research* 111.3, pp. 435–441.

De Vito, S., G. Fattoruso, R. Liguoro, A. Oliviero, E. Massera, C. Sansone, V. Casola, and G. Di Francia (2011). "Cooperative 3D Air Quality Assessment with Wireless Chemical Sensing Networks". In: *Procedia Engineering* 25, pp. 84–87.

Dons, E., L. Int Panis, M. Van Poppel, J. Theunis, H. Willems, R. Torfs, and G. Wets (2011). "Impact of time-activity patterns on personal exposure to black carbon". In: *Atmospheric Environment* 45.21, pp. 3594–3602.

Dons, E., M. Van Poppel, B. Kochan, G. Wets, and L. Int Panis (2013). "Modeling temporal and spatial variability of traffic-related air pollution: Hourly land use regression models for black carbon". In: *Atmospheric Environment* 74, pp. 237–246.

Dronova, I., P. Gong, N. E. Clinton, L. Wang, W. Fu, S. Qi, and Y. Liu (2012). “Landscape analysis of wetland plant functional types: The effects of image segmentation scale, vegetation classes and classification methods”. In: *Remote Sensing of Environment* 127, pp. 357–369. ISSN: 00344257.

Dugord, P.-A., S. Lauf, C. Schuster, and B. Kleinschmit (2014). “Land use patterns, temperature distribution, and potential heat stress risk - The case study Berlin, Germany”. In: *Computers, Environment and Urban Systems* 48, pp. 86–98. ISSN: 01989715.

Duro, D. C., S. E. Franklin, and M. G. Dubé (2012). “A comparison of pixel-based and object-based image analysis with selected machine learning algorithms for the classification of agricultural landscapes using SPOT-5 HRG imagery”. In: *Remote Sensing of Environment* 118, pp. 259–272. ISSN: 00344257.

Eeftens, M., R. Beelen, K. de Hoogh, T. Bellander, G. Cesaroni, M. Cirach, C. Declercq, A. Dedele, E. Dons, A. de Nazelle, K. Dimakopoulou, K. Eriksen, G. Falq, P. Fischer, C. Galassi, R. Gražulevičiene, J. Heinrich, B. Hoffmann, M. Jerrett, D. Keidel, M. Korek, T. Lanki, S. Lindley, C. Madsen, A. Mölter, G. Nádor, M. Nieuwenhuijsen, M. Nonnemacher, X. Pedeli, O. Raaschou-Nielsen, E. Patelarou, U. Quass, A. Ranzi, C. Schindler, M. Stempfelet, E. Stephanou, D. Sugiri, M.-Y. Tsai, T. Yli-Tuomi, M. J. Varró, D. Vienneau, S. von Klot, K. Wolf, B. Brunekreef, and G. Hoek (2012). “Development of Land Use Regression Models for PM2.5, PM2.5 Absorbance, PM10 and PMcoarse in 20 European Study Areas, Results of the ESCAPE Project”. In: *Environmental Science & Technology* 46.20, pp. 11195–11205.

Elbir, T., N. Mangir, M. Kara, S. Simsir, T. Eren, and S. Ozdemir (2010). “Development of a GIS-based decision support system for urban air quality management in the city of Istanbul”. In: *Atmospheric Environment* 44.4, pp. 441–454.

EPA, U. (2010). “Integrated Science Assessment for Carbon Monoxide”. In: *National Center for Environmental Assessment, Office of Research and Development* 019B.EPA/600/R-09/019B.

Finardi, S., R. De Maria, A. D’Allura, C. Cascone, G. Calori, and F. Lollobrigida (2008). “A deterministic air quality forecasting system for Torino urban area, Italy”. In: *Environmental Modelling & Software* 23.3, pp. 344–355. ISSN: 13648152.

Flammini, A., P. Ferrari, D. Marioli, E. Sisinni, and A. Taroni (2009). “Wired and wireless sensor networks for industrial applications”. In: *Microelectronics Journal* 40.9, pp. 1322–1336.

Foody, G. (2008). “GIS: biodiversity applications”. In: *Progress in Physical Geography* 32.2, pp. 223–235. ISSN: 0309-1333.

Gasiewicz, T. (2010). “The Exposome: A powerful approach for evaluating environmental exposures and their influences on human disease”. In: *Emerging science for environmental health decisions - Workshop*. Nas Building, 2100c street, Washington DC.

Gidhagen, L., G. Omstedt, G. Pershagen, S. Willers, and T. Bellander (2013). “High-resolution modeling of residential outdoor particulate levels in Sweden.” In: *Journal of exposure science & environmental epidemiology* 23.3, pp. 306–14. ISSN: 1559-064X.

Gitelson, A. and M. Merzlyak (1996). “Signature analysis of leaf reflectance spectra: Algorithm development for remote sensing of chlorophyll”. In: *Journal of Plant Physiology* 148.4, pp. 494–500.

Gomez-Mejiba, S. E., Z Zhai, H Akram, Q. N. Pye, K Hensley, B. T. Kurien, R. H. Scofield, and D. C. Ramirez (2009). “Inhalation of environmental stressors & chronic inflammation: autoimmunity and neurodegeneration.” In: *Mutat Res* 674.1-2, pp. 62–72.

González, A., A. Donnelly, M. Jones, N. Chrysoulakis, and M. Lopes (2013). “A decision-support system for sustainable urban metabolism in Europe”. In: *Environmental Impact Assessment Review* 38, pp. 109–119. ISSN: 01959255.

Grohmann, C. H. (2015). “Effects of spatial resolution on slope and aspect derivation for regional-scale analysis”. In: *Computers & Geosciences* 77.0, pp. 111–117. ISSN: 0098-3004.

Gulliver, J. and D. Briggs (2011). “STEMS-Air: a simple GIS-based air pollution dispersion model for city-wide exposure assessment.” In: *The Science of the total environment* 409.12, pp. 2419–29.

Guo, D., R. Guo, and C. Thiart (2007). “Predicting air pollution using fuzzy membership grade Kriging”. In: *Computers, Environment and Urban Systems* 31.1, pp. 33–51.

Hamzelou, J. (2011). “Welcome to the exposome”. In: *The New Scientist* 208.2792, pp. 6–7.

Hoek, G., R. Beelen, K. de Hoogh, D. Vienneau, J. Gulliver, P. Fischer, and D. Briggs (2008). “A review of land-use regression models to assess spatial variation of outdoor air pollution”. In: *Atmospheric Environment* 42.33, pp. 7561–7578.

Hofierka, J. and M. Zlocha (2012). “A New 3-D Solar Radiation Model for 3-D City Models”. In: *Transactions in GIS* 16.5, pp. 681–690. ISSN: 1467-9671.

Hoogh, K. de, M. Korek, D. Vienneau, M. Keuken, J. Kukkonen, M. J. Nieuwenhuijsen, C. Badaloni, R. Beelen, A. Bolignano, G. Cesaroni, M. C. Pradas, J. Cyrys, J. Douros, M. Eeftens, F. Forastiere, B. Forsberg, K. Fuks, U. Gehring, A. Gryparis, J. Gulliver, A. L. Hansell, B. Hoffmann, C. Johansson, S. Jonkers, L. Kangas, K. Katsouyanni, N. Künzli, T. Lanki, M. Memmesheimer, N. Moussiopoulos, L. Modig, G. Pershagen, N. Probst-Hensch, C. Schindler, T. Schikowski, D. Sugiri, O. Teixidó, M.-Y. Tsai, T. Yli-Tuomi, B. Brunekreef, G. Hoek, and T. Bellander (2014). “Comparing land use regression and dispersion modelling to assess residential exposure to ambient air pollution for epidemiological studies”. In: *Environment International* 73, pp. 382–392. ISSN: 01604120.

Hu, S.-C., Y.-C. Wang, C.-Y. Huang, and Y.-C. Tseng (2011). “Measuring air quality in city areas by vehicular wireless sensor networks”. In: *Journal of Systems and Software* 84.11, pp. 2005–2012. ISSN: 01641212.

Hunter, J. D. (2007). “Matplotlib: A 2D Graphics Environment”. In: *Computing in Science & Engineering* 9.9, pp. 90–95.

Huth, J., C. Kuenzer, T. Wehrmann, S. Gebhardt, V. Tuan, and S. Dech (2012). “Land cover and land use classification with TWOPAC: Towards automated processing for pixel- and object-based image classification.” In: *Remote Sensing* 4, pp. 2530–2553.

Inglada, J. and E Christophe (2009). “The Orfeo Toolbox Remote Sensing Image Processing Software”. In: *IEEE International Geoscience and Remote Sensing Symposium*. Cape Town, South Africa, 12-17 July 2009, pp. IV–733–IV–736.

InterImage (2014). *InterImage User Manual 1.41*. Laboratório de Visão Computacional : Rio de Janeiro, Brazil.

Janssen, N., P. Fischer, M. Marra, C. Ameling, and F. Cassee (2013). “Short-term effects of PM2.5, PM10 and PM2.5-10 on daily mortality in the Netherlands”. In: *Science of The Total Environment* 463-464, pp. 20–26.

Janssen, S., G. Dumont, F. Fierens, and C. Mensink (2008). “Spatial interpolation of air pollution measurements using CORINE land cover data”. In: *Atmospheric Environment* 42.20, pp. 4884–4903.

Jephcote, C. and H. Chen (2012). “Environmental injustices of children’s exposure to air pollution from road-transport within the model British multicultural city of Leicester: 2000-09”. In: *Science of The Total Environment* 414.0, pp. 140–151.

Jin, S., J. Guo, S. Wheeler, L. Kan, and S. Che (2014). “Evaluation of Impacts of Trees on PM2.5 Dispersion in Urban Streets”. In: *Atmospheric Environment* 99, pp. 277–287. ISSN: 13522310.

Johnson, M., V. Isakov, J. S. Touma, S. Mukerjee, and H. Özkaynak (2010). “Evaluation of land-use regression models used to predict air quality concentrations in an urban area”. In: *Atmospheric Environment* 44.30, pp. 3660–3668.

Jones, E., T. Oliphant, P. Peterson, and S. Community (2013). “SciPy: Open Source Scientific Tools for Python”. In:

Jones, E., T. Oliphant, P. Peterson, et al. (2001-2015). *SciPy: Open source scientific tools for Python*. [Online; accessed 2014-07-23].

Käffer, M. I., A. T. Lemos, M. A. Apel, J. V. Rocha, S. M. D. A. Martins, and V. M. F. a. Vargas (2012). “Use of bioindicators to evaluate air quality and genotoxic compounds in an urban environment in Southern Brazil.” In: *Environmental pollution (Barking, Essex : 1987)* 163, pp. 24–31.

Kardel, F., K. Wuyts, B. Maher, R. Hansard, and R. Samson (2011). “Leaf saturation isothermal remanent magnetization (SIRM) as a proxy for particulate matter monitoring: Inter-species differences and in-season variation”. In: *Atmospheric Environment* 45.29, pp. 5164–5171.

Kardel, F., K. Wuyts, M. Babanezhad, T. Wuytack, S. Adriaenssens, and R. Samson (2012). “Tree leaf wettability as passive bio-indicator of urban habitat quality”. In: *Environmental and Experimental Botany* 75, pp. 277–285.

Kaur, S., M. Nieuwenhuijsen, and R. Colvile (2007). “Fine particulate matter and carbon monoxide exposure concentrations in urban street transport microenvironments”. In: *Atmospheric Environment* 41.23, pp. 4781–4810. ISSN: 13522310.

Kesarkar, A. P., M. Dalvi, A. Kaginalkar, and A. Ojha (2007). “Coupling of the Weather Research and Forecasting Model with AERMOD for pollutant dispersion modeling. A case study for PM10 dispersion over Pune, India”. In: *Atmospheric Environment* 41.9, pp. 1976–1988.

Kotthaus, S. and C. Grimmond (2013). “Energy exchange in a dense urban environment - Part I: Temporal variability of long-term observations in central London”. In: *Urban Climate*. ISSN: 22120955.

Lehmann, I., J. Mathey, S. Röß ler, A. Bräuer, and V. Goldberg (2014). “Urban vegetation structure types as a methodological approach for identifying ecosystem services - Application to the analysis of micro-climatic effects”. In: *Ecological Indicators* 42, pp. 58–72. ISSN: 1470160X.

Levy, I., C. Mihele, and G. Lu (2014). “Evaluating multipollutant exposure and urban air quality: pollutant interrelationships, neighborhood variability, and nitrogen dioxide as a proxy pollutant”. In: *Environmental health . . .* 122.1, pp. 65–72.

Li, X., S. W. Myint, Y. Zhang, C. Galletti, X. Zhang, and B. L. Turner (2014). “Object-based land-cover classification for metropolitan Phoenix, Arizona, using aerial photography”. In: *International Journal of Applied Earth Observation and Geoinformation* 33, pp. 321–330. ISSN: 03032434.

Li, X., L. Di, W. Han, P. Zhao, and U. Dadi (2010). “Sharing geoscience algorithms in a Web service-oriented environment (GRASS GIS example)”. In: *Computers & Geosciences* 36.8, pp. 1060–1068. ISSN: 00983004.

Liu, H. and Y. Shen (2014). “The Impact of Green Space Changes on Air Pollution and Microclimates: A Case Study of the Taipei Metropolitan Area”. In: *Sustainability* 6, pp. 8827–8855.

Llop, E., P. Pinho, P. Matos, M. J. a. Pereira, and C. Branquinho (2012). “The use of lichen functional groups as indicators of air quality in a Mediterranean urban environment”. In: *Ecological Indicators* 13.1, pp. 215–221. ISSN: 1470160X.

Madsen, C., K. C. Carlsen, G. Hoek, B. Oftedal, P. Nafstad, K. Meliefste, R. Jacobsen, W. Nystad, and B. Brunekreef (2007). “Modeling the intra-urban variability of outdoor traffic pollution in Oslo, Norway–A GA2LEN project”. In: *Atmospheric Environment* 41.35, pp. 7500–7511.

Marchetti, A., C. Piccini, S. Santucci, I. Chiuchiarelli, and R. Francaviglia (2011). “Simulation of soil types in Teramo province (Central Italy) with terrain parameters and remote sensing data”. In: *Catena* 85.3, pp. 267–273.

Mavroulidou, M., S. J. Hughes, and E. E. Hellawell (2004). “A qualitative tool combining an interaction matrix and a GIS to map vulnerability to traffic induced air pollution.” In: *Journal of environmental management* 70.4, pp. 283–9.

Meijer, M., J. Röhl, K. Bloomfield, and U. Grittner (2012). "Do neighborhoods affect individual mortality? A systematic review and meta-analysis of multilevel studies." In: *Social science & medicine (1982)* 74.8, pp. 1204–12.

Merbitz, H., M. Buttstädt, S. Michael, W. Dott, and C. Schneider (2012). "GIS-based identification of spatial variables enhancing heat and poor air quality in urban areas". In: *Applied Geography* 33.0, pp. 94–106. ISSN: 0143-6228.

Millman, K. J. and M. Aivazis (2011). "Python for Scientists and Engineers". In: *Computing in Science & Engineering* 13.13, pp. 9–12.

Mölter, A, S Lindley, F de Vocht, A Simpson, and R Agius (2010). "Modelling air pollution for epidemiologic research – Part I: A novel approach combining land use regression and air dispersion". In: *Science of The Total Environment* 408.23, pp. 5862–5869. ISSN: 0048-9697.

Mölter, A., S. Lindley, F. de Vocht, R. Agius, G. Kerry, K. Johnson, M. Ashmore, A. Terry, S. Dimitroulopoulou, and A. Simpson (2012). "Performance of a microenviromental model for estimating personal NO2 exposure in children". In: *Atmospheric Environment* 51.0, pp. 225–233.

Moskal, L. M., D. M. Styers, and M. Halabisky (2011). "Monitoring Urban Tree Cover Using Object-Based Image Analysis and Public Domain Remotely Sensed Data". In: *Remote Sensing* 3.12, pp. 2243–2262. ISSN: 2072-4292.

Motaghian, H. R. and J. Mohammadi (2011). "Spatial Estimation of Saturated Hydraulic Conductivity from Terrain Attributes Using Regression, Kriging, and Artificial Neural Networks". In: *Pedosphere* 21.2, pp. 170–177.

Mukerjee, S., L. a. Smith, M. M. Johnson, L. M. Neas, and C. a. Stallings (2009). "Spatial analysis and land use regression of VOCs and NO2 from school-based urban air monitoring in Detroit/Dearborn, USA". In: *Science of The Total Environment* 407.16, pp. 4642–4651.

National Records of Scotland (2013). *Mid-Year Population Estimates*. Tech. rep. Edinburgh: National Records of Scotland.

Neteler, M., M. H. Bowman, M. Landa, and M. Metz (2012). "GRASS GIS: A multi-purpose open source GIS". In: *Environmental Modelling & Software* 31, pp. 124–130. ISSN: 13648152.

Novack, T., T. Esch, H. Kux, and U. Stilla (2011). "Machine Learning Comparison between WorldView-2 and QuickBird-2-Simulated Imagery Regarding Object-Based Urban Land Cover Classification". In: *Remote Sensing* 3.12, pp. 2263–2282. ISSN: 2072-4292.

Okabe, A., T. Satoh, and K. Sugihara (2009). "A kernel density estimation method for networks, its computational method and a GIS-based tool". In: *International Journal of Geographical Information Science* 23.1, pp. 7–32. ISSN: 1365-8816.

Oliphant, T. E. (2007). "Python for Scientific Computing". In: *Computing in Science & Engineering* 9.9, pp. 10–20.

Pearce, J. L., S. L. Rathbun, M. Aguilar-Villalobos, and L. P. Naeher (2009). "Characterizing the spatiotemporal variability of PM2.5 in Cusco, Peru using kriging with external drift". In: *Atmospheric Environment* 43.12, pp. 2060–2069.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Pirjola, L., T. Lähde, J. Niemi, A. Kousa, T. Rönkkö, P. Karjalainen, J. Keskinen, A. Frey, and R. Hillamo (2012). "Spatial and temporal characterization of traffic emissions in urban microenvironments with a mobile laboratory". In: *Atmospheric Environment* 63, pp. 156–167. ISSN: 13522310.

Preatoni, D., C. Tattoni, F. Bisi, E. Masseroni, D. D'Acunto, S. Lunardi, I. Grimod, and G. Tosi (2012). "Open source evaluation of kilometric indexes of abundance". In: *Ecological Informatics* 7.1, pp. 35–40.

Rada, E. C. (2014). "Sustainable city and air pollution". In: *Wit Transactions on ecology and the environment*, pp. 1369–1380.

Rada, E. C., M. Ragazzi, M. Brini, L. Marmo, P. Zambelli, M. Chelodi, and M. Ciolli (2012). "Perspectives of low-cost sensors adoption for air quality monitoring". In: *Scientific Bulletin - Politehnica University Of Bucharest. Series D, Mechanical Engineering* 74.2, pp. 241–250.

Ragazzi, M., E. C. Rada, A. Chisté, M. Schiavon, M. Ciolli, P. Zambelli, M. Brini, L. Marmo, and M. Chelodi (2012). "A contribution to the development of a public - private integrated network of wireless sensors for an enhanced management of air quality". In: *Sustainable Technology for Environmental Protection*, pp. 1–4.

Rahman, M. M., G. J. Hay, I. Couloigner, and B. Hemachandran (2014). "Transforming Image-Objects into Multiscale Fields: A GEOBIA Approach to Mitigate Urban Microclimatic Variability within H-Res Thermal Infrared Airborne Flight-Lines". In: *Remote Sensing* 6, pp. 9435 –9457.

Ram, S. S., S Majumdar, P Chaudhuri, S Chanda, S. C. Santra, P. K. Maiti, M Sudarshan, and a Chakraborty (2012). "SEMEDS: an important tool for air pollution bio-monitoring." In: *Micron (Oxford, England : 1993)* 43.2-3, pp. 490–3.

Rawi, M. and A. Al-Anbuky (2011). "Development of Intelligent Wireless Sensor Networks for Human Comfort Index Measurement". In: *Procedia Computer Science* 5, pp. 232–239.

Richmond-Bryant, J., S. S. Isukapalli, and D. A. Vallero (2011). "Air pollutant retention within a complex of urban street canyons". In: *Atmospheric Environment* 45.40, pp. 7612–7618.

Ross, Z., M. Jerrett, K. Ito, B. Tempalski, and G. D. Thurston (2007). "A land use regression for predicting fine particulate matter concentrations in the New York City region". In: *Atmospheric Environment* 41.11, pp. 2255–2269.

Sacchelli, S., I. Bernetti, I. De Meo, L. Fiori, A. Paletto, P. Zambelli, and M. Ciolli (2014). "Matching socio-economic and environmental efficiency of wood-residues energy chain: a partial equilibrium model for a case study in Alpine area". In: *Journal of Cleaner Production* 66, pp. 431–442. ISSN: 09596526.

Sacchelli, S., P. Zambelli, and M. Ciolli (2013). "Biomasfor - An open source holistic model for the assessment of sustainable forest bioenergy". In: *iForest* 1.Biogeosciences and Forestry, pp. 285–293.

Sally Liu, L.-J., M.-Y. Tsai, D. Keidel, A. Gemperli, A. Ineichen, M. Hazenkamp-von Arx, L. Bayer-Oglesby, T. Rochat, N. Künzli, U. Ackermann-Liebrich, P. Straehl, J. Schwartz, and C. Schindler (2012). "Long-term exposure models for traffic related NO2 across geographically diverse areas over separate years". In: *Atmospheric Environment* 46.2, pp. 460–471. ISSN: 13522310.

Salo, H., M. S. Bućko, E. Vaahtovuo, J. Limo, J. Mäkinen, and L. J. Pesonen (2012). "Biomonitoring of air pollution in SW Finland by magnetic and chemical measurements of moss bags and lichens". In: *Journal of Geochemical Exploration* 115, pp. 69–81.

Schiavon, M., G. Antonacci, E. C. Rada, M. Ragazzi, and D. Zardi (2014). "Modelling human exposure to air pollutants in an urban area". In: *Revista de Chimie*, pp. 61–64.

Schoolman, E. D. and C. Ma (2012). "Migration, class and environmental inequality: Exposure to pollution in China's Jiangsu Province". In: *Ecological Economics* 75, pp. 140–151.

Senthilnath, J., S. Bajpai, S. Omkar, P. Diwakar, and V. Mani (2012). "An approach to multi-temporal MODIS image analysis using image classification and segmentation". In: *Advances in Space Research* 50.9, pp. 1274–1287. ISSN: 02731177.

Shad, R., M. S. Mesgari, A. Abkar, and A. Shad (2009). “Predicting air pollution using fuzzy genetic linear membership kriging in GIS”. In: *Computers, Environment and Urban Systems* 33.6, pp. 472–481.

Sheppard, L., J. C. Slaughter, J. Schildcrout, L.-J. S. Liu, and T. Lumley (2005). “Exposure and measurement contributions to estimates of acute air pollution effects.” In: *Journal of exposure analysis and environmental epidemiology* 15.4, pp. 366–76.

Singh, V., C. Carnevale, G. Finzi, E. Pisoni, and M. Volta (2011). “A cokriging based approach to reconstruct air pollution maps, processing measurement station concentrations and deterministic model simulations”. In: *Environmental Modelling & Software* 26.6, pp. 778–786.

Skelhorn, C., S. Lindley, and G. Levermore (2014). “The impact of vegetation types on air and surface temperatures in a temperate city: A fine scale assessment in Manchester, UK”. In: *Landscape and Urban Planning* 121, pp. 129–140. ISSN: 01692046.

Steiniger, S. and G. J. Hay (2009). “Free and open source geographic information tools for landscape ecology”. In: *Ecological Informatics* 4.4, pp. 183–195. ISSN: 15749541.

Steinle, S., S. Reis, and C. E. Sabel (2013). “Quantifying human exposure to air pollution-Moving from static monitoring to spatio-temporally resolved personal exposure assessment.” In: *The Science of the total environment* 443, pp. 184–93. ISSN: 1879-1026.

Stevens, C. J., S. M. Smart, P. a. Henrys, L. C. Maskell, A. Crowe, J. Simkin, C. M. Cheffings, C. Whitfield, D. J. Gowing, E. C. Rowe, A. J. Dore, and B. a. Emmett (2012). “Terricolous lichens as indicators of nitrogen deposition: Evidence from national records”. In: *Ecological Indicators* 20, pp. 196–203.

Stuart, A. L. and M. Zeager (2011). “An inequality study of ambient nitrogen dioxide and traffic levels near elementary schools in the Tampa area.” In: *Journal of environmental management* 92.8, pp. 1923–30.

Su, J. G., M. Brauer, B. Ainslie, D. Steyn, T. Larson, and M. Buzzelli (2008a). “An innovative land use regression model incorporating meteorology for exposure analysis”. In: *Science of the Total Environment* 390, pp. 520–529. ISSN: 00489697.

Su, J. G., M. Brauer, and M. Buzzelli (2008b). “Estimating urban morphometry at the neighborhood scale for improvement in modeling long-term average air pollution concentrations”. In: *Atmospheric Environment* 42.34, pp. 7884–7893.

Tattoni, C., M. Ciolli, F. Ferretti, and M. G. Cantiani (2010). “Monitoring spatial and temporal pattern of Paneveggio forest (Northern Italy) from 1859 to 2006”. In: *iForest* 1.Biogeosciences and Forestry 3, pp. 72–80.

Tattoni, C., F. Rizzoli, and P. Pedrini (2012). “Can LiDAR data improve bird habitat suitability models?” In: *Ecological Modelling* 245, pp. 103–110.

USEPA - U.S. Environmental Protection Agency (2010). *Integrated Science Assessment for Carbon Monoxide*. Tech. rep. National Center for Environmental Assessment, Office of Research and Development., EPA/600/R–09/019B.

van Rossum, G. (1995). *Python library reference*. Report CS-R9524. pub-CWI, pp. iv + 186.

Varrazzo, D. and Psycopg Community (2013). “Psycopg - PostgreSQL database adapter for Python”. In:

Vedrenne, M., R. Borge, J. Lumbreras, and M. E. Rodríguez (2014). “Advancements in the design and validation of an air pollution integrated assessment model for Spain”. In: *Environmental Modelling & Software* 57, pp. 177–191. ISSN: 13648152.

Vettorato, D. and P. Zambelli (2009). “Estimation of Energy Sustainability at Local Scale : An Approach Based on Innovative Analytical and Mapping Tools and Multicriteria Analysis”. In: In Proceedings of 45th ISOCARP Congress Porto - Portugal, 18–22 October 2009, pp. 1–12.

Vettorato, D., D. Geneletti, and P. Zambelli (2011). “Spatial comparison of renewable energy supply and energy demand for low-carbon settlements”. In: *Cities* 28.6, pp. 557–566. ISSN: 0264-2751.

Vieira, M. A., A. R. Formaggio, C. D. Rennó, C. Atzberger, D. A. Aguiar, and M. P. Mello (2012). “Object Based Image Analysis and Data Mining applied to a remotely sensed Landsat time-series to map sugarcane over large areas”. In: *Remote Sensing of Environment* 123, pp. 553–562. ISSN: 00344257.

Vienneau, D, K de Hoogh, and D Briggs (2009). “A GIS-based method for modelling air pollution exposures across Europe”. In: *Science of The Total Environment* 408.2, pp. 255–266.

Vlachokostas, C., C. Achillas, N Moussiopoulos, and G Banias (2011). “Multicriteria methodological approach to manage urban air pollution”. In: *Atmospheric Environment* 45.25, pp. 4160–4169.

Walt, S. van der, S. C. Colbert, and G. Varoquaux (2011). “The NumPy Array: A Structure for Efficient Numerical Computation”. In: *Computing in Science & Engineering* 13.13, pp. 22–30.

Wang, N., N. Zhang, and M. Wang (2006). “Wireless sensors in agriculture and food industry âĂŤ Recent development and future perspective”. In: *Computers and Electronics in Agriculture* 50.1, pp. 1 –14.

Wieland, M. and M. Pittore (2014). “Performance Evaluation of Machine Learning Algorithms for Urban Pattern Recognition from Multi-spectral Satellite Images”. In: *Remote Sensing* 6.4, pp. 2912–2939. ISSN: 2072-4292.

Wikipedia (2014a). *Confusion matrix — Wikipedia, The Free Encyclopedia*. [Online; accessed 28-September-2014].

— (2014b). *Python (programming language) — Wikipedia, The Free Encyclopedia*. [Online; accessed 2014-07-23].

Wild, C. P. (2005). “Complementing the genome with an "exposome": the outstanding challenge of environmental exposure measurement in molecolar epidemiology”. In: *Cancer Epidemiol Biomakers Prev* 14.8, pp. 1847–1850.

World Health Organization (WHO) (2014a). *7 million premature deaths annually linked to air pollution*. GENEVA.

— (2014b). *Ambient (outdoor) air quality and health, Fact sheet N.313,*

Zambelli, P., C. Lora, M. Ciolli, R. Spinelli, C. Tattoni, A. Vitti, and P. Zatelli (2010). “A FOSS4G model to estimate forest extraction methods and biomass availability for renewable energy production”. In: *FOSS4G*. Vol. 1. Barcelona.

Zambelli, P., C. Lora, R. Spinelli, C. Tattoni, A. Vitti, P. Zatelli, and M. Ciolli (2012). “A GIS decision support system for regional forest management to assess biomass availability for renewable energy production”. In: *Environmental Modelling & Software* 38, pp. 203–213. ISSN: 13648152.

Zambelli, P., S. Gebbert, and M. Ciolli (2013). “Pygrass : an object oriented Python API for GRASS GIS”. In: *ISPRS International Journal of Geo-Information* 2, pp. 201–219.

Zhang, K. and B. Hu (2012). “Individual Urban Tree Species Classification Using Very High Spatial Resolution Airborne Multi-Spectral Imagery Using Longitudinal Profiles”. In: *Remote Sensing* 4.12, pp. 1741–1757. ISSN: 2072-4292.