GIS-Enabled Water Quality Risk Assessment in a Drinking Water Catchment

Ian Douglas McDonald

B. Sc. (Hons), Dip. Ed., Dip. Comp. Sci.

Health and Environment Group

College of Science and Engineering

Flinders University

Submitted to the College of Science and Engineering in partial fulfilment of the requirements for the degree of Master of Geospatial Information Science at Flinders University – Adelaide Australia

Date of submission of thesis: 11 March 2020

Contents

LIST OF FIGURES

LIST OF TABLES

SUMMARY

South Australian Water (SAW) provided a well-established environment and workflow to model the movement of *Cryptosporidium* from agricultural sources in the Mount Lofty Ranges (MLR) to primary water storages. This environment included a methodology for maintaining project data in Excel spreadsheets. The modelling system was ArcMap and its Raster Calculator was used to implement the model. SAW users expressed frustration with the Raster Calculator approach and its lack of record-keeping facilities.

This project sought to address these, and other, issues through the development of a decision support infrastructure. Superficially it comprises a simple interface for data maintenance and reporting. Underlying this is a suite of support functions that enable model definitions in a spreadsheet and facilitate the record-keeping necessary for long-term sensitivity studies. The development environment for the project was Python and the script was written in such a way that it could readily accommodate different pollutants and new model definitions.

The new model added to the capabilities of the previous one by:

- accommodating the absence of pathogens
- focusing on just that part of the catchment, i.e., a watershed, that supplied water to the monitoring stations
- introducing a flow parameter that was dependent on the watershed definitions
- providing for interactions between variables in addition to simply adding their effects.

The project has developed the position that flowing water is required to deliver pathogens and nutrients to the SAW monitoring stations. As defined in previous SAW work, the main drivers for flowing water were slope and rainfall.

Previously the risk evaluated at a cell was determined solely by the values of the contributing variables at that cell. Now, a new spatial characteristic, flow, has been introduced. Its value at a cell is influenced by what is occurring upstream of that cell. This variable has been employed in two ways, either to add to the risk like other terms, or to interact with the source and transport terms to influence their contributions.

The model has been modified considerably during this project. The generation of negative risk values has been an ongoing concern. This has been addressed by reducing the impact of the sources of negative risk – buffers and fencing. Also, the growth of the land use file from 10,000 rows to over 44,000 rows changed the disposition of risk significantly. In the light of changes to the model, and weight and risk estimates, it is recommended that the system now go back to SAW for a comprehensive review by the catchment management scientists.

DECLARATION

I certify that this thesis does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person except where due reference is made in the text.

Signed...I D McDonald.....

Date.....11 March 2020.....

ACKNOWLEDGEMENTS

I am indebted to my supervisor, Professor Howard Fallowfield, for his dedicated oversight. Our regular conversations about the project and other matters have been both invaluable and enjoyable.

The staff at SA Water have been particularly generous in sponsoring this project. Their prior work provided a secure foundation for my research which could not have succeeded without access to their “corporate memory”. Particular thanks go to Ms Hayley Abbott, Dr Brooke Swaffer, Ms Mel Abela, Mr Jeff Newman and Dr Ben van den Akker for their generous contributions of encouragement, time and information. I am also aware of the enthusiastic support in the background of Dr Jacqueline Frizenschaf.

Mr Robert Keane of Flinders University has a deep knowledge of the ArcGIS application. He willingly shared his understandings with me on numerous occasions during my studies.

Finally, I am at a loss to find a sober academic statement to express my deep gratitude to my wife, Dr Jo Ankor for her unerring support. She has gone before me as a mature-age higher degree student so her insights have been invaluable. But in so many other ways, she has “kept the home fires burning” so I could focus both on the study and the huge health scare I (we!) had to endure at the same time.

1. INTRODUCTION

1.1 Background

About 60% of Adelaide's water supply is collected from the Mount Lofty Ranges (MLR) (EPA 2004). The remainder of the water is pumped from the Murray River into the MLR and is therefore subjected to the same factors affecting the quality of the water (EPA 2007, p. 3). At the time of writing, the demand on Murray River water was set to be reduced by substituting the output from a desalination plant south of Adelaide at Port Stanvac.

Around 90% of the MLR is privately owned, subject to human activity (EPA 2019). This dual use has created problems for water quality since the 1880s when settlement and agriculture were prioritised over the need for safe drinking water (SA Water 2019a). There are now over 50,000 people living in the MLR (EPA 2007, p. 3). Land use includes light industry, grazing, forestry, orchards, market gardening and urban (including unsewered) development (EPA 2007, pp. 5-6).

The regulations of the 1924 Waterworks Act were amended in 1974 to create the Mount Lofty Ranges Watershed which encompassed existing and potential water supply catchments within the MLR. This Act provided the legal framework for the State government to establish controls on land development and management to protect water supplies (EPA 2007, p. 3).

A number of agencies have been given responsibilities in the management of the quality of water from the MLR, ranging from local councils, through to state and federal government departments. At a local level, for example, the Adelaide Hills Council (AHC) has developed a water management plan (AHC 2017). This plan reported on the state of the AHC district, e.g., with commentary on the poor health of riparian vegetation (John & Flehr 2011, p. 100). It also offers guidance on developments that are appropriate (or not) in the watershed primary production zone (AHC 2013).

Some agencies that operate at a State level include:

- South Australia's Environment Protection Authority (EPA), which was created in the Environmental Protection Act 1993 (Government of South Australia 2019a) with responsibilities for overseeing the development of policy and its implementation.
- In 2004, the Natural Resources Management Act, which established eight regional Natural Resources Management (NRM) boards (Government of South Australia 2019b). As with all the NRM boards, the one for Adelaide and Mount Lofty Ranges endeavours to work with all levels of government, industry, primary producers and the community to develop a ten-year strategic outlook for the management of natural resources, including water. It offers specific advice, e.g., on development and management requirements for farm dams (Natural Resources AMLR 2019a).

- The State's Department of Health and Ageing, which has published a series of annual reports in relation to the Safe Drinking Water Act (Department for Health and Ageing 2016).

At the national level, the Commonwealth Environmental Water Office (Australian Government 2019) manages the disposition of environmental water in the Murray-Darling Basin according to policy and advice available in the Water Act 2007, including the Murray-Darling Basin Agreement (Federal Register of Legislation 2019). Water from the basin is pumped into MLR storages from Mannum and Murray Bridge (SA Water 2019b). The National Health and Medical Research Council (NH&MRC) operates as a statutory agency reporting to the Minister for Health and Ageing (NH&MRC 2019). It supports the development of guidelines for drinking water (NH&MRC 2011) and the augmentation of drinking water by recycling (NH&MRC 2009). The Drinking Water Guidelines are discussed further below (Section 1.3).

In 2000, the EPA (EPA 2000, p. 25) reported on the state of the health of the MLR catchments. It nominated a number of issues that affected water quality:

- toxic algal blooms in dams and reservoirs
- stock deaths resulting from consumption of water contaminated by toxic algae
- pathogen (e.g., *Cryptosporidium* and *Giardia*), pesticide and sediment contamination of rivers and streams
- localised heavy metal contamination.

The report identified a number of causes of the contamination, including:

- poor management of septic tank systems
- erosion resulting from stock access to watercourses, overgrazing and cropping on steep slopes
- a large number of farm dams that impede the flow of water in many major watercourses
- inappropriate planning consents.

These matters raised significant management concerns. The report posed the question, "Who manages streams in the Mount Lofty Ranges?" and noted that the following entities were involved:

- seven local government councils (Barossa, Playford, Tea Tree Gully, Onkaparinga, Mount Barker, Adelaide Hills and Alexandrina)
- five catchment water management boards (Torrens, Patawalonga, Onkaparinga, River Murray, and Northern Adelaide and Barossa)
- approximately 67 Landcare groups
- four soil boards (Central Hills, Northern Hills, Southern Hills and Murray Plains)
- thousands of landholders
- Mount Lofty Ranges Catchment Program Board

- State Government departments and agencies (Department for Environment and Heritage (EPA); SA Water; PIRSA; Planning SA; Transport SA; Department for Water Resources.)

The South Australian Water Corporation (SAW) was established in place of the Engineering and Water Supply Department in 1994. SAW traces its origins back to 1856 when it was known as the Waterworks & Drainage Commission. It had responsibilities then "to solve the water supply and sewerage problems of the city". It still has these responsibilities although its reach is far greater, across the State of South Australia (SA Water 2019a) and the regulatory and management regimes are clearly far more complex.

1.2 The nature of water contamination in the MLR

Water contamination can be caused by pathogens, nutrients, pesticides, sediments and other compounds. In this project, most of the development relates to pathogens with some application to nutrients. The presence in surface run-off of pesticides and fungicides has also been a concern, e.g., in MLR apple and cherry orchards in the period 2007 – 2009 (Oliver, DP et al. 2012b; Oliver, DP et al. 2012a; Oliver, DP et al. 2012c), but has not been studied here.

1.2.1 Pathogens

Adelaide Hills Council's Water Management Plan describes the use of Community Wastewater Management Systems (CWMS) to collect wastewater from customer's septic tanks. The plan notes that not all parts of the district are serviced by CWMS. Some customer equipment is faulty, some pollutants are deliberately washed down drains and others drain off roads and other hard surfaces (AHC 2017, pp. 9-12).

Agricultural practices also contribute to the contamination problem due to pathogen distribution from animal waste into surface water. The review by Ferguson et al. (2003) of pathogens in surface water noted that there are over 100 enteric viruses of concern including coxsackie, rotaviruses and Norwalk-like viruses. Bacterial pathogens may be endemic (*Aeromonas, Legionella, Mycobacterium* and *Vibrio*) or sourced from sheep and cattle (*Campylobacter, Salmonella* and *Escherichia coli*). Common protozoa include *Giardia* and *Cryptosporidium*. The latter is considered a better model for understanding the transport and fate of water-borne protozoa because of the persistence of their oocysts compared to those of *Giardia*. Movement of oocysts through soils is also possible (Darnault et al. 2017) but is not considered here.

In order to model and manage the behaviour of *Cryptosporidium* in the MLR catchments, reasonable measures of its numbers and disposition must be obtained. Roser and Ashbolt (2007) studied six catchments (and two aquifers) across southern Australia to understand what biological, physical and chemical markers could be used to determine water quality. *Cryptosporidium* was one of the microorganisms studied. The South Australian sites were in the MLR – Sixth Creek, Aldgate Creek and Myponga River. They were chosen for their "part impacted" (i.e., some dual

use), urbanised and intensive agriculture characteristics respectively. Flow weighted (event) means (per 10L) for *Cryptosporidium* ranged from 31 (Myponga River) to 290 (Aldgate Creek).

Peak flow measurements have been obtained for Clarendon Weir (Swaffer et al. 2014) in the mid-MLR and for nine sites in the northern MLR measuring flows into the Little Para and Millbrook Reservoirs (Swaffer et al. 2018). Peak flows were measured as the systems are ephemeral, with little or no flow for much of the year. Data collected included *Cryptosporidium* species, infectivity rates and land use types. The *Cryptosporidium* spp. of most concern for human infection (*C. parvum, C. cuniculus*) were attributed to livestock, rabbits, dogs and foxes. Zahedi et al. (2018) made similar observations across 11 catchments in Queensland, New South Wales and Western Australia, although they particularly noted the addition of *C. hominis* in cattle. In all, they recorded 13 species of *Cryptosporidium* that are of concern for human health. Average numbers (per 10L) seen during run-off events ranged from 12 to 187 in the Millbrook precinct although maximum values could be as high as 2,176 (Swaffer et al. 2018).

Bukhari and Smith (1997) cited in Swaffer et al. (2014) estimated the infectivity rate of oocysts in a fresh excretion of experimentally infected lambs of about 50%. However, downstream measured oocyst infectivity rate was determined to be 3.1%, suggesting that a 1 log reduction in infectivity could occur naturally within the catchments themselves. Also, the annual peak in human infection was not related to the peak in animal oocyst release (in calves), suggesting other, human-to-human factors may be operating downstream from the reservoirs (Swaffer et al. 2014).

King and Monis (2007) reviewed the environmental factors that could affect the survival of *Cryptosporidium* oocysts:

- temperature
- ammonia
- desiccation
- soil matrix and vegetation
- solar (UV) radiation
- predation by other organisms such as rotifers, ciliates, amoebae etc.
- settling into the sediment of lakes and reservoirs

Modification of existing practices can also influence the ability of oocysts to survive to the water treatment plant (WTP). For example, if the inflow to a reservoir is colder and therefore denser than the reservoir water, the incoming water can flow along the bottom of the reservoir to the dam wall. A change in the water off-take level could reduce the level of organisms and other organic matter reaching the WTP (Hobson et al. 2010). A number of strategies was evaluated in the Myponga catchment (Bryan et al. 2009; Kandulu & Bryan 2009):

- microfiltration
- enhanced coagulation
- ultraviolet
- dung beetles
- targeted watercourse management

If an assessment is made just on cost, the first of these might be judged the most appropriate. However, a wider cost benefit analysis can be made that includes ecosystem benefits (improved water quality, biodiversity, carbon sequestration). When the changes are developed and implemented with the engagement of landholders, watercourse management comes into its own. Reductions of up to 90% in export of *Cryptosporidium* to the Myponga reservoir could be anticipated (Bryan et al. 2009).

1.2.2 Nutrients

Studies of nutrients in surface water focus on total Phosphorus (TP), total Nitrogen (TN) and total organic Carbon (TOC). MLR studies have investigated local nutrient and total suspended solids (TSS) runoff from apple and cherry orchards (Cock Creek catchment) and a vineyard (Charleston catchment) (Cox, JW et al. 2012). TP was found to exceed the "Australian environmental trigger value" (AETV) in >90% of samples from the vineyard but <10% of samples from the apple and cherry orchard. Maximum TP amounts were 0.31mg/L (apple orchard), 0.54mg/L (cherry orchard) and 2.79mg/L (vineyard). The AETV for TP was reported at the time to be 0.1mg/L by the Australian and New Zealand Environment and Conservation Council and Agriculture and Resource Management Council of Australia and New Zealand (2000), cited by (Cox, JW et al. 2012). Conversely, >90% of the samples from the cherry orchard and the vineyard exceeded the standard for TN.

Detailed studies have also been carried out across the wider MLR (Fleming et al. 2010a, 2010b). Data were sourced from small single land-use catchments (2 – 200Ha) and from the composite sampler program largely managed by the Adelaide and Mount Lofty NRM. In all, 21 datasets were studied for use in the Source Catchments modelling framework. This led to an updating (Table 1.1) of event mean concentrations of TP, TN and total suspended solids (TSS) by land use type for use in catchments modelling (Fleming et al. 2010a).

Table 1.1: Updated TSS, TN and TP event mean concentrations (EMC) for each MLR land use type (derived from Fleming et al. 2010a, 2010b, used with permission)

Land Use	EMC (mg/L)		
	TSS	TN	TP
Conservation area	43	1.8	0.18
Grazing	184	2.1	0.24
Suburban	43	1.2	0.12
Intensive grazing	300	2.8	0.50
Annual horticulture	308	5.3	0.93
Perennial horticulture	146	1.6	0.13

Data such as these are being incorporated into models and decision support tools to predict the impact of land use changes (Cox, JW et al. 2013, pp. 36-7). The expectation is that it will be possible to incorporate temporal and spatial variations identified with changing seasons in MLR catchments.

1.3 Australian Drinking Water Guidelines and risk

The Australian Drinking Water Guidelines (ADWG) provide an administrative and legislative framework that holds suppliers and health authorities accountable for the quality of drinking water in Australia. In this context, 'quality' is assessed against the physical, microbial, chemical and radiological characteristics of the water. The framework comprises 12 elements (NH&MRC 2011, p. 14):

1. Commitment to drinking water quality management
2. Assessment of the drinking water supply system
3. Preventative measures for drinking water quality management
4. Operational procedures and process control
5. Verification of drinking water quality
6. Management of incidents and emergencies
7. Employee awareness and training
8. Community involvement and awareness
9. Research and development
10. Documentation and reporting
11. Evaluation and audit
12. Review and continual improvement.

This project addresses aspects of elements two and three in particular.

1.3.1 Element 2: assessment of the drinking water supply system

The ADWG define the supply system as everything between the collection point and the consumer, including catchments, source waters, storage waters and intakes, treatment systems, distribution systems and consumers (NH&MRC 2011, p. 26). Assessments of these components are made in relation to:

- hazards – biological, chemical, physical or radiological agents that may cause harm, e.g., *Cryptosporidium parvum.* Hazards can occur throughout the water supply system.
- hazardous events – incidents causing a hazard to be manifested, e.g., failure in a treatment plant resulting in the release of *C. parvum* into the distribution system
- risks – the likelihood that a hazardous event will occur, resulting in harm, and the severity of the harm, e.g., the probability that *C. parvum* is present in source waters and reaches the treatment plant in sufficient numbers to cause illness should the plant fail (NH&MRC 2011, p. 28).

The Water Services Association of Australia (WSAA 2015, pp. 13-9) proposes two strategies to meet the ADWG Element 2 requirement for assessment of source water:

1. Tier 1 – mandatory
    a. perform sanitary surveys of pathogen sources and their relationship to water sources and barriers
    b. aggregate sanitary survey data to derive a vulnerability assessment for the source
    c. accumulate microbial contamination data for raw water immediately before treatment

    The above information is used to assign the source to one of four catchment vulnerability categories (protected, moderately protected, poorly protected, unprotected)
2. Tier 2 – optional

    If sufficient raw water pathogen data is available, a quantitative microbial risk assessment (QMRA) can be made. The aim of this calculation is to determine what $\log_{10}$ reduction in oocyte concentrations is required to achieve one µDALY (disability-adjusted life year) (Water Research Australia 2019). Tier 2 complements (i.e., does not replace) the assessment developed in the Tier 1 work. Both of these strategies offer guidance on treatments required to make the water safe.

1.3.2 Element 3: preventative measures for drinking water quality management

Just as hazards can occur throughout the drinking water system, so too should the application of preventative measures. These should be applied as near as possible to the source of the hazard. Preventative measures are thought of as barriers. They are established throughout the system so that if one fails, others downstream will be able to compensate. Barriers may include:

- catchment management and source water protection, e.g., development controls, exclude human activity, protect riparian vegetation

- detention in protected reservoirs or storages (microorganisms are inactivated due to settling or UV irradiation)
- extraction management, e.g., vary extraction points vertically and horizontally according to the quality, volume and density of reservoir inflows
- coagulation, flocculation, sedimentation and filtration
- disinfection
- protection and maintenance of the distribution system (NH&MRC 2011, pp. 31-4).

A significant advantage in applying barriers as near as possible to the source is that it reduces the need for disinfection at the treatment plant. This has economic benefits due to:

- reduced cost of chemicals
- reduced energy use
- improved health outcomes resulting from a reduction in disinfection by-products

It is not common for sufficient information to be available to permit a fully quantified risk assessment. As will be seen below, professional judgement will be called upon to assess risk levels, and a tolerance for uncertainty will be required. Nevertheless, consumers must be kept safe so robust incident response capabilities must be in place (NH&MRC 2011, p. 46). The ADWG provide a methodology for assessing risk, and guidance on the application of elements two and three of the framework (NH&MRC 2011, pp. 29-30, 1110-24).

1.4 Modelling flows of water and pollutants

Oliver, DM et al. (2016) reviewed modelling approaches used in predicting microbial water quality as measured by faecal indicator organisms (FIO) in catchment systems. Their review nominated three classes of model:

- export coefficient and regression
- probabilistic or risk-based approaches where inputs are presented as samples from probability distributions
- mechanistic or process-based models

They cautioned that the choice of modelling approach will be influenced by the temporal and spatial scales used and by any requirement for transferability of the approach across contrasting agricultural systems.

Neill et al. (2018) used two different modelling approaches to study an 11-year dataset that recorded observations of *E. coli* at ten monitoring sites distributed across a mixed land use catchment in the north-east of Scotland. The data did not display a clear flow-concentration relationship. However, a number of linear regression models indicated that anthropogenic point

sources were significant predictors of the spatial patterns of *E. coli* concentrations. Neither arable nor pasture land was found to be significant.

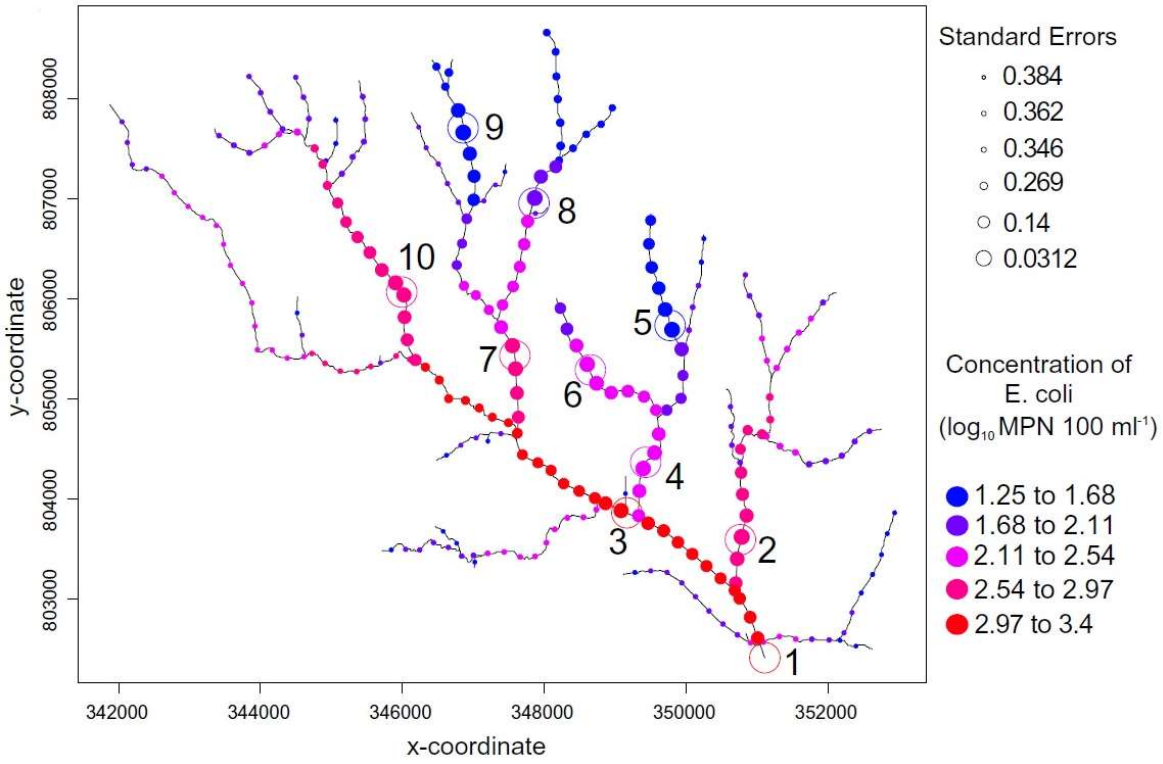


Figure 1.1: Predicted concentrations of *E. coli* in relation to monitoring sites (numbered) for Tarland Burn in NE Scotland (Neill et al. 2018, used with permission)

Spatial-stream-network models (SSNM) were better at predicting the catchment scale concentrations that could be targeted with remediation work. Figure 1.1 illustrates the SSNM output showing increasing concentrations of *E. coli* down through the stream network.

Porter et al. (2017) tested the transferability of a modelling approach by taking one developed for analysing diffuse fine sediment pollution and applying it to the mapping of diffuse FIO risk. Good results were obtained for one of the two catchments studied (r_s of 0.88, p, 0.01), but not for the other (r_s = -0.357, $p > 0.05$). It was found that modifications to the original sediments' framework would be required for a better realisation of the fate and transport of FIOs.

1.4.1 Regression models

The simplest form of regression model assumes a linear relationship between a dependent variable and one or more independent variables (Rogerson 2015, p. 225). The linear relationship is determined using ordinary least squares (OLS) to find the line of best fit to the observations. McGrane, Tetzlaff and Soulsby (2014) employed this approach to estimate faecal coliform (FC) bacteria in the aquatic systems of catchments in relation to catchment characteristics. Using a multiple linear regression, they established that variation in mean annual FC concentrations was best explained by percentage of improved pasture (90%) and human population size (62%).

Although this outcome provides guidance in predicting the effects of changing population sizes and agricultural practices, the authors concluded that more information was required in relation to short term dynamics. A more expansive sampling network would aid in identifying contaminant sources.

Not all relationships between dependent and independent variables are linear. The influence of a parameter may vary spatially, i.e., *what* is observed may vary in part as a consequence of *where* it is observed – it exhibits non-stationarity. Problems like this can be analysed with geographically weighted regression (GWR) which assigns higher weights to other observations closer to a given data point compared to those that are further away (Fotheringham, Brunsdon & Charlton 2002). Aljassim (2018) used GWR to analyse the effect of land use changes on the prevalence of FC bacteria in a river shellfish production area. Predictors for FC risk included rainfall, water and air temperature, salinity and tide stage, plus the land use classes, residential, forestlands and open spaces. However, the influences of these parameters on the model outcome were most significant within 1,800 m of the shellfish monitoring stations.

Chen et al. (2016) compared multivariate OLS with GWR to study the effect of land use and population on N and P concentrations in surface water. A manual method was used to include or exclude independent variables in order to reduce the impact of collinearity (rather than exploit it). Compared with stepwise multiple linear regressions, their optimised OLS technique improved the predictive capability of the model (as measured by Adjusted R^2 values) by 14.3%. The improvement produced by the GWR approach was 59.2%.

Multivariate adaptive regression splines (MARS) (Friedman 1991) accommodates multicollinearity as well as a large number of variables, non-linearity and a substantial interaction amongst predictors (Muñoz & Felicísimo 2004). The goal of MARS is to produce a parsimonious model.

Alonso Fernández et al. (2014) developed a data mining strategy using MARS to model eutrophication and risk prevention in the Trasona reservoir in Northern Spain. The objective was to predict chlorophyll in the reservoir. They analysed seven biological (e.g., densities of *Cyanobacteria*, *Cryptophytes*, etc.) and 15 physical/chemical variables (water temperature, turbidity, total Nitrogen, etc.). MARS determined that 10 of the variables would be used in the model (in order of importance):

- Cyanobacteria
- Phosphates_concentration
- Dinophlagellata
- Ambient_temperature
- Alkalinity
- Water_temperature
- Conductivity

- Turbidity
- Total_phosphorus
- Nitrite_concentration

The model displayed a good fit with the 149 observations obtained at the reservoir over the period 2006 – 2010 (Figure 1.2).

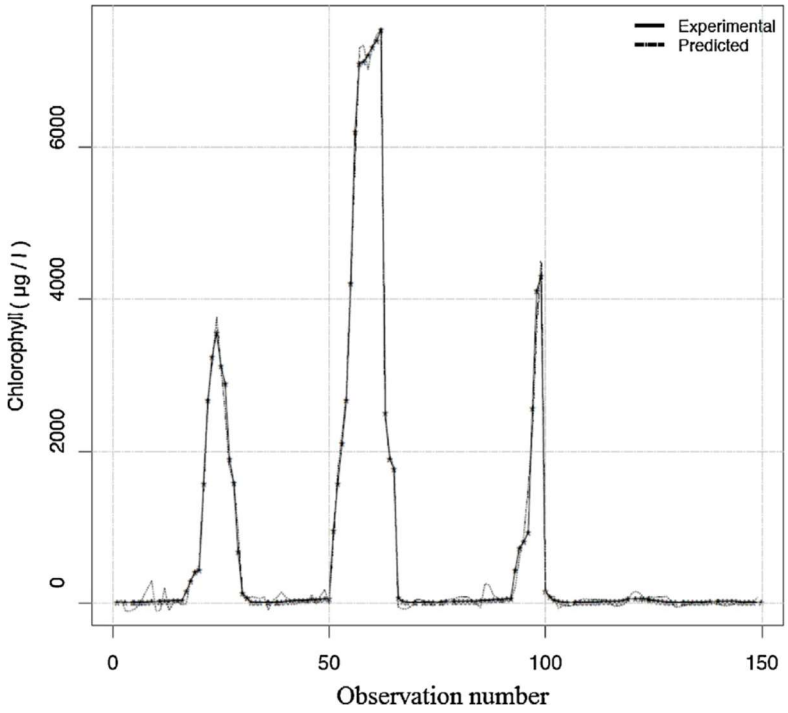


Figure 1.2: Comparison between observed and predicted levels of eutrophication in the Trasona reservoir (Alonso Fernández et al. 2014, used with permission)

1.4.2 Probabilistic models

Probabilistic, or stochastic, models draw random samples from probability distributions to initiate variables. Consequently, each execution of the model produces a different output. The alternative, deterministic, approach fixes initial values, e.g., at the mean value, so that any given initial state always produces the same output.

Muirhead, Elliott and Monaghan (2011) used a Monte Carlo (stochastic) approach to model FIO contamination of streams by dairy cows and ducks. They simulated a two-pond effluent system in the Toenepi catchment in New Zealand. A number of distributions were used (Table 1.2):

Table 1.2: Examples of sampling distributions for variables (Muirhead, Elliott & Monaghan 2011, used with permission)

Distribution	Examples of use
Log-normal	Concentration of *E. coli* in cow and duck faeces, Flow rate in the stream, *E. coli* in-stream attenuation ratio over a kilometre
Triangular	Wet weight of an individual cow pat
Poisson	Number of ducks living in the stream
Normal	Proportion of cows in the herd defecating in the stream at each crossing
Discrete	Proportion of days per year that cows have access to the stream

The *E. coli* concentration of the simulated stream output median showed good agreement with the measured one. However, the range of concentration values was larger for the modelled one which the authors attributed to, "averaging of spatial variability at the larger scale". A sensitivity analysis of the simulation system suggested that farm inputs of $<10^6$ *E. coli* ha^{-1} day^{-1} would not affect the concentrations in water leaving the farm.

The choice of distribution needs consideration to ensure it is fit for purpose, as defined, for example, by a set of statistical measures. Medda and Bhar (2019) investigated stochastic models to represent seasonal flows in single-site (single river) and multi-site (a river and its tributaries) systems. They sought to represent the stream flow of an existing river and another with a tributary. Models of the latter need to account for the cross-correlation (between two sites in the same month) inherent in multi-site systems. Normal and gamma distributions were sampled in the generation of the stream flows. Mean, standard deviation and serial correlation (between successive months) were well represented by both single- and multi-site models when applied to either river system and using either distribution. However, if skewness was to be preserved, the gamma distribution had to be used and the multi-site model was required to account for cross-correlation.

1.4.3 Mechanistic or process-based models

A mechanistic or process-based model represents each component of an environmental system with a corresponding variable and processes that are intended to reflect the behaviour of the natural system (Meineri et al. 2015).

Coffey et al. (2010) modified the Soil and Water Assessment Tool (SWAT) to study the effect of changes in land use practices on water quality in a catchment. The model incorporated data for topology, hydrology, climate, land use and *E. coli* concentrations. Land use scenarios were defined for grazing, fertiliser use and on-site wastewater treatment. Good results were obtained for

flow simulation (R^2 = 0.83) but the modelled *E. coli* predictions were variable (R^2 = 0.68). Nevertheless, the model was sufficiently sensitive to some bacteria settings (e.g., variation across manure types and soil adsorption) to provide useful land management advice. For example, limiting the application of manure to periods of low rainfall. The authors identified a need for more reliable input data for many variables – fertiliser application, distribution and loading of wastewater treatment systems, livestock access to streams, wildlife contribution, etc.

Whitehead et al. (2016) developed a process model to represent the effects of land use on pathogen contamination (as measured by total coliforms (TC)) in the Thames catchment of the United Kingdom. A schematic view of the model can be seen below (Figure 1.3).

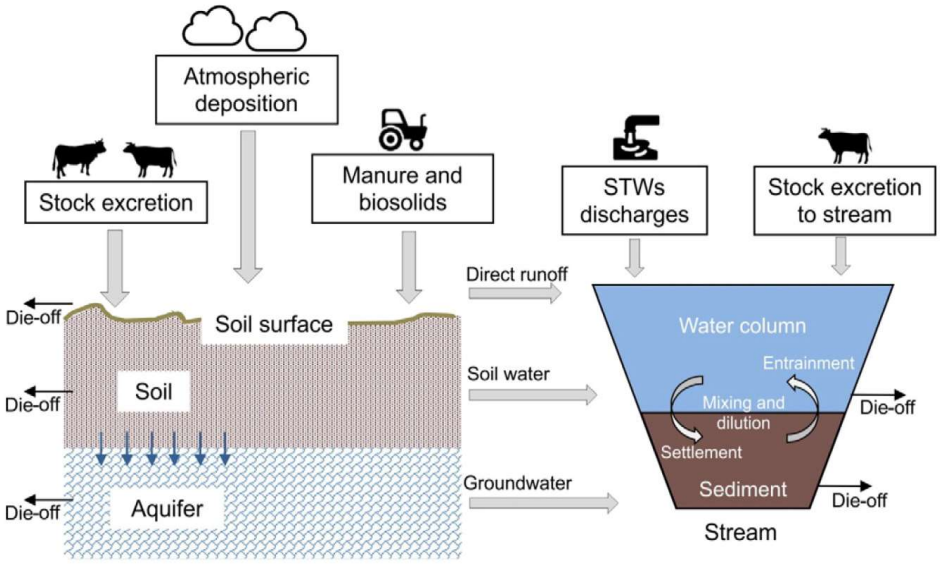


Figure 1.3: Pathogen process model of the Thames catchment (Whitehead et al. 2016, used with permission)

Each of the entities in the model was described by differential and rate equations that were solved numerically. The results were incorporated into a modification of a catchment model previously used for studies of nutrients and other contaminants. The model generated a reasonable representation of the observed daily flow (R^2 = 0.67). However, as in the previous example, the authors acknowledged that applying the model to the Thames catchment was a complex task, hampered by a lack of quantitative microbiological data.

Mechanistic models use a “bottom up” approach. They seek to simulate underlying interactions and the temporal and spatial features of the observed system. The demands for an adequate amount of reliable input data can be substantial. The size and the complexity of relationships in mechanistic models can create difficulties for any attempt at parsimony, although simplification is possible if some model components are replaced with constants (Cox, GM et al. 2006). Contrasted with this, regression techniques such as MARS start from a more agnostic position, teasing out the minimum number of parameters and relationships that comprise a parsimonious model by analysing accumulated observations empirically.

1.5 Description of the Mount Lofty Ranges precinct

The MLR is subject to a variety of uses such as grazing, horticulture, forestry, urban or rural living and tourism. There are over 20,000 properties that are subject to the oversight of nine local councils (EPA 2007, p. 3).

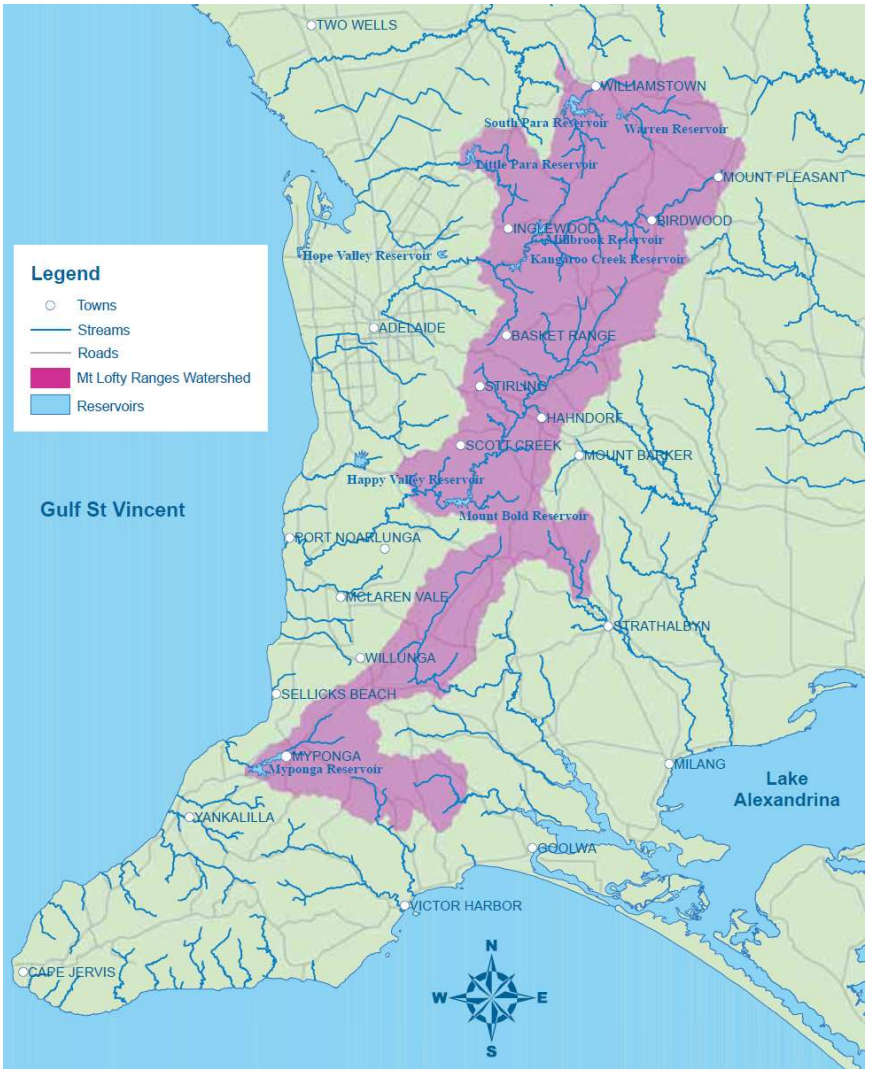


Figure 1.4: Mount Lofty Ranges watershed (EPA 2005, used with permission)

Characteristics of the MLR watershed (EPA 2004, 2005, 2019):

- it supplies about 60% of Adelaide's water
- about 90% is privately owned
- there are over 50,000 residents
- it covers 1640 km^2
- most of the runoff (about 90%) occurs between July and September
- the yield-to-catchment ratio is relatively low
- 25% of wastewater is treated on-site.

There are over 15,000 private dams in the Western MLR and 2,000 water allocation licences have been granted for the area. The Adelaide and MRL region contributes 17% of South Australia's agricultural production (Daniels & Good 2015).

This study focuses on the drinking water catchments of six MLR water supply systems (Figure 1.5).

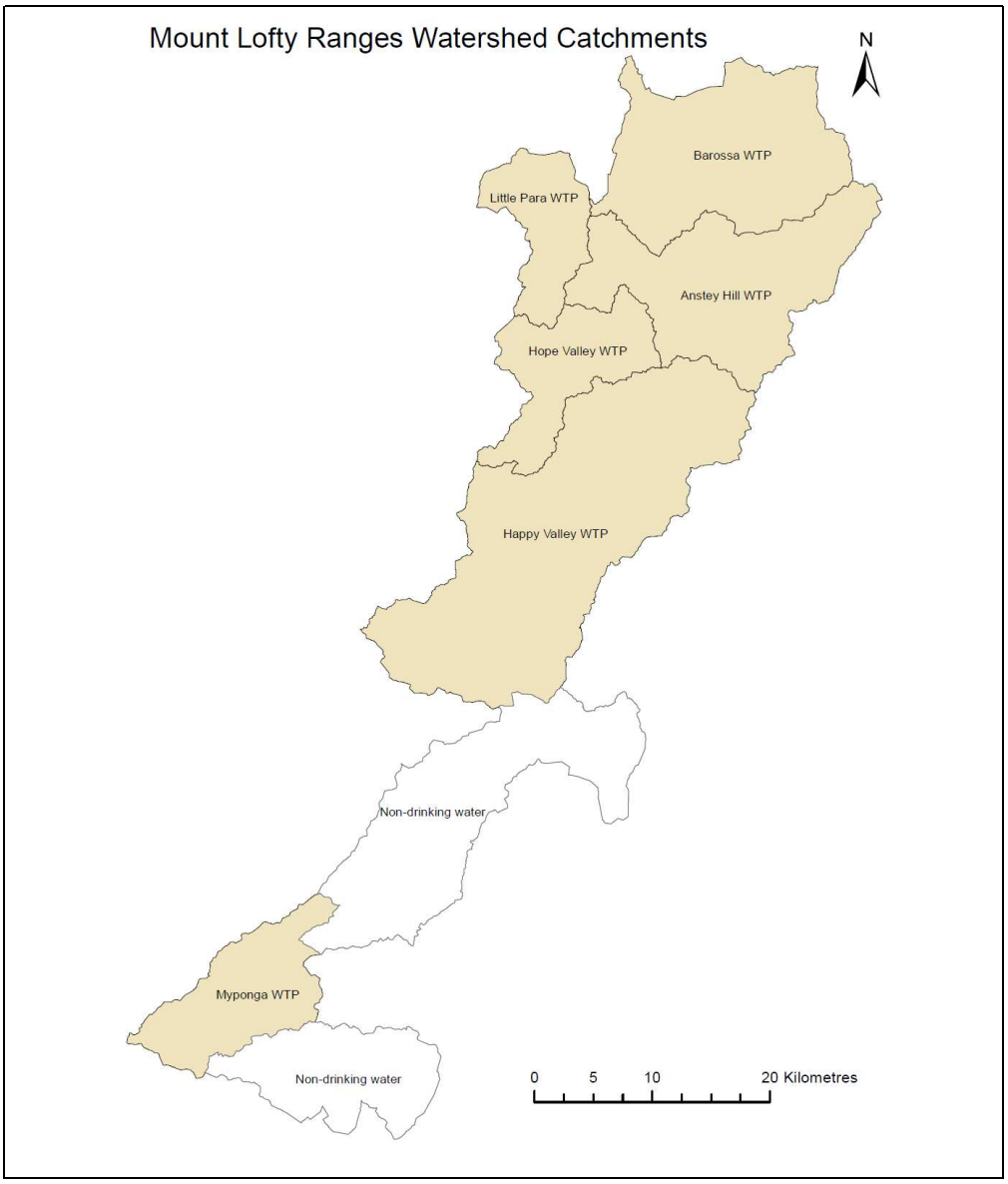


Figure 1.5: Drinking water catchments in the Mount Lofty Ranges

Bradley and Billington (2002) described a trial to accumulate land cover and land use information in the MLR. The trial focused on a region around Lobethal that displayed a wide range of land cover and use. The trial helped to refine the methodology and classification approaches used to extract information from 1:20,000 aerial photographs. Through the efforts of a number of government departments and other agencies, it is now possible to access a comprehensive range of land cover data, not just for the MLR, but for the whole state (DEW 2019; Willoughby et al.

2018). The dataset comprises 55 land cover layers, one for each land cover class plus supplementary confidence and most likely layers. The cell/pixel size for these data is 25m.

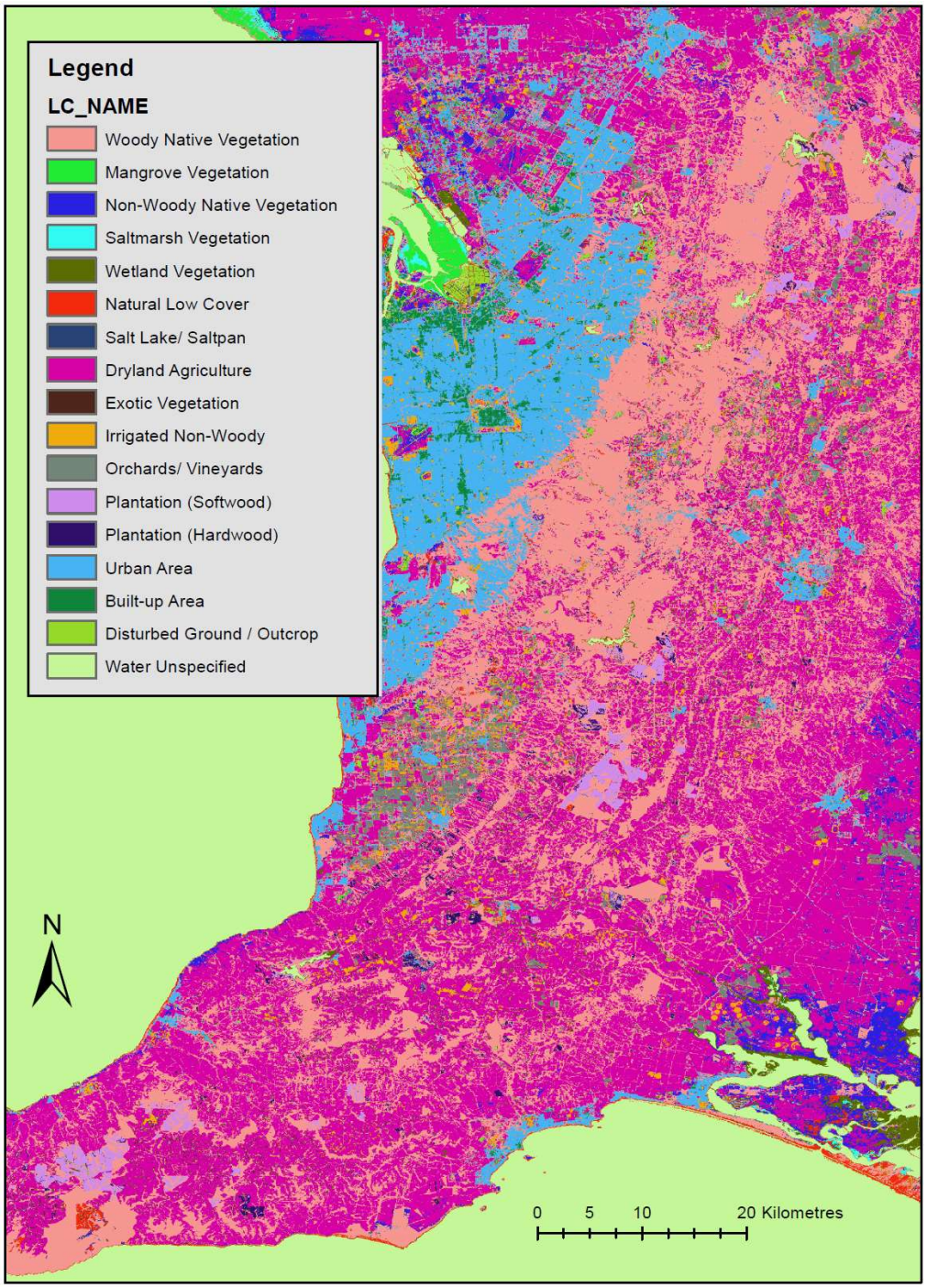


Figure 1.6: SA land cover model (2010 - 2015) (DEW 2019, used with permission)

Figure 1.6 shows a sample of the South Australian land cover information (DEW 2019). The large blue area is the urban region of Adelaide. To the East is part of the Mount Lofty Ranges that is the subject of this project. It shows a preponderance of dryland agriculture and woody native vegetation.

1.6 Risk studies in the Mount Lofty Ranges

Packer (2010a) applied the ADWG and other risk management frameworks and standards across the MLR. This study sought to evaluate risks associated with all land uses and canvassed means of responding to those risks. An extensive table summarised the approach by listing:

- source of hazard, e.g., vegetation clearance
- hazardous event, e.g., infrastructure failure such as seepage
- hazard, e.g., pathogens
- point and/or diffuse source
- reason, e.g., bare ground
- solution, e.g., development planning regulations
- cost/benefit/priority

This information was then used as input to a source catchments modelling framework (Packer 2010b).

An Australia-wide view adopted a similar approach to the enumeration of hazards, events and sources (Miller, Guice & Deere 2009). However, it provides more guidance on determining levels and impact of risk and uncertainty. For example Table 1.3 outlines the source information relating to each risk tier level and the effort required to address it.

Table 1.3: Explanations of risk tier level (edited for microbial content only) (Miller, Guice & Deere 2009, p. 29, used with permission)

Risk Tier Level	Source vulnerability	Water Quality risk microbial	Political or social issues	Downstream barriers present	Resourcing requirements
1	Source recharge area is well protected; source itself has good dilution and detention barriers.	Risk of raw water contamination by microbial parameters is low	Limited to none, few stakeholders	Source has robust downstream barriers that address all known raw water risks	Desktop study with supervisory support from project team
2	Source recharge area having some protection, source itself has good dilution and detention barriers, but they can fail during high risk events.	Risk of raw water contamination by microbial parameters is moderate	Some local issues in past and predicted occasionally for the future	Source has robust downstream barriers that address most raw water risks, except under event conditions	Preliminary desktop study and then verification through workshop-based process
3	Source recharge area having only limited effective protection, source itself having good dilution and detention barriers, but they can fail during high risk events.	Risk of raw water contamination by microbial parameters is high	Local and/or state based political and social issues in the past and anticipated for the future	Source has robust downstream barriers that address most raw water risks, except under event conditions	Workshop-based process – including external stakeholders (if required)

According to these criteria a small community may be serviced by an isolated source with no known issues. Current land uses pose minimal risk. This scenario would be assigned a risk tier level value of 1 and be addressed by a desktop study. Conversely, a large metropolitan community serviced by a source with some reportable events each year and questionable barrier integrity would score 3. This would be assessed by means of workshops and a full application of the risk assessment process.

Swaffer et al. (2018) carried out a Tier 2 QMRA (quantitative microbial risk assessment) on datasets obtained from monitoring stations in the Little Para and Anstey Hill catchments. They determined that one µDALY (disability-adjusted life year) could be achieved with "significantly lowered" treatment in most sub-catchments. For two of the monitoring sites, an increase in treatment was indicated.

Dooley et al. (2011a, 2011b) studied the risks to the quality of water in the MLR's potable water supply and aquatic ecosystems. They identified hazards and their likelihood and consequences for each land use type. Likelihood was scored 1 (rare) to 5 (almost certain) and consequences were scored 1 (insignificant) to 5 (catastrophic). Risk was defined as a function of likelihood and consequences and was summarised in Table 1.4.

Table 1.4: Risk matrix (Spies and Woodgate (2005, used with permission) cited by Dooley et al. (2011a, p. 9))

LIKELIHOOD	CONSEQUENCE				
	Insignificant (1)	Minor (2)	Moderate (3)	Major (4)	Catastrophic (5)
(5) Almost certain	S	S	H	H	H
(4) Likely	M	S	S	H	H
(3) Possible	L	M	S	H	H
(2) Unlikely	L	L	M	S	H
(1) Rare	L	L	M	S	S

H High risk	– immediate action required
S Significant risk	– high-level management attention needed
M Moderate risk	– management responsibility must be specified
L Low risk	– manage by routine procedures

Care should be taken when applying this table to ensure that the interval between the various steps on both the consequence and likelihood scales are commensurate, as mathematical rigour must be employed when multiplying out the table to determine risk. In practice, the ratings are often assumed to be logarithmically spaced, so that risk = impact + likelihood.

Raster files of 5m cell size were developed representing likelihood and consequence for each land use type. Each cell of the likelihood and consequence rasters was allocated a value of 1 – 5 according to the ratings in Table 1.4. The overall risk hazard raster was determined by multiplying the likelihood values by the consequence values (Dooley et al. 2011b, p. 5). An example of the output of this process is shown in Figure 1.7.

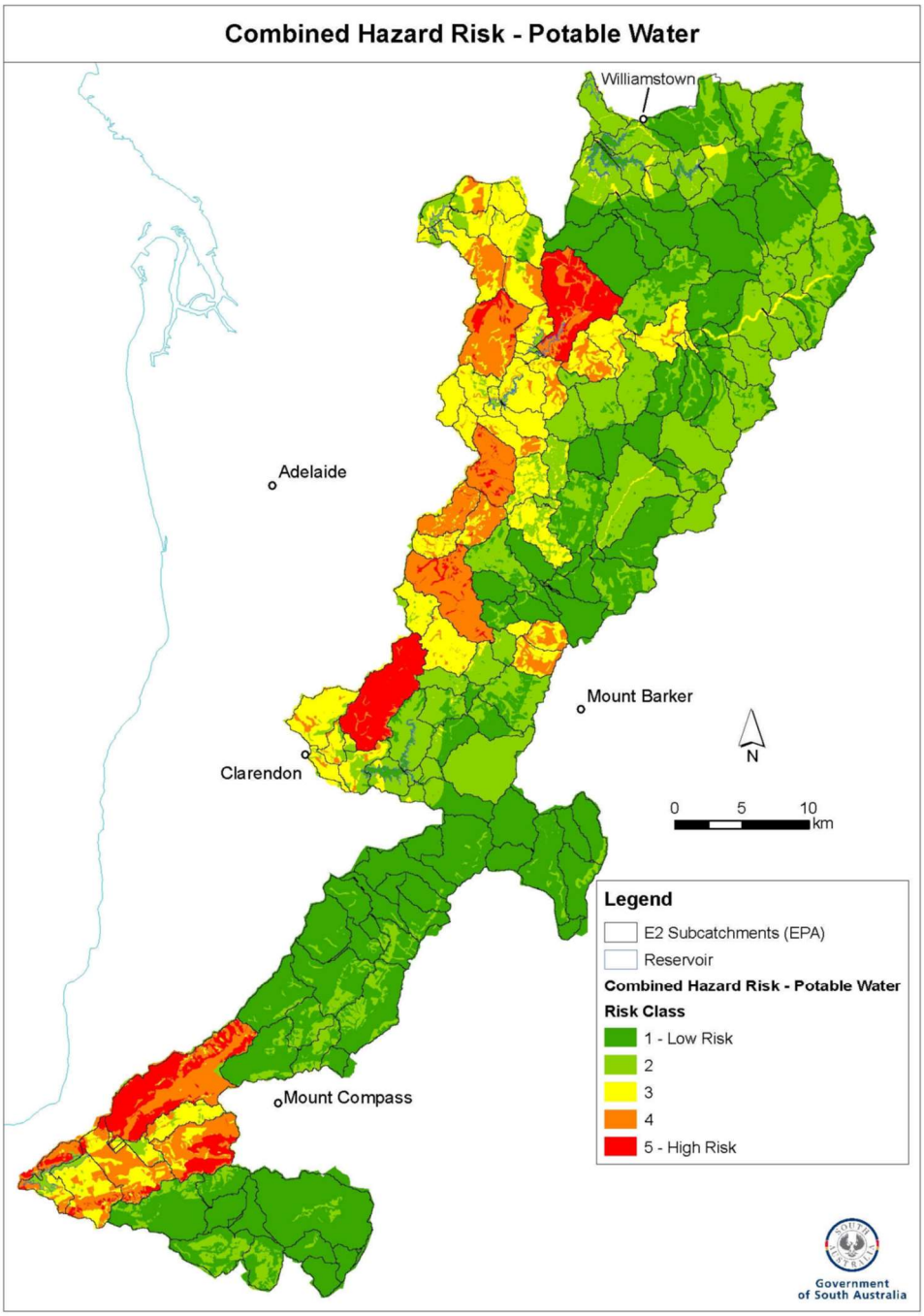


Figure 1.7: Consequent (risk) ratings for the nutrient hazard to potable water (Dooley et al. 2011a, p. 19, used with permission)

Swaffer (2014) and SA Water Corporation (2014) adopted a different approach to the determination of the risk value derived from each land use type. They reported that the maximum stocking rate in the MLR is estimated to be 20 dry sheep equivalents (DSE) (McLaren 1997) and that breeding cattle presented the highest risk on grazing land of 12 DSE (Swaffer 2014, p. 13). (These numbers appear to be a little low – AMLRNRMB (2017, p. 57) suggest a range for breeding cattle of 9 – 25 DSE). Assuming:

- 1.7 breeding cows/ha

- grazing land fully occupied by breeding cows
- manure production of 20kg/animal
- 4000 oocysts/g of manure
- 15% of cows infected with *Cryptosporidium*

a worst-case risk for *Cryptosporidium* could be determined. This was scored as 100. Given that:

- not all grazing animals are breeding cows
- stocking rates vary for different animals
- *Cryptosporidium* infection rates vary
- total manure production and oocyst loadings vary,

it is possible to calculate a likely risk for all animal sources of *Cryptosporidium* for grazing land use and express this as a proportion of the breeding cattle maximum. Calculations like these led to infection risk ratings for different land use types (Swaffer 2014, pp. 13-8).

The studies by Swaffer (2014) and SA Water Corporation (2014) were also based on a spatial approach with 5m cell size. This dimension was chosen in anticipation of modelling risks inside and outside of fences, e.g., protecting watercourses. The focus was on modelling *Cryptosporidium* risks to water quality across the MLR and was referred to as a Catchment Risk Analysis (CRA). The modelling environment was a geographical information system (GIS) which produced a risk raster such as can be seen in Figure 4.3 below. In combination with sanitary surveys, the CRA satisfies the requirements of a Tier 1 assessment (Swaffer 2014, p. 43). This work formed the basis of the project described in the following chapters.

1.7 Research agreement project plan

A research agreement was negotiated between Flinders University and South Australian Water Corporation. The agreement included the outline of a project plan (Appendix A). Briefly, this plan guided the research described below to:

- develop a Python application to replicate the existing pathogen CRA
- use the Python application to review and revise the current pathogen CRA for at least one drinking water catchment, but preferably all of them, in the MLR
- undertake a sensitivity analysis
- verify the outputs using end-of-catchment data
- apply the experience gained above to nutrients, sediments and pesticides
- update literature
- document the steps from input spreadsheet data to output risk raster files.

1.8 Scope

This thesis describes work done under the auspices of SAW to visualise risks of water contamination in the MLR using GIS tools. Experience gained during the SAW phase of the work revealed that the GIS implementation could be awkward to use for a number of reasons. In particular, it is a large application not well-suited to the quick turn-around and record-keeping associated with "what-if" or sensitivity analyses. It was also developed with just *Cryptosporidium* contamination in mind. Adaptation to other pollutants may not have been a simple exercise.

A number of assumptions underlie the SAW work:

- some variables, e.g., reservoir impedance, have been applied as catchment-wide averages
- the ephemeral systems of the MLR can be represented by annual averages
- measurements at a monitoring station represent the entire catchment
- there is a linear relationship between calculated risk and end-of-catchment observations.

This study sets out to address these and other issues by rewriting the system as a Python application that offers:

- an uncomplicated interface
- data maintenance
- simplification of the model definition
- record-keeping associated with repetitive model execution
- extension to other pollutants
- enhancements to the model
- expanded reporting options.

The development of the Python application and the evolution of the model are described in this thesis.

2. METHODS

2.1 SAW prior work

In previous work, SAW described the development of a model to assess the risk of *Cryptosporidium* contamination of drinking water catchments in the Mount Lofty Ranges (MLR). This work was described in two unpublished reports, a summary (SA Water Corporation 2014) and a detailed report (Swaffer 2014). These documents and an agreement (Appendix A) between the University and SAW form the basis of the current project.

2.1.1 SAW workflow

The SAW workflow is described in:

- Swaffer (2014) – the business processes workflow (Appendix B), and
- SA Water Corporation (2014) – The ArcMap (ESRI 2018) processes workflow (Appendix C). The SAW work was carried out in ArcMap v10.3 but the current project employed v10.6.1. Note: ArcMap is also known as ArcGIS Desktop.

Briefly, SAW maintains data in Excel spreadsheets (Microsoft Corporation 2016) about pathogen sources and environmental factors that affect the distribution of the pathogens. Sanitary surveys (Swaffer 2014, p. 20) are conducted to observe changes in land use or other environmental factors. These observations and a literature review were used to estimate maximum and likely risks of contamination associated with each land use type. The likely risk estimates are scaled against the maximum resulting in values the range 0 – 100. These values are imported into shape file attribute tables in ArcMap (Swaffer 2014, pp. 12-9).

Figure 2.1 is an example of how the source data are managed in the shape file environment. It shows the shape file outline of the MLR study region with the boundaries of irrigated areas displayed and information about these, such as risk estimate, shape area, etc., in the attribute table. For the rows showing in the attribute table, the risk estimate, mlrirrigat is 21.5.

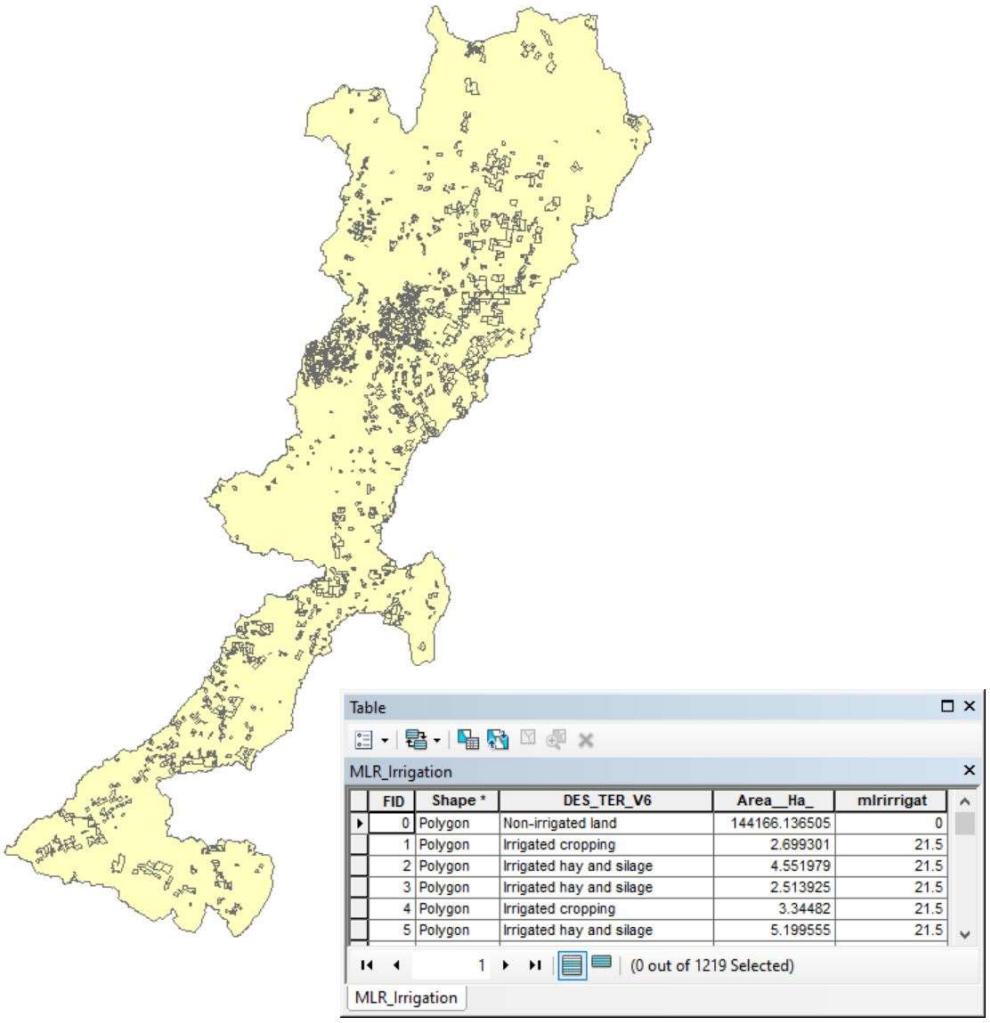


Figure 2.1: ArcMap shape file, MLR_Irrigation and attribute table for the irrigation variable mlrirrigat

In general, variables of interest in the tables are each exported into raster files (cell size 5 x 5 m) for subsequent processing in the model computations. A few raster files, such as for slope, streams and the Digital Elevation Model (DEM), were sourced separately (but also from SAW). The variables used in the SAW model, and hence raster file names, are listed in Appendix D.

2.1.2 SAW model definition

The original SAW model was constructed from terms of the form:

*W * R*

Where,

W is a weight estimated by SAW professional staff

R is the risk for a given variable as described above

The value of weight is derived from a budget of 1.0 allocated across all variables (Swaffer 2014, pp. 31-2). An example of SAW's original model definition follows (Figure 2.2). This is the form in which it would have been entered into ArcMap's Raster Calculator to generate a single risk surface for the entire MLR catchments region.

```
((0.15 * " pathgraz2") + (0.15 * " pathirrpas2") + (0.15 * " pathintapr") + (0.05 * " pathresid ") + (0.075 * " pathnatveg") + (0.05 * "pathwaste") + (0.025 * "pathwwfail") + (0.025 * "pathresdist") + (0.025 * "pathstrdist") + (0.025 * "mlrrain") + (0.025 * "mlrslope") + (0.025 * "pathgulero") + (0.025 * "pathwatero") + (0.025 * "pathirrigate") - (0.15 * "mlrbuffer") - (0.05 * "pathfencing")) * "mlrsaw"
```

Figure 2.2: Defining the model in Raster Calculator (variables defined in Appendix D)

Most terms add to the risk. The exceptions are fencing and buffers (Figure 2.2, pathfencing and mlrbuffer respectively). Fences can be used to exclude stock from stream beds, thereby reducing the risk level. Buffers represent secondary storage within which pathogens can complete their life cycle and die before reaching a primary reservoir. Again, the risk is reduced. The multiplicative term, mlrsaw, is simply a mask that excludes non-drinking water catchments from this study.

2.1.3 SAW model optimisation

Figure 2.2 represents the end point of the SAW development in 2015. A number of sensitivity studies ("*sens1*" up to "*sens6*") had been performed prior to this. For example, in the *sens1* study, all variables (or layers in the raster calculation) were given equal weighting. In *sens3*, the hydrological terms rainfall and slope were considered to have greater influence on propagating pathogen risk. In the last analysis, *sens6*, land use is most important for pathogen generation while fencing and buffers were applied to reduce the risk levels (Swaffer 2014, pp. 34-8).

At the time the original SAW modelling work was completed, experimental studies were revealing that most oocysts were not infective. An overlay function was applied to the *sens6* output to reflect this. The *sens6* output was adjusted as follows in Raster Calculator (Swaffer 2014, pp. 40-1):

$$sens6_{infect} = 0.2 * sens6 * infectivity\% + 0.8 * sens6 \quad \textbf{(Eq. 2.1)}$$

2.1.4 SAW verification

Monitoring stations are available for each catchment in the MLR. These provided "end-of-catchment" water quality measures (described below in 8.1) that could be used to assess the veracity of the model. Least squares regression analyses were undertaken for this (Swaffer 2014, pp. 38-41). The model's ArcMap Zonal Statistics mean output for each catchment was plotted against the end-of-catchment water quality data. The model was assumed to be linear.

2.2 New approach to workflow

A number of limitations were identified in the approach described above:

- limited support for "what if" sensitivity studies
- difficulty keeping a record of a study series
- single pollutant focus

These limitations and other issues are addressed in the current study.

2.2.1 Spreadsheet use

The use of spreadsheets to maintain source data has been retained. A new spreadsheet-based model definition simplifies the task of allocating the weights budget across the variables (Table 2.1). Note that the same variables (renamed here as described in the naming conventions section below) and weights still apply.

Table 2.1: Extract from the spreadsheet model definition

Variable /Layer	Weight	Operator
presid	0.050	+
pwaste	0.050	+
pwwfail	0.025	+
tbuffer	0.150	-
tfence	0.050	-
tgulero	0.025	+

The familiar device of a spreadsheet adds the capability of recording different weight settings for each execution of the model. This enhances the ability to quickly modify weight settings in a model and follow the trail of these changes during a sensitivity study. A facility was also added to store a user reference number that appeared on the standard risk surface report.

2.2.2 Model development in Python

The SAW workflow has been abstracted from ArcMap to a simple Graphical User Interface (GUI) by means of a Python V2.7.15 (Python Software Foundation 2019) script (Appendix E). Facilities provided by the Python application (relevant function definitions are shown in parentheses) include:

- initialisation:
    - present the GUI (initGUI)
    - read the list of variable/layer names from a text file
    - receive choice of pollutant (pathogen, nutrient, pesticide or sediment) or transport from the user (initGUI)
    - read the relevant model from a spreadsheet (modelUpdate)
- receive an action choice and implement it:
    - update a shape file from the spreadsheet source (shapesUpdate)
    - generate one or more raster files from a shape file (rastersUpdate)
    - generate a risk surface from the raster files collection (risksUpdate)
    - generate the standard risk surface and Zonal Statics reports (reportsUpdate)
    - generate a detailed sub-catchment report (detailedAnalysis).

Figure 2.3 illustrates how data flow through the Python implementation from spreadsheet sources to the final reports.

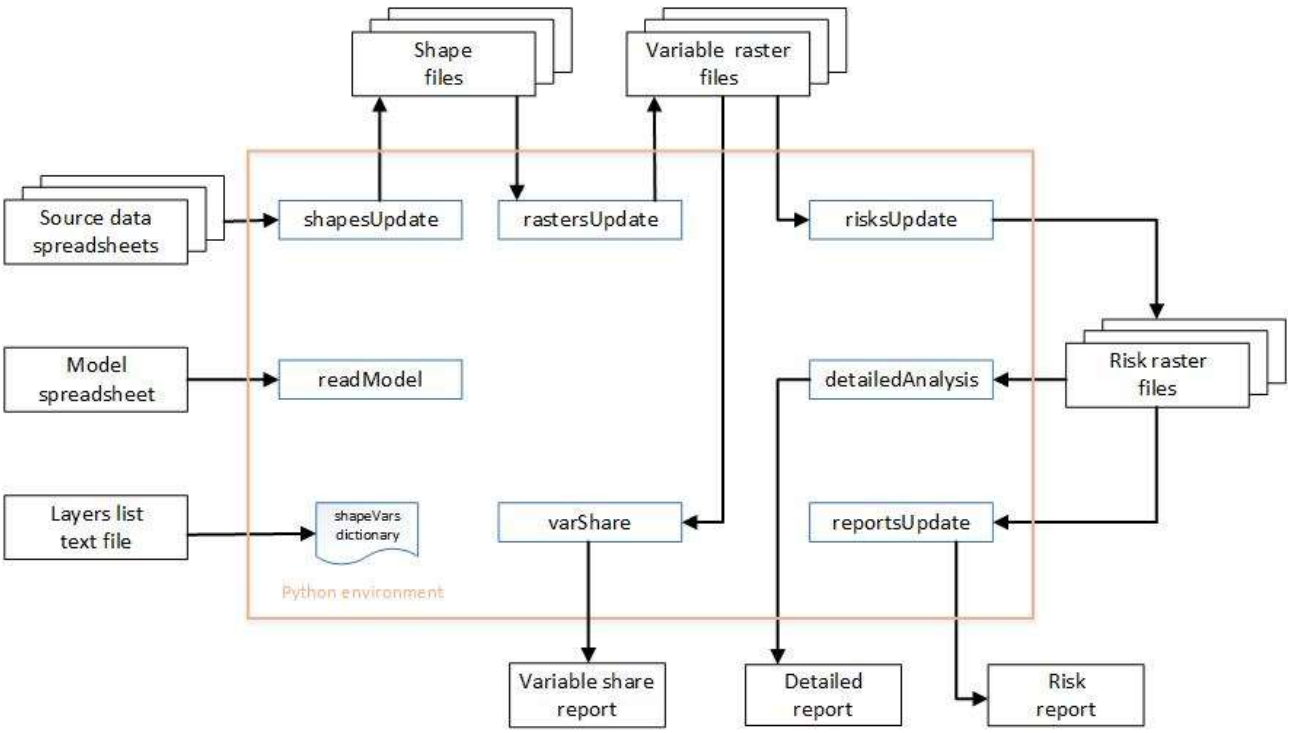


Figure 2.3: Flow of data through the Python environment

2.2.3 Naming conventions

The names of the variables have been modified to distinguish the classes of variables, i.e., pathogens, nutrients and transport. In Table 2.1 above, the variables prefixed with a 'p' relate to pathogen sources while those prefixed with a 't' influence the transport of the pathogens.

The risk surface names have also been changed to reflect the extension of the application to include different pollutants and to support the version control requested by users. Hence, the risk surface raster file name "nriskv005" refers to the fifth version of a nutrients risk raster. Similarly, "MLR_Landusepv001.shp" refers to version one of the pathogens land use shape file. In one version of the watershed studies, the variable names had a 'w' appended, e.g., pgraz became pgrazw. Later, a separate folder structure was used instead.

The risk surface report now includes all the variables and their weights in a table. The table also includes the user reference number.

2.2.4 Selected data structures

The key data structures for the Python implementation are listed below. The shapeVars dictionary is used to look up raster names associated with each shape file. The OPTIONS array lists shape file names for use in a drop-down menu selection for shape or rasters file updates. The rcDict dictionary holds the model variables, weights and operators read from the spreadsheet instance of the model.

- shapeVars – dictionary with shape file names (key) and corresponding tuple of raster layer(s) (value), e.g.,

  shapeVars = {'MLR_Stream_Distance': ('tstrdist',), 'MLR_WW_Risk': ('pwwfail',), …}
- OPTIONS – list of shape file names, e.g.,

  OPTIONS = ['DWC_MLR', 'MLR_Buffer', …]
- rcDict – dictionary of raster layers (key) and list of corresponding weight and operator (value) read from the model spreadsheet, e.g.,

  rcDict = {u'tstrdist': [0.025, u'+'], u'tresdist': [0.025, …]}
- Raster files – ESRI GRID format, 5 x 5m cell size,
  Geographic Coordinate Reference: GCS_GDA_1994
  Projection: GDA_1994_Lambert_Conformal_Conic XY Coordinate System.
- Shape files – ESRI Polygon Feature Class
  Projected Coordinate System: GDA_1994_South_Australia_Lambert,
  Projection: Lambert_Conformal_Conic

2.2.5 Reporting

Standard reports

Standard reports include an overall MLR risk surface annotated with catchment boundaries, version number and a table of raster names, weights and the reference number that applied to the model run. Zonal Statistics output for both catchments and sub-catchments are also generated.

Detailed ad hoc reports

The availability of a GUI presentation introduces other reporting possibilities. An example developed in this project uses a series of drop-down menus to select a catchment, or sub-catchment. The application then generates a report for the selected area that illustrates fenced and high-risk areas in relation to a stream network.

2.3 Adapting the model

The Python implementation of the SAW *sens6* workflow represents the baseline for the current project. This initial development is referred to below as "*sens7*". It facilitated subsequent modifications to the model, including:

- explicitly dealing with the absence of any pollutants in a given cell
- distinguishing between watersheds and catchments at the monitoring sites
- substituting a flow rate risk in place of the slope risk
- adaptation for other contaminants.

Further, the influence of each variable on the overall risk outcome could be investigated. As both *sens6* and *sens7* add all terms in the model equation, they are referred to as "all risks" versions below.

2.3.1 Account for the absence of pollutants

The initial land use table that houses the seven pathogen sources had about 10,000 rows at the beginning of this study. It has subsequently grown to over 44,000 rows. This has implications for the model definition as many of the rows don't list a contaminant source, e.g., because they refer to roads, quarries, etc. However, as Figure 2.2 illustrates, even if all the pathogen terms are zero, the model still determines a risk (of pathogen contamination) from the transport terms. Hence, the model has been modified from:

pathogen risk = source risk + transport risk

to:

***if** source risk > 0*
then
pathogen risk = source risk + transport risk
else
pathogen risk = 0

This version of the model was allocated the name "*sens8*" and a qualifier was added to emphasise that terms with zero source have been excluded from the model's output.

2.3.2 Contribution of each variable to total risk

Other supporting utilities were also developed during this project. One of these determined the total contribution of each variable to the overall risk for the entire MLR (Appendix F). This development was prompted by the discovery that the fencing and buffer variables appeared to carry too much weight in the model, to the extent that significant negative total risk values were appearing. This issue should properly be addressed by the catchment scientists at SAW. In the interim, the risk values for the buffer variable were reduced as a workaround that permitted continued development.

2.3.3 Watersheds vs catchments

Other Python scripts developed outside the GUI application introduced a new approach to the modelling of water as a transport medium (Appendix G). This was in addition to the rainfall. The intent, initially, was to address the static nature of the slope contribution by substituting a flow rate (see below). The basic model used slope to reflect the flow of water. The difficulty with this is that a cell with a given slope, e.g., three degrees, made the same contribution to the final risk calculation as any other cell with the same slope elsewhere in the catchment. This is a problem because it does not take account of the flow of water from the contributing area upstream from the cell.

The application of a flow rate entailed the determination of the watersheds that discharged towards the monitoring stations where contaminants were sampled. It emerged that for only two of the sampling points (Happy Valley and Hope Valley) did the watershed coincide with the catchment. A new approach was therefore developed that focused only on the watersheds related to each sampling point rather than the entire surrounding catchment. This watersheds version was called "*sens9*" and was evaluated in Raster Calculator.

During the development of the watershed views, it became apparent that the one for the Myponga region was incorrect. Comparison of the flow accumulation raster with a separate shapefile of the streams network showed a significant discrepancy. It emerged that there were errors in the underlying digital elevation model (DEM). These were resolved with the assistance of Keane (2019), enabling the expected watershed to be determined.

2.3.4 Water flow rates in place of slope

For the "*sens10*" (flow replaces slope) version, a flow rate was used in place of slope. Python scripts were developed to assist in determining the flow rates (Appendix G) while the final computations were carried out in Raster Calculator.

The Python scripts have been adapted from the interactive approach described in an on-line tutorial (ESRI 2019). Starting with the DEM, raster layers were derived for flow direction and accumulation. At this point, a separate raster was defined manually that positioned a "pour point" or outlet on the flow accumulation path nearest the latitude/longitude position of each SAW monitoring point. Further Python code then determined the watershed, slope, slope area and finally a raster layer that encoded flow rates throughout the MLR. ArcMap's Raster Calculator was used to limit this last layer to a maximum of 2m/s according to the advice in ESRI (2019).

The technique outlined in ESRI (2019) is a simplified one that excludes variations over time and discharge rate. It focuses on velocity as affected by spatial components, including slope and flow accumulation. The computation was derived from Maidment et al. (1996):

$$V = V_m * (s^b A^c) / (s^b A^c{}_m) \quad \textbf{(Eq. 2.2)}$$

Where,

V is velocity at a cell of slope s and upstream contributing area A

$b = c = 0.5$m/s

V_m is an assumed average velocity of 0.1m/s

$s^b A^c{}_m$ is the average slope area throughout the watershed.

The local slope of each cell and the area of upstream cells that flow into it are used to determine a velocity for the cell. Note that *sens10* is still an additive model.

2.3.5 Final multiplicative version

The flow rates can be very low, even zero. To account for this, the final version of the model was created, where the flow variable (tflows) was removed from the transport group and a product, rather than a sum, of the terms was generated:

*total risk = (sum of source risks) * (sum of transport risks without tflows) * tflows*

This version is referred to below as "*sens11"* or flow modifier version.

2.3.6 Other pollutants

The modular development of the Python script, together with the categorisation of source data, support adaptation of the modelling system other pollutants. Modules and data structures have been designed to work readily with the different pollutant choices. In the event, the pathogens processes were the most fully developed. Some nutrient processing was also performed in relation to Phosphorus (described below).

2.4 Verification

The existing regression analysis employed by SAW was applied in this project for comparison.

3. MODEL DEVELOPMENT - SPREADSHEET

3.1 Results – spreadsheet implementation of the model

The original model Raster Calculator version of the model can now be represented by a spreadsheet (Figure 3.1). Note: the variable definitions can be found in Appendix D.

Raster Calculator

```
((0.15 * "pgraz") + (0.15 * "pirrpas") +
(0.15 * "pintapr") + (0.05 * "presid ") +
(0.075 * "pnatveg") + (0.05 * "pwaste") +
(0.025 * "pwwfail") + (0.025 * "tresdist") +
(0.025 * "tstrdist") + (0.025 * "train") +
(0.025 * "tslope") + (0.025 * "tgulero") +
(0.025 * "twatero") + (0.025 * "tirrigate") -
(0.15 * "tbuffer") - (0.05 * "tfence")) *
"tsaw"
```

Spreadsheet

	A	B	C
1	**Layer**	**Coeff**	**TermOper**
2	pgraz	0.150	+
3	pinfect	0.000	+
4	pintapr	0.150	+
5	pirrpas	0.150	+
6	plife	0.000	+
7	pnatveg	0.075	+
8	presid	0.050	+
9	pwaste	0.050	+
10	pwwfail	0.025	+
11	tbuffer	0.150	-
12	tfence	0.050	-
13	tgulero	0.025	+
14	tirrigate	0.025	+
15	train	0.025	+
16	tresdist	0.025	+
17	tsaw	1.000	*
18	tslope	0.025	+
19	tstrdist	0.025	+
20	twatero	0.025	+

20 ...

Average: 0.150357143 Count: 16

Figure 3.1: Model definition in the Raster Calculator (L) and in a spreadsheet (R)

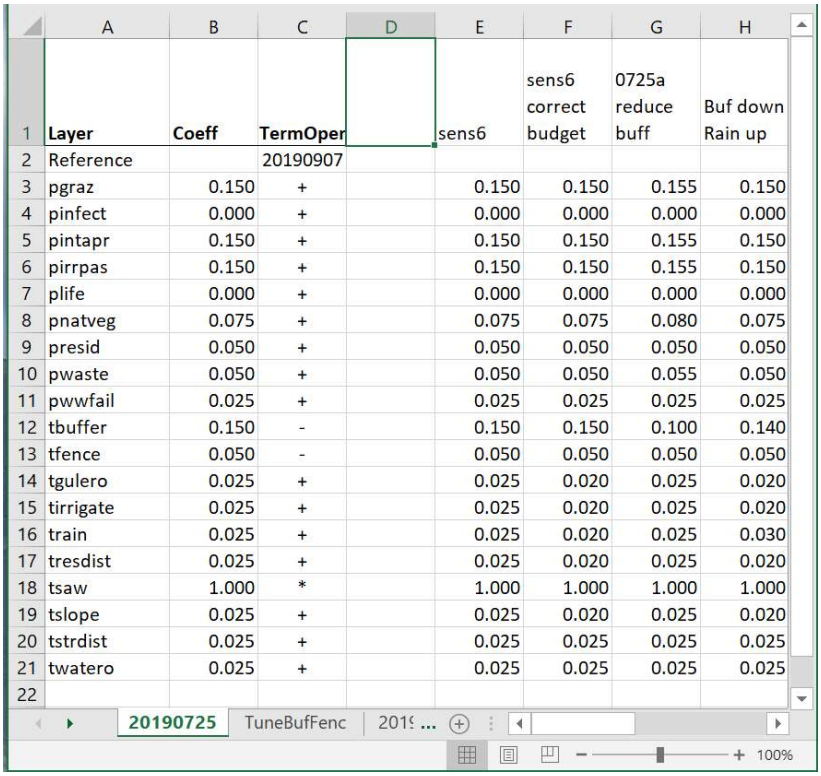
The Python application reads the first three columns of the spreadsheet comprising variable names, the weight applied to each of them and whether the variable increases or decreases the risk of contamination. Fencing, for example, can be used to exclude dairy cattle from water courses so its value (tfence) reduces the risk. Both the Raster Calculator version and the spreadsheet representation describe the same information, namely the original *sens6* (all risks) definitions.

Figure 3.1 also illustrates the categorisation of the variables in the model into two classes. The source class nominates the variables that generate the contamination. In this case, the 'p' prefix indicates pathogens. Those variables prefixed with a 't' influence the transport of pathogens. These class distinctions became important in later variations of the model.

The advent of a spreadsheet version facilitates sensitivity analyses (Figure 3.2). The current weights definitions in the second column are copied to an unused column on the right where the

weights can be altered and copied back for a new run of the model. It is a simple matter to insert temporarily a built-in SUM function at the bottom of the weights column to ensure the total weights values are held within a budget of 1.0 (excluding the drinking water mask, tsaw). Further details can be recorded in other sheets if necessary.

Figure 3.2 also demonstrates the use of a user-defined reference number. This is recorded on the main risk surface report and can be used to tie the report to the spreadsheet definition.



	A	B	C	D	E	F	G	H
1	**Layer**	**Coeff**	**TermOper**		sens6	sens6 correct budget	0725a reduce buff	Buf down Rain up
2	Reference		20190907					
3	pgraz	0.150	+		0.150	0.150	0.155	0.150
4	pinfect	0.000	+		0.000	0.000	0.000	0.000
5	pintapr	0.150	+		0.150	0.150	0.155	0.150
6	pirrpas	0.150	+		0.150	0.150	0.155	0.150
7	plife	0.000	+		0.000	0.000	0.000	0.000
8	pnatveg	0.075	+		0.075	0.075	0.080	0.075
9	presid	0.050	+		0.050	0.050	0.050	0.050
10	pwaste	0.050	+		0.050	0.050	0.055	0.050
11	pwwfail	0.025	+		0.025	0.025	0.025	0.025
12	tbuffer	0.150	-		0.150	0.150	0.100	0.140
13	tfence	0.050	-		0.050	0.050	0.050	0.050
14	tgulero	0.025	+		0.025	0.020	0.025	0.020
15	tirrigate	0.025	+		0.025	0.020	0.025	0.020
16	train	0.025	+		0.025	0.020	0.025	0.030
17	tresdist	0.025	+		0.025	0.020	0.025	0.020
18	tsaw	1.000	*		1.000	1.000	1.000	1.000
19	tslope	0.025	+		0.025	0.020	0.025	0.020
20	tstrdist	0.025	+		0.025	0.025	0.025	0.025
21	twatero	0.025	+		0.025	0.025	0.025	0.025
22								

Figure 3.2: The spreadsheet version weights manipulation and recording

3.2 Discussion of Spreadsheet Model Development

SAW staff defined a model of the risks of *Cryptosporidium* contamination in the MLR. In order to test the model, they devised a series of sensitivity studies. These studies varied the weights applied to each variable (or combinations of them) to assess how well the model performed against end-of-catchment measures of *Cryptosporidium* contamination. This task proved tedious. The ArcMap Raster Calculator is awkward to use in a repetitive series of experiments and does not provide a record of activities over time.

The current project sought to address these concerns by providing an uncomplicated interface, that simplified the user's view of model implementation and could maintain a record of model execution. These aims are achieved through a simple spreadsheet layout that preserves the basic arithmetic application of the model but adds a number of record-keeping capabilities:

- A free text reference number can be assigned by the user to each model run. This could, for example, be date related: 20190805a, 20190805b, etc.
- The weights assigned to each run can be stored in their own spreadsheet column, in sequence and with a meaningful heading, e.g., in Figure 3.2, “Buf down Rain up” is a reminder that the buffer weight was traded off against rainfall in that experiment.
- Whole series of experiments can be stored in separate spreadsheet tabs prior to a major change in strategy.

4. MODEL DEVELOPMENT – PYTHON

4.1 Results – Graphical User Interface

The workflow illustrated in Figure 2.3 has been implemented as a Python script (Appendix E) which presents the following interface to the user (Figure 4.1):

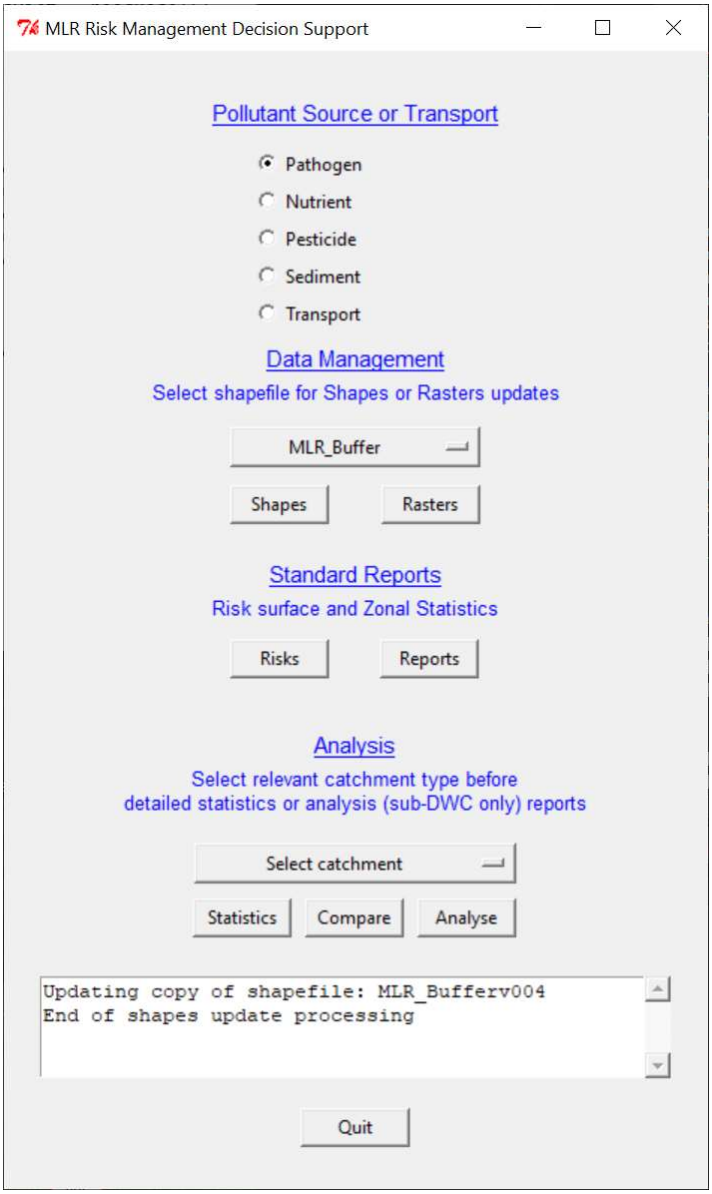


Figure 4.1: GUI interface provided by the Python script

Features that can be noted in this interface include:

- radio buttons to choose the pollutant source, or transport, for study. This is the origin of the prefix letter used in filenames (e.g., nriskv002) and for internal decision-making in the script
- drop-down menus to select data sources or catchments
- buttons to invoke the various updating functions or report generation
- a message window to keep the user informed to progress during execution.

Due to a bug in the Python implementation used for this project, the code for the Compare function had to be developed separately. The Statistics function has been set aside for now.

4.2 Results - Python implementation of the model

ESRI has made available a library, *arcpy*, that exposes much of the capability of ArcMap to the Python programmer. Specifically, it provides "map algebra" functions that mimic the behaviour of the Raster Calculator. The use of this can be found at the core of the model's execution in the function risksUpdate main loop from which the following edited script has been extracted (Figure 4.2):

```
for cR in currentRasters:
    cRName = <raster name string>
    if cRName[0] == 't':
        tSumRaster = tSumRaster + coeff(cRName) * arcpy.Raster(rastersDir + cR)
    elif cRName[0] == pollutantSrc:
        sSumRaster = sSumRaster + coeff(cRName) * arcpy.Raster(rastersDir + cR)
sumRaster = tSumRaster + sSumRaster
```

Figure 4.2: The core loop of the model execution in the risksUpdate function

Here, the weight (i.e., the coefficient obtained previously from the model spreadsheet) is multiplied by the value of the corresponding variable and added to the accumulating sum. This is equivalent to the arithmetic shown previously in Figure 3.1 above.

Although the final value of the raster calculations above is expressed as the sum of transport and source values, the logic is the same as used by SAW (Figure 3.1). It has been expressed in this way to facilitate modifications to the model described below.

The raw output of the risksUpdate computation (colourised to match the symbology used by SAW) is shown in Figure 4.3. This represents the SAW *sens6* output (now termed *sens7*), as amended by the larger land use table and reduced buffer risk values. Unfortunately, detailed comparisons with the previous work are not possible as its output is no longer available. The two white regions represent non-drinking water catchments that are not part of this study. The variable tsaw was used to mask them out.

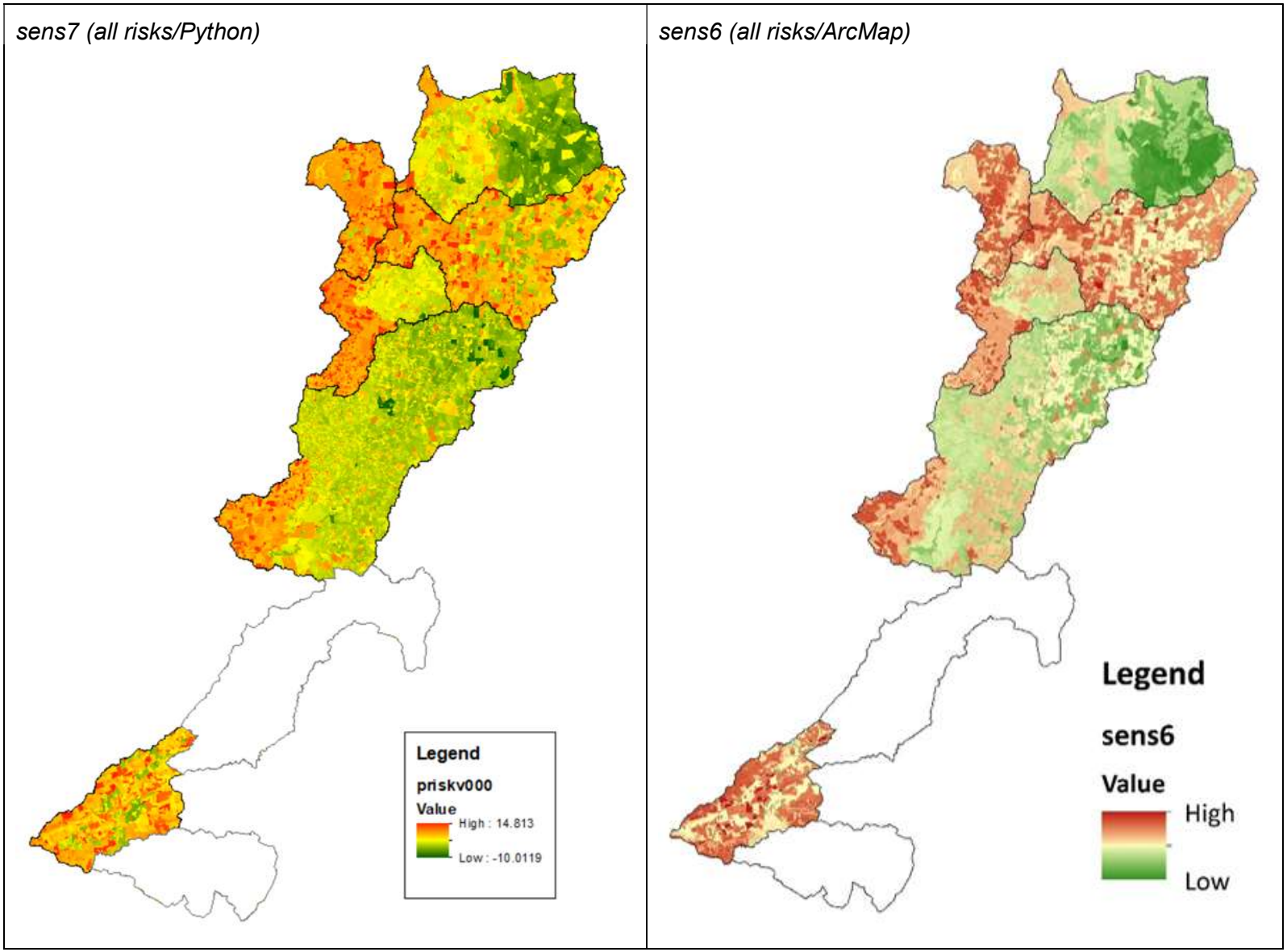


Figure 4.3: Raw output from the Python risksUpdate *sens7* computation (L) and the corresponding SAW ArcMap Raster Calculator *sens6* view (R) from Swaffer (2014)

When the reports function is invoked on the GUI, it causes the following three standard reports to be generated – risks summary, Zonal Statistics for catchments and Zonal Statistics for sub-catchments. The first, Figure 4.4, shows the summary risk surface covering all the drinking water catchments of the MLR. It illustrates the new features offered by the Python implementation, namely, version control, the table of weights and reference number.

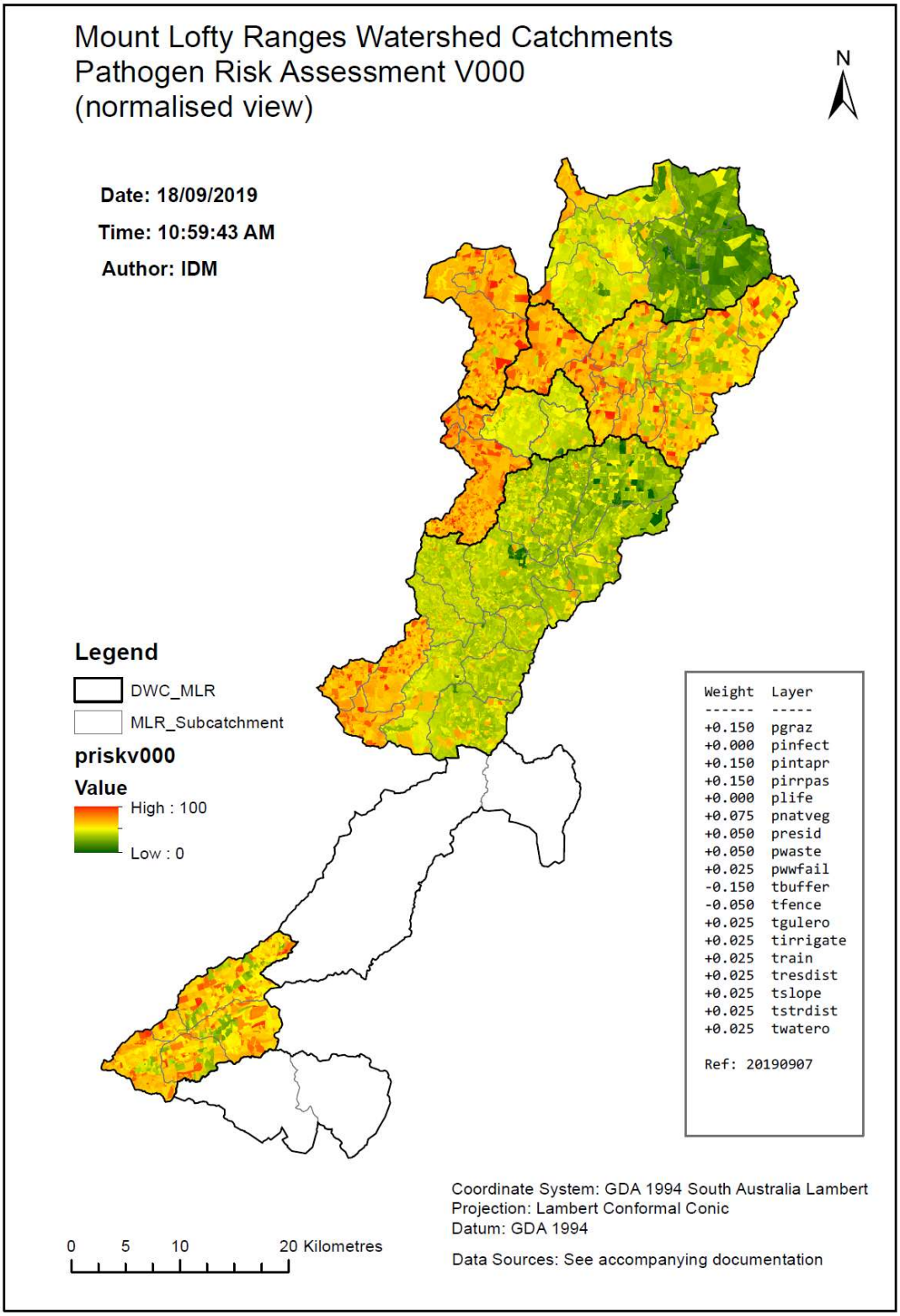


Figure 4.4: Standard reports summary all risks surface for pathogen contamination across the MLR

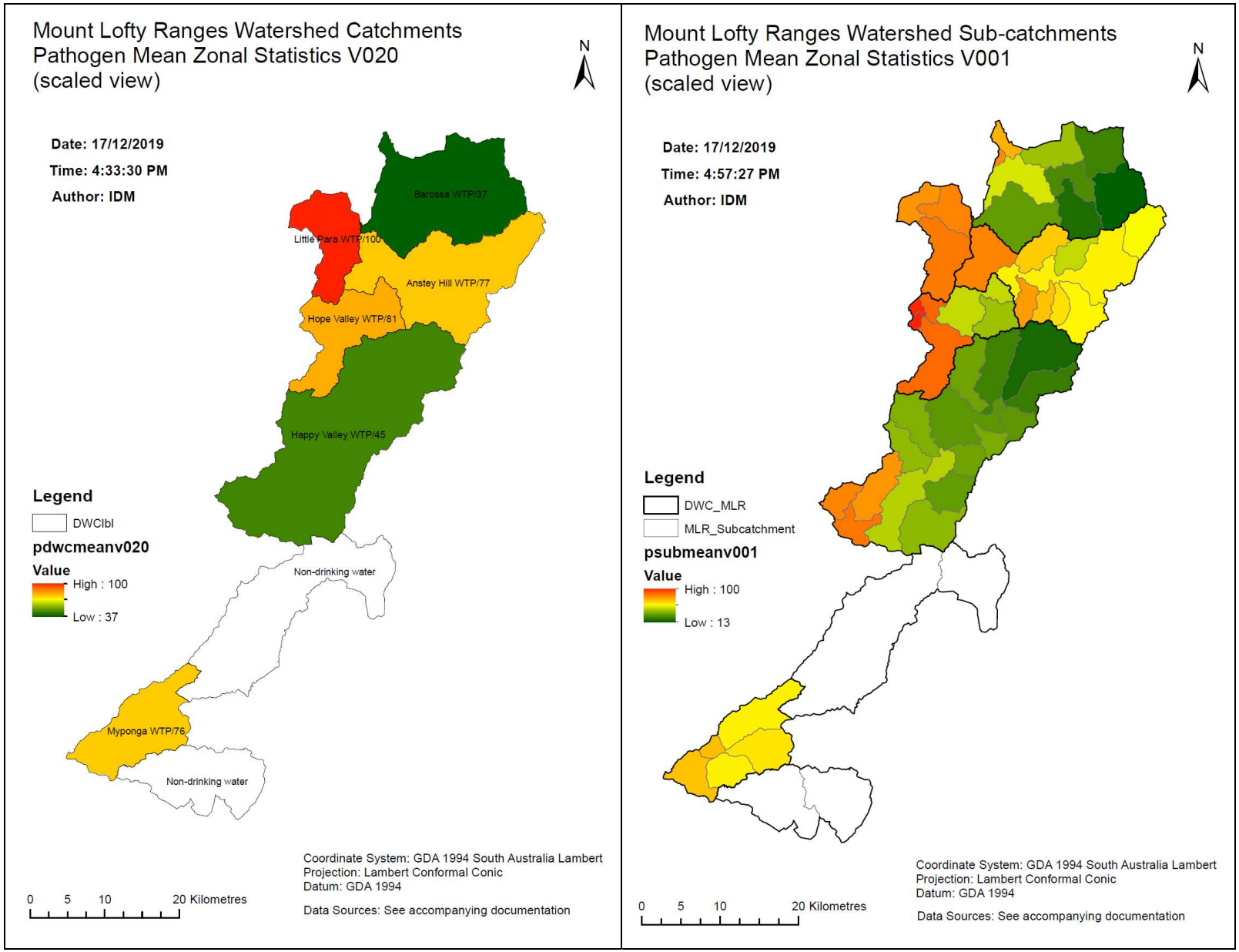


Figure 4.5: Standard reports of Zonal Statistics means for catchments (L) and sub-catchments (R)

The standard reports images in Figure 4.5 are of the Zonal Statistics output for catchments and for sub-catchments. In this case, the option of the statistical MEAN for each (sub-)catchment was chosen as this is the value that was used in the SAW verification strategy (see Chapter 8 below).

4.3 Discussion of Python Model Development

The Python-defined GUI supports the endeavour of enhancing the model environment for the user. Here, the emphasis is on simplifying interactions with the model support systems:

- the user sees the simple, familiar device of a radio button menu to choose between pollutant sources or transport variable data maintenance. Underlying this choice are supporting data definitions (the file name prefix letter referred to above) and generic Python script sequences that can accept the radio button selection and work with the relevant files
- data maintenance has been simplified, with a simple button click required to update shape and raster files, to generate the summary risk surface or to produce the set of standard reports. Formerly, an error-prone copy-and-paste from Excel to ArcMap was required to incorporate the spreadsheet data into an attribute table
- the main risk report records the spreadsheet-defined weights and reference number, plus the automatically updated risk file version number
- the Zonal Statistics output automatically includes the mean for each catchment for use in the validating process.

In order to support this level of abstraction, various support capabilities have been added to the Python script. These include function or class definitions for initialising the GUI, managing the cursor activity status, file name version control, purging intermediate files, displaying spreadsheet contents, sending messages to the GUI window, laying out the reports, statistics generation, shape selection, etc.

In its *sens7* (all risks) form, the model generates negative risk values and these don't reflect any activity occurring in the real world of the MLR. This issue is discussed in detail below but in the interim two further support functions have been developed, normalise and scale. The first of these simply converts the raw risk values into the range 0 to 100. This is currently used in the standard risk report where SAW previously used the terms Low and High. Ultimately, when the issue of negative values is resolved, calculated risks will be scaled against the maximum for the MLR.

5. APPLYING THE PATHOGEN MODEL

5.1 Results – generating new views

Given the ability in the GUI to choose a catchment or sub-catchment (Figure 5.1), further analysis beyond the standard reports can now be made available.

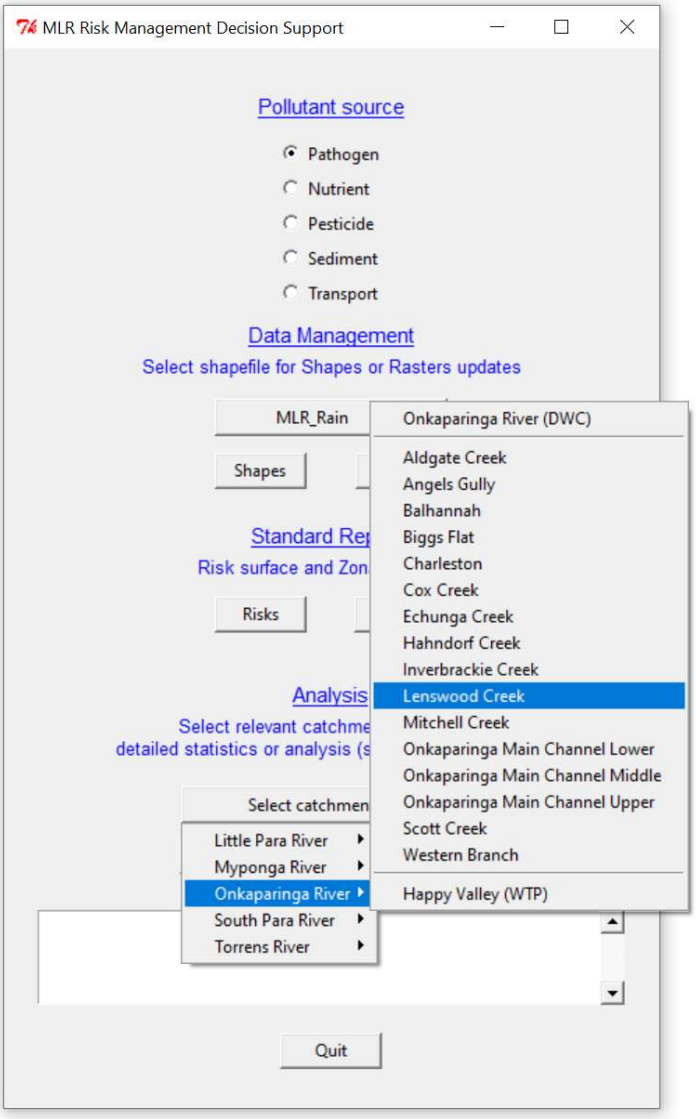


Figure 5.1: Choosing a catchment or sub-catchment for further analysis

Clicking on the Analyse button causes the detailedAnalysis function to be invoked. This produces a report displaying fenced and high-risk regions in relation to streams (Figure 5.2). High risk is defined as a risk level greater than 60% as suggested in Swaffer (2014, p. 54).

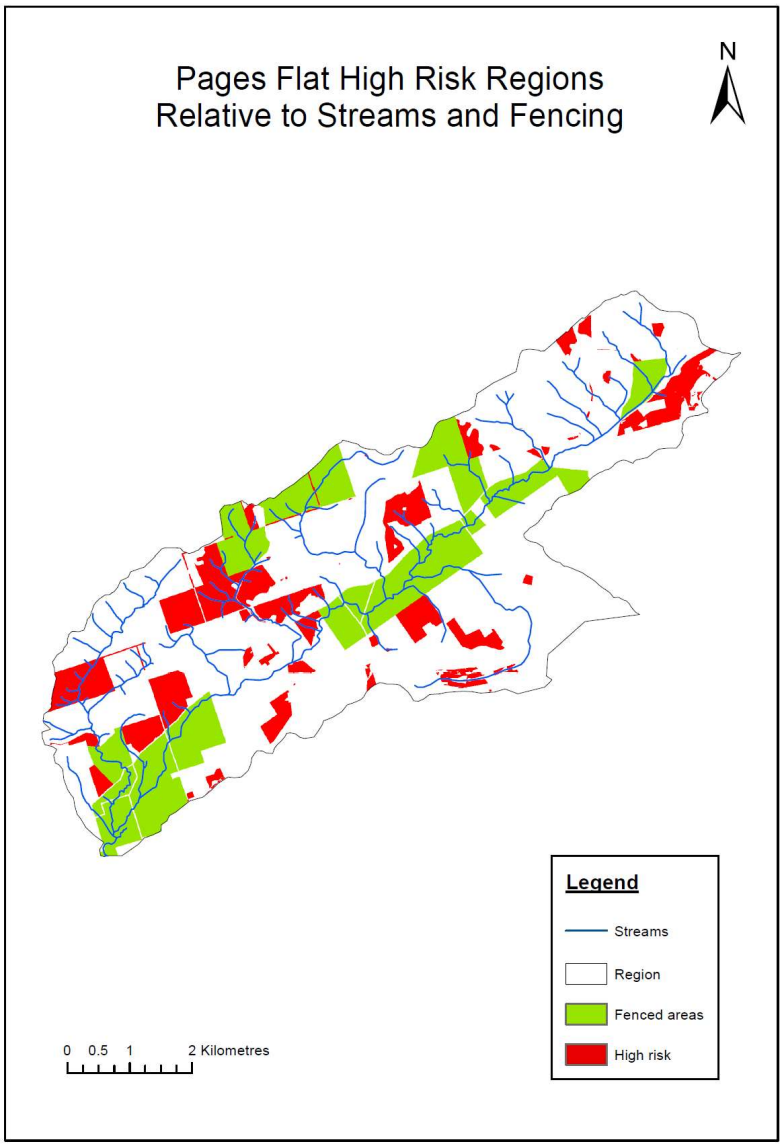


Figure 5.2: Pages Flat high risk relative to fencing

5.2 Discussion of Model Application

The Python scripts provide an infrastructure to support new developments in this project. In later chapters, variations in the model itself are considered. However, even in the current model, new *ad hoc* views become relatively easy to implement. An example is given here that displays potentially high-risk areas in relation to streams and fenced areas. Other possibilities include:

- refining the risk display according to particular variables, e.g., intensive animal production (pintapr) or irrigated pasture (pirrpas)
- estimating the costs and water quality benefits of riparian plantings within a nominated stream buffer zone in the high-risk areas (Connor et al. 2019)
- signalling high-risk regions that don't comply with the Development Act's restriction on development within 60 m of watercourses (Government of South Australia 2019c)
- plotting the distribution of dams in relation to high risk areas to assess the impact of dam failures (Pisaniello 2010).

The detailed output addresses another issue with the original workflow. The feedback loop from a change in one of the source file spreadsheets to the standard risk surface report or regression analysis is long and indistinct. By displaying data in a finer grain as shown in Figure 5.2, the spatial disposition of a particular variable against risk can be readily appreciated by developers and planners. This could guide decisions as to where new physical mitigation work might be considered. The addition of before and after views of a change would assist this understanding.

6. REFINING THE PATHOGEN MODEL

A series of six results is presented here relating to the developments enabled by the Python representation of the model, including:

- dealing with the absence of pollutants
- contribution of each variable to the final risk outcome
- refining the measurement regions from catchments to watersheds
- determining the risks associated with watersheds
- substituting water flow velocities in place of slope
- combining the source and transport variable classes with flow in the form of a product calculation rather than simple addition.

6.1 Results

6.1.1 Pathogen absence

In Figure 4.2 above which illustrated the core loop of risk calculations in the risksUpdate function, total pathogen risk was shown to be the sum of source risks and transport risks. This implies there will always be a pathogen risk due, at least, to the transport values. As argued in 2.3.1, the risk should be zero in the absence of pathogens. Thus, the last line in Figure 4.2 now becomes:

```
sumRaster = Con(sSumRaster > 0, sSumRaster + tSumRaster, 0)
```

Figure 6.1: Returning zero pathogen risk if the source risk is zero

That is, the total risk is only positive if the sum of the source rasters is positive. Otherwise it is zero. This generates the following total MLR risk surface (the previous version is included for comparison) (Figure 6.2):

Two features should be noted in the images in Figure 6.2. First, in the *sens7* output, the NW region of the Little Para system (circled in green) shows relatively high risk. This decreases in the *sens8* output. Second, in contrast, the NE region of the Barossa system (circled in blue) appears to have done the opposite, i.e., the risk has increased in *sens8*. This is an artefact of the way ArcMap's symbology has been applied. When zero and below values are removed, what were formerly low positive values are given higher value colours when the colour scale is applied. These and other observations prompted an investigation, presented below, into the contributions each of the risk variables to the total risk outcome.

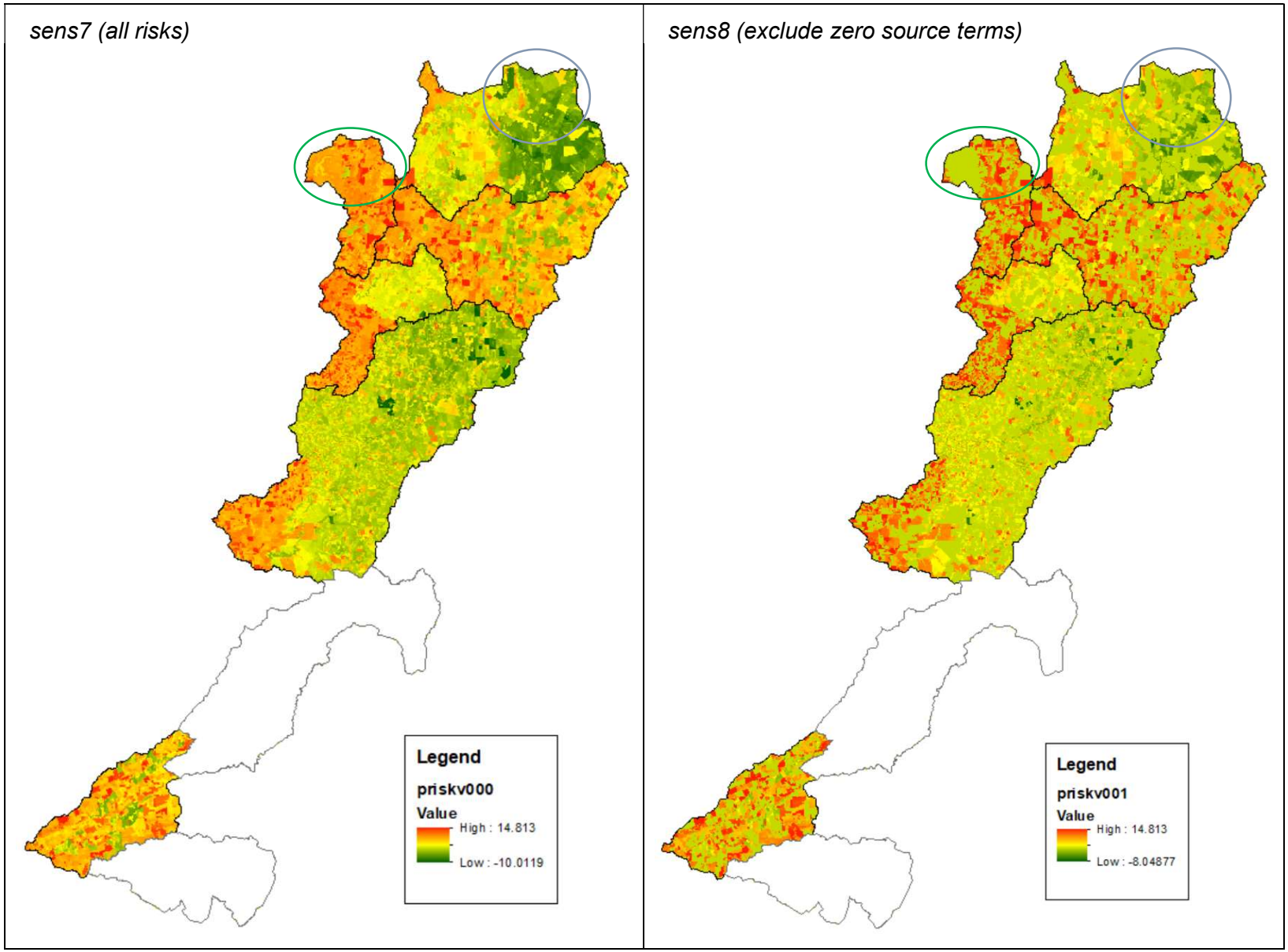


Figure 6.2: Original *sens7* risk surface (L); new *sens8* surface reduced by zero pathogen transport risk (R). The circles are to assist in comparing regions (see below)

6.1.2 Relative contributions of model variables

Appendix F lists the Python script that analysed the risk contributed by each variable to the total. The output from that script is shown below in Figure 6.3:

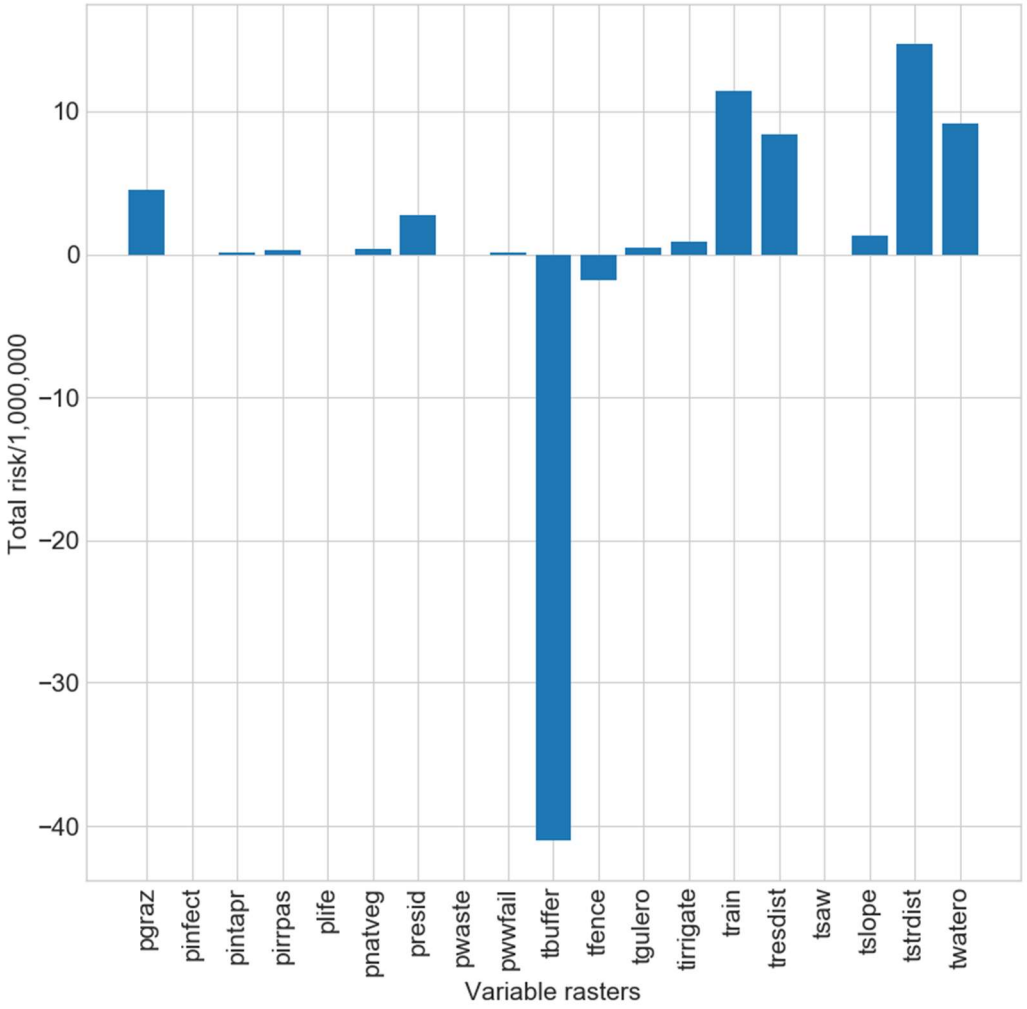


Figure 6.3: Risk contributed by each variable to the total

It is apparent that the tbuffer (upstream storage impedance) variable contributes a substantial negative amount of risk to the total. This reflects both the high initial risk and the weight assigned to this variable in the source and model data. Also, the pathogen source variables appear to have considerably less influence than the transport variables on what is actually a pathogen risk calculation. Note that the pinfect and plife (see Appendix D) terms were set to zero as they were not being studied here.

This variable shares Python script could be readily adapted to address the requirement in Swaffer (2014, p. 54) to report on which source rasters (land use type, transport, etc.) at a given location contributed most to the overall risk.

6.1.3 Watersheds vs catchments

In order to introduce the concept of a flow rate in place of slope in the model, it was necessary to determine the watersheds contributing to the flow past the six catchment monitoring points.

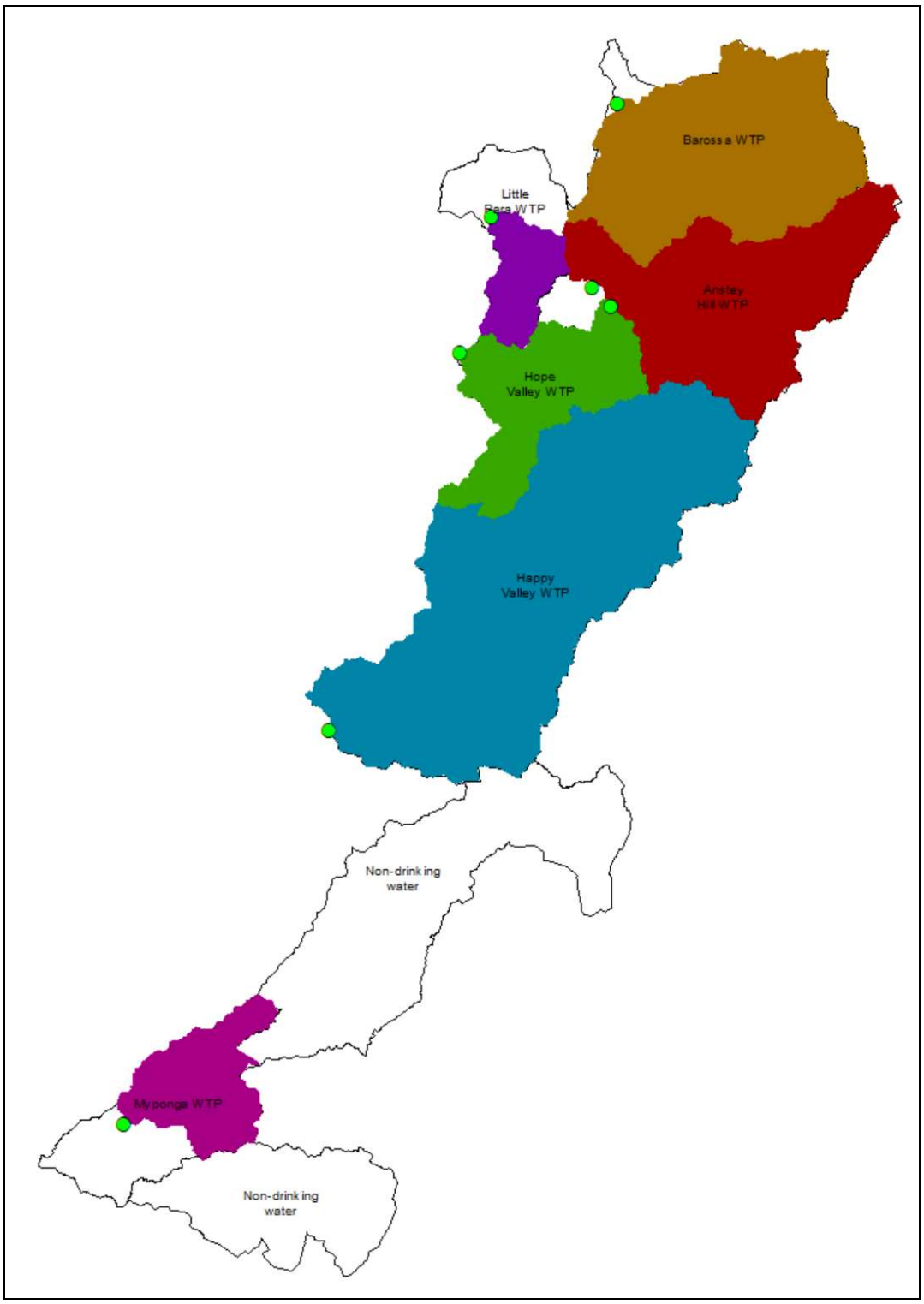


Figure 6.4: Monitoring points (●) and their associated watersheds

Figure 6.4 shows the MLR catchments in outline with the six monitoring points and their associated watersheds. It is apparent that for some systems, the watersheds and catchments do not coincide. This is particularly the case for Myponga in the south and Little Para in the north-west. It should

also be noted that the Anstey Hill catchment (in red) shows two monitoring points. These were each connected to separate watersheds but their analysis was combined as only one water quality figure was quoted in Swaffer (2014). Only one of this pair was employed in the nutrients study (see below).

6.1.4 Watershed risks

Given the finding above, it was appropriate to investigate the risks associated with the watersheds. This output was designated *sens9* and is compared with the previous whole-of-catchment output (*sens8*) in Figure 6.5. Where the watersheds overlay the catchments, the images are largely the same. However, there are some variations in symbology as the range of values over which it is applied may vary. The circled areas demonstrate this where the colour for the low valued green regions is less intense in the *sens9* version.

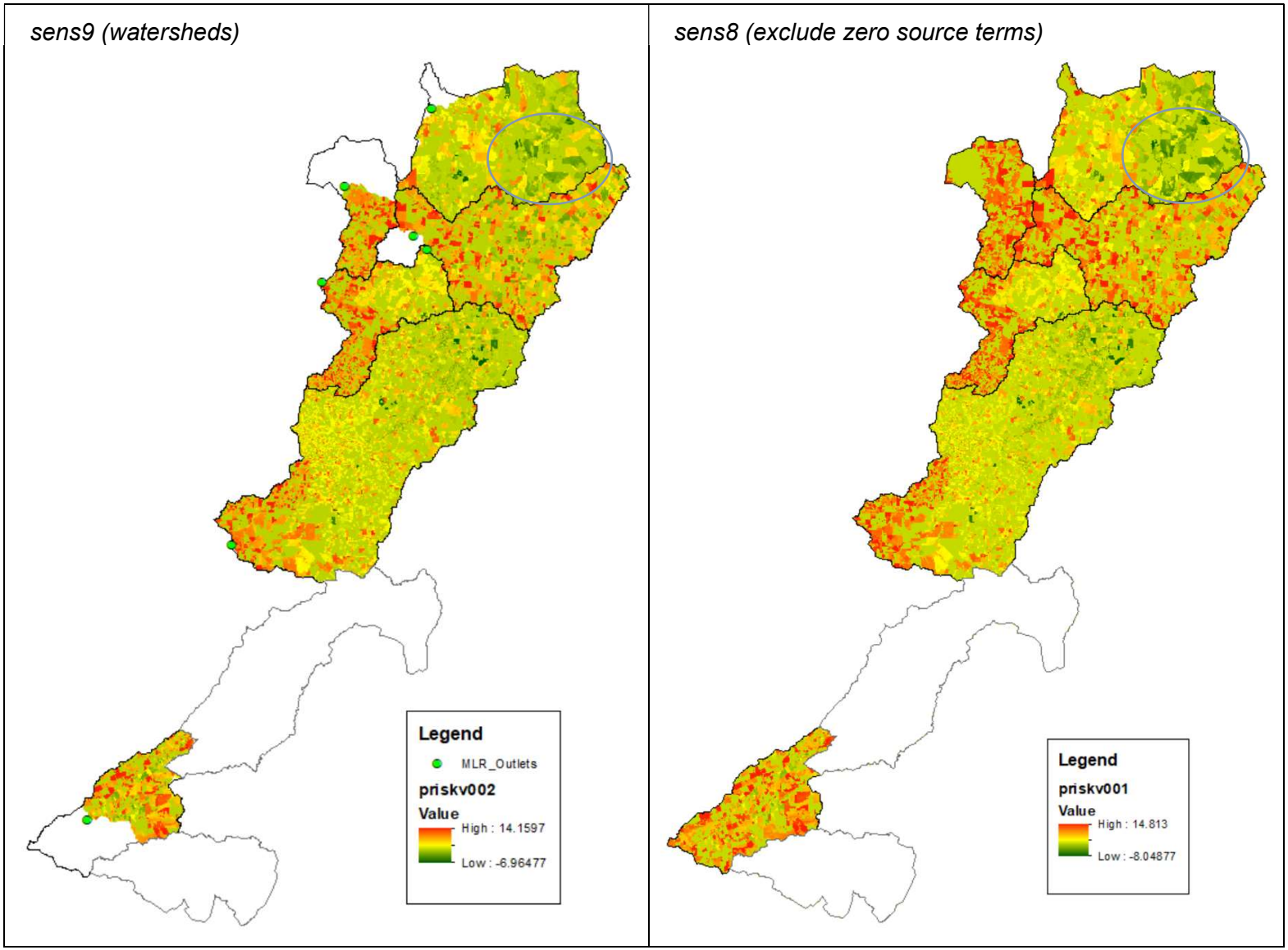


Figure 6.5: *sens9* watershed (L) and *sens8* whole-of-catchment (R) risks. Circled areas are for comparison

6.1.5 Surface water flow rates in place of slope

Another variant of the model incorporates the previous changes with an even further reduced tbuffer risk value to counter the generation of negative risk numbers. In addition, this variant substitutes a surface flow rate (tflows) in place of slope (tslope), resulting in the "*sens10*" version of the model (Figure 6.6). Note that this version still adds all the risk terms.

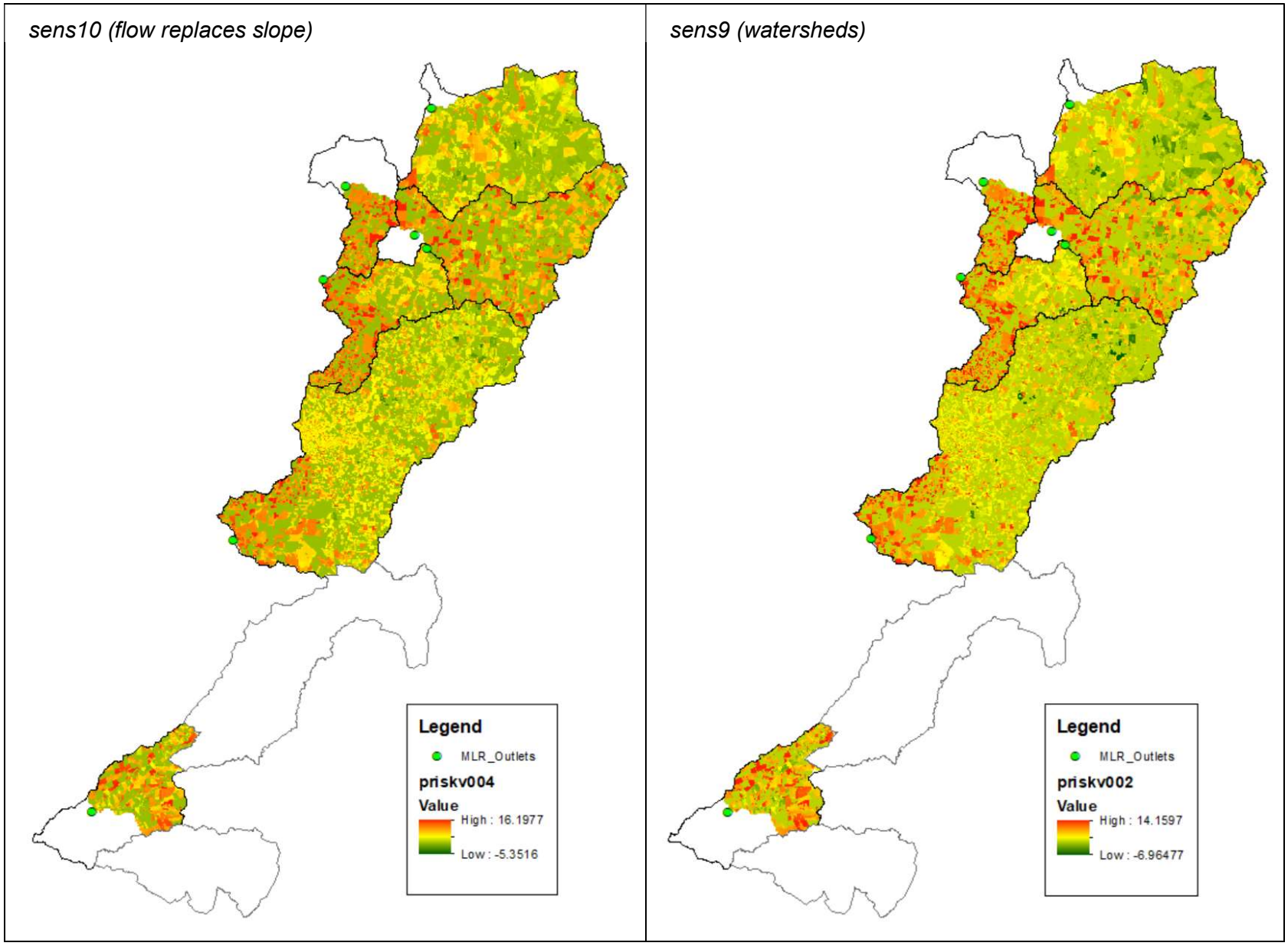


Figure 6.6: Comparison of *sens10*, flow rate (L) and *sens9*, watersheds (R) versions of the model

The *sens10* version of the model shows a wider range of risk values compared with the previous version.

In a variation of this approach, the weight applied to tbuffer and tfence was traded off to other parameters such as train, pintapr (intensive animal production) and pirrpas (irrigated pasture) in order to reduce the negative impact of the first two (Figure 6.7).

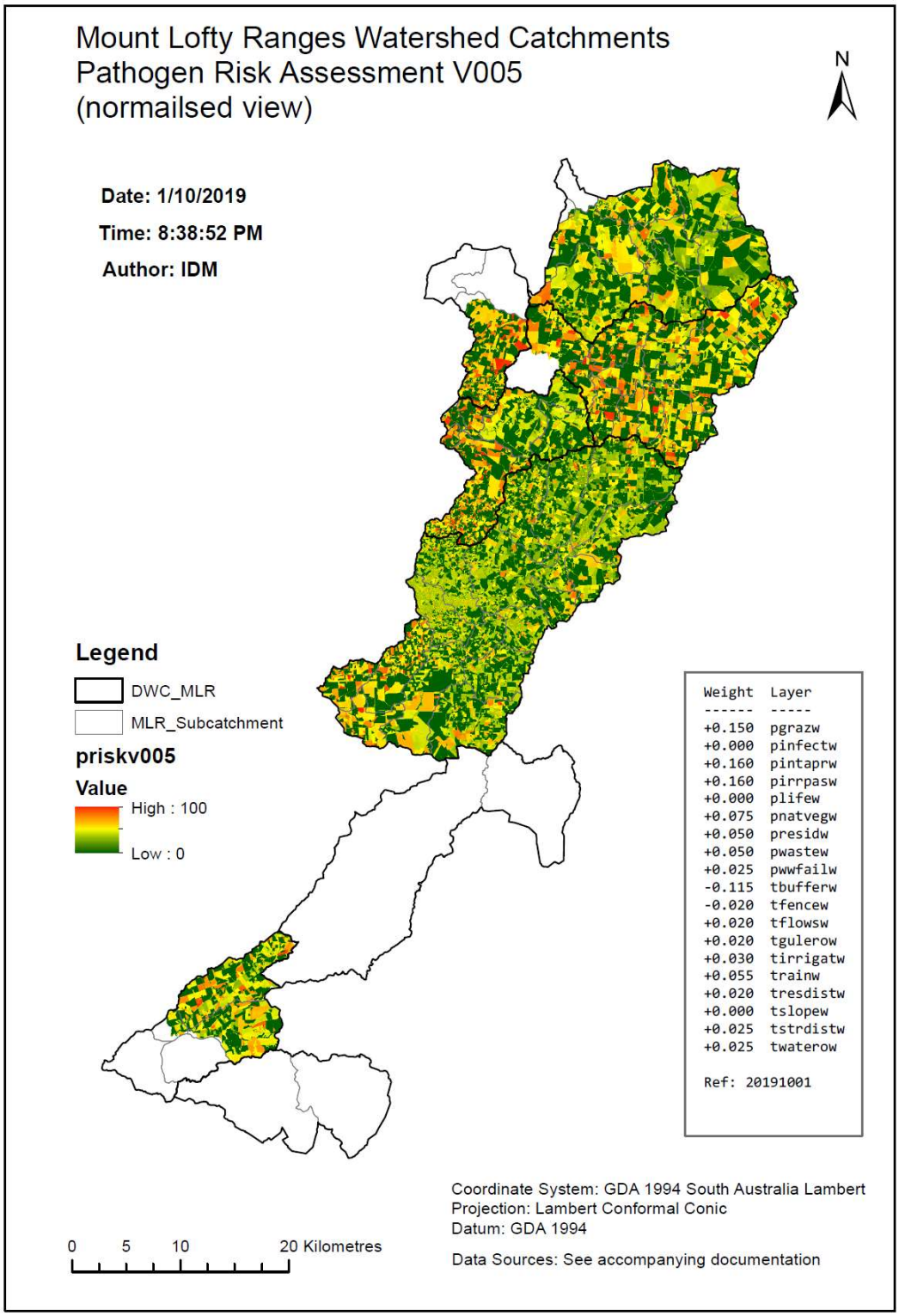


Figure 6.7: The final expression (*sens10*) of the additive version of the pathogens model

It should be noted that Figure 6.7 represents the final version of the model that adds (or subtracts) the value of each weighted variable. In summary, it accommodates the lack of pathogen sources, the distinction between watersheds and catchments, and it substitutes surface water flow risk in place of a slope risk. It appears in this view that the extent of high-risk regions has been reduced compared with previous versions. There is a corresponding increase in low-risk areas.

6.1.6 Final version with flow rate as a multiplier

It has been assumed above that the absence of pathogens (the source) implies a lack of risk of pathogens. By analogy here, it is intended that if there is no risk of water flow, there will be no transport of pathogens and hence little or no risk of contamination at the cell being evaluated. Hence, in this final implementation, rather than simply adding the flow rate component to the model, it is used as a multiplier on the transport and source groups. This approach produced the following output (Figure 6.8), which will be referred to as the "*sens11*" flow rate modifier version:

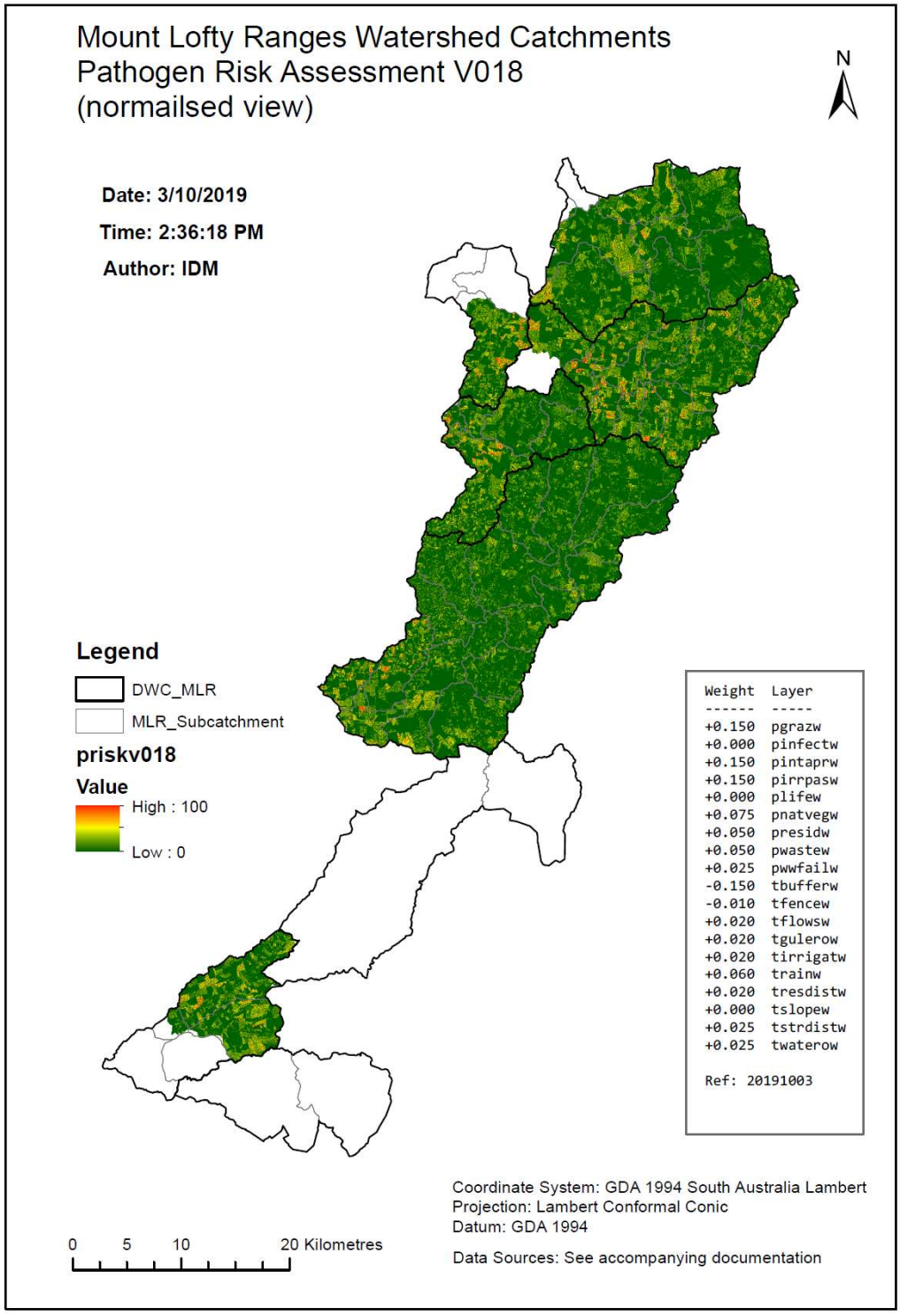


Figure 6.8: Multiplicative pathogens model (source * transport * flow)

For comparison, this last version is shown below with the original (*sens7*). Note that the unscaled risk values are shown. The negative minimum for *sens11* has almost been eliminated.

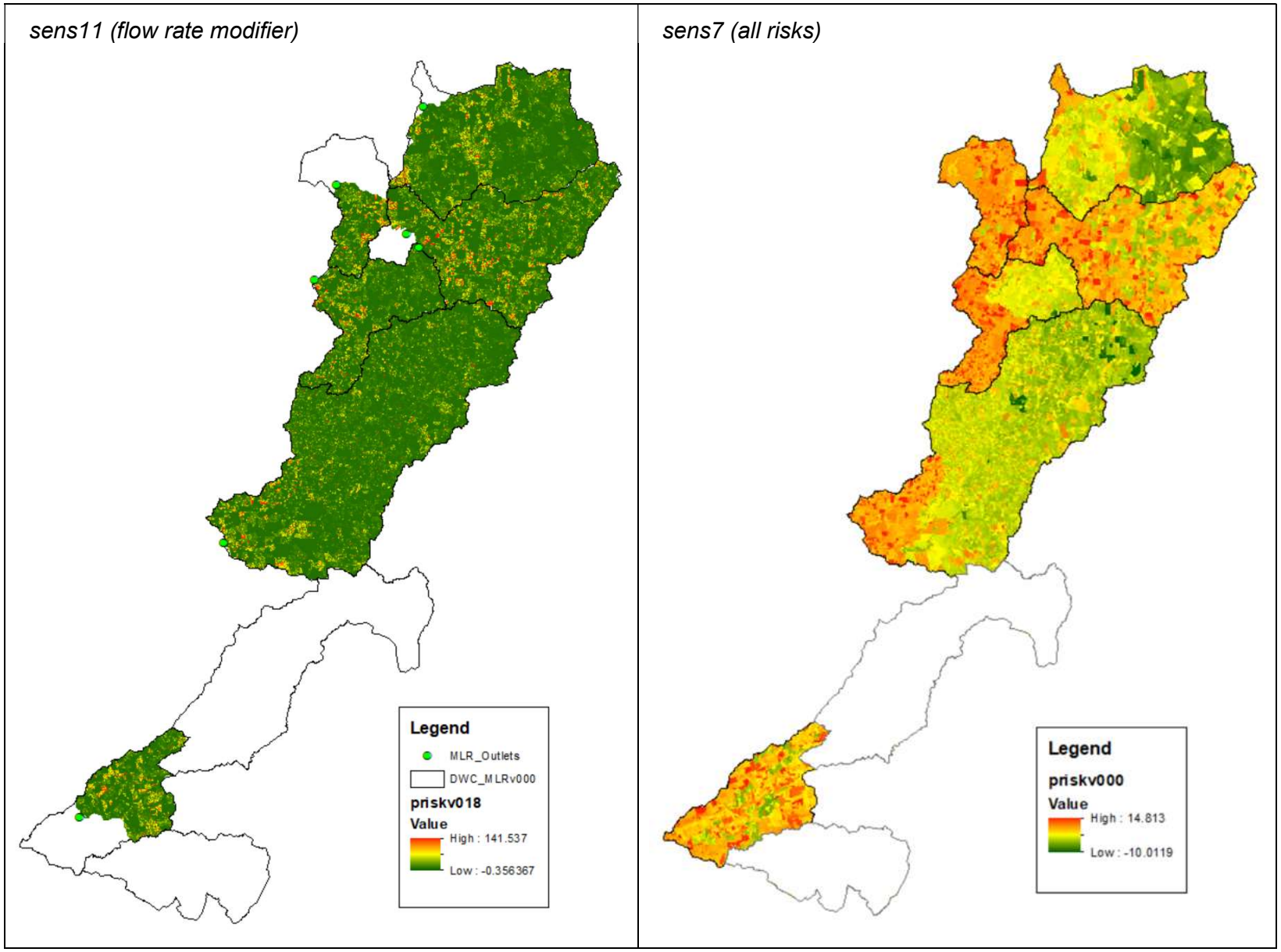


Figure 6.9: Final (*sens11*) and original (*sens7*) pathogens model outputs

In Chapter 8 below, comparisons will be made between the scaled mean risk values for *sens11* watersheds and *sens6* catchments in order to demonstrate the changes brought about by the new model. The Zonal Statistics MEAN was determined for the watersheds and catchments as illustrated in Figure 4.5 above. The maximum mean value in each case was assigned the scaled value of 100 and the values for the other regions was scaled against this. The effect is represented visually below in Figure 6.10 but the assessment of the numerical details is detailed further in Chapter 8. Note that, as in many cases throughout this report, the only imagery available from the original SAW report is in the form of a non-revisable figure so some detail has been lost.

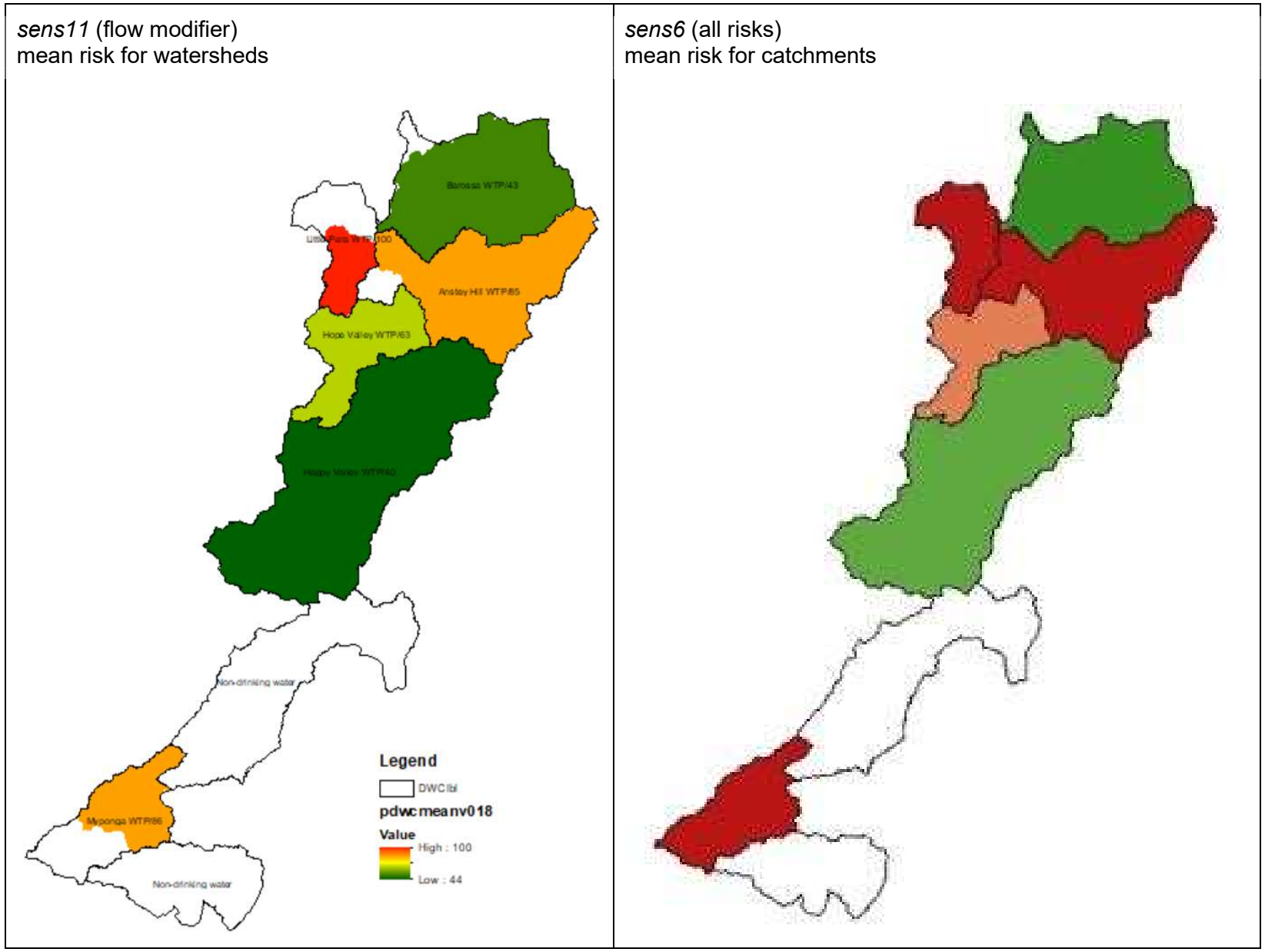


Figure 6.10: Comparison of pathogens mean risk values for *sens11* watersheds (L) and *sens6* catchments (R) (Swaffer 2014, p. 38)

Table 6.1: Risk statistics for the final (*sens11*) and first (*sens7*) pathogens models

	sens11	***sens7***
Minimum	-0.4	-10.0
Mean	0.6	1.8
Maximum	141.5	14.8
Standard Deviation	2.2	3.1

Although the maximum value for the *sens11* output is ten times the one for *sens7* (Figure 6.9), its mean is closer to zero (Table 6.1). This is unsurprising given that flow rate risk is now one of the drivers of the model as defined here. As Figure 6.11 illustrates clearly, low to medium (green-yellow) risk values predominate in this layer. Note, however, that its weighting in the model is quite low – the same as was previously used for slope (0.020).

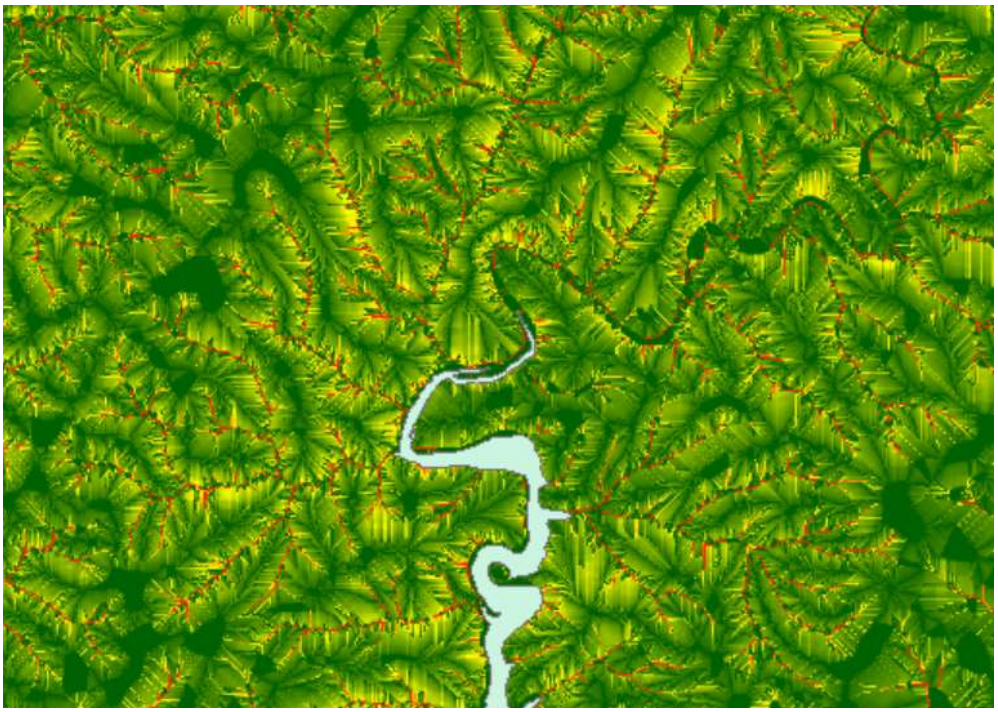


Figure 6.11: 1:30,000 view of the flow rate raster at the northern tip of Mt Bold Reservoir

Table 6.2 summarises the features that have been added or modified at each stage in the refinement of the model.

Table 6.2: Summary of modifications for *sens7 – sens11*

Model variant	Add source and transport groups	Ignore transport if source is zero	Account for Watersheds	Substitute flow rate for slope	Multiply source, transport and flow terms
sens6					
sens7	✓				
sens8	✓	✓			
sens9	✓	✓	✓		
sens10	✓	✓	✓	✓	
sens11	✓	✓	✓	✓	✓

The final configurations of the pathogen and nutrient models can be found in Appendix H.

6.2 Discussion of Model Refinements

In a model defined at a resolution of 5 x 5 m cells, there are more than 236 million cells defining the MLR. For each of these, an evaluation is performed of 16 raster images representing the contamination risk associated with each of the model variables. It is conceivable that for many of these cells, the result of the evaluation would be zero. This was not the case for the model at the start of the project. There was almost no circumstance where the addition of 14 variables' risks and the subtraction of two could result in zero. In fact, there were situations where substantial negative values appeared. This doesn't seem to represent any activity occurring in the real environment.

In order for a model to represent the contamination by pathogens of surface water arriving at reservoir monitoring stations, two necessary conditions must obtain:

- there is a source of pathogens, and
- surface water must be able to flow in the direction of the reservoir in order to carry the pathogens there.

There are many situations across the MLR where at least one of these conditions is not met. For example, a cell covers a road pavement which cannot be the source of pathogens. Alternatively, a

cell may represent water pooled in a dam where oocytes may be held long enough to lose their viability.

This project set out to account for these observations in new ways, providing a more nuanced view of the impact of the variables and their interactions. Instead of simply adding terms, the model now makes pathogen presence and surface water flow the drivers of the model. This is achieved by multiplying the source group of terms by the transport group and by the water flow risk. Any of these terms can now force the entire calculation for a cell to zero, resulting in the overall lower risk profile of Figure 6.8 above.

The process of developing the final model from the original one, led to some difficulties having to be overcome. Two strategies were applied to reduce the negative risk values originally observed. The initial risk values for buffer and the weights for fence were both reduced. The weight for rainfall was increased (see the table in Figure 6.8) to maintain the overall weight budget at 1.0. This choice was also in keeping with making water movement a key influence on the outcomes. However, it is acknowledged that these modifications need to be reviewed by the catchment management professionals at SAW as they have the deeper understanding of the data.

The determination of water flow rates across the MLR required the derivation from the DEM of watersheds above each of the catchments water quality monitoring sites. In an intermediate step in this process, the flow accumulation network was also determined. For the Myponga catchment, a disconnect appeared between the flow accumulation view and the stream network view which was available separately. In the event, new DEM and flow accumulation rasters had to be calculated. This has only been completed for the Myponga catchment as the discrepancy was not observed in the northern catchments. Nevertheless, this should be followed up with the agency that manages the DEM.

It became apparent during this process that the watersheds above monitoring points can be quite different from entire catchments. This led to a new approach where risk was evaluated for watersheds only. Others may be in a position to argue that conclusions drawn from watershed studies may be applied more broadly to their hosting catchments and that, therefore, the original approach should stand. However, a deeper statistical analysis of the layers may suggest otherwise. There are some opportunities available to do this analysis, even from the existing limited data set:

- compare the watersheds and catchments for Happy Valley and Hope Valley – they should be substantially the same
- compare the watersheds and catchments for Little Para and Myponga where the two entities differ substantially

- compare the two watersheds in the Anstey Hill catchment – this will provide support, or otherwise, for the argument that a watershed can realistically represent a catchment.

The Python script to evaluate the contribution of each variable to the overall risk was developed in response to a specific need at the time. However, it is anticipated that it will become a useful tool in its own right to monitor the effect of changes to any variable, or combinations of them, not just the ones generating a negative outcome. The routine can be folded into the main Python application, under the Compare button, once a bug fix becomes available to overcome a problem with file access from the Python system.

7. MODELLING NUTRIENT RISK

The design of the Python application facilitates its extension to models of pollutants other than pathogens. This study applied the approach to nutrients, specifically the release of Phosphorus into the MLR catchments. The results are discussed below.

7.1 Results

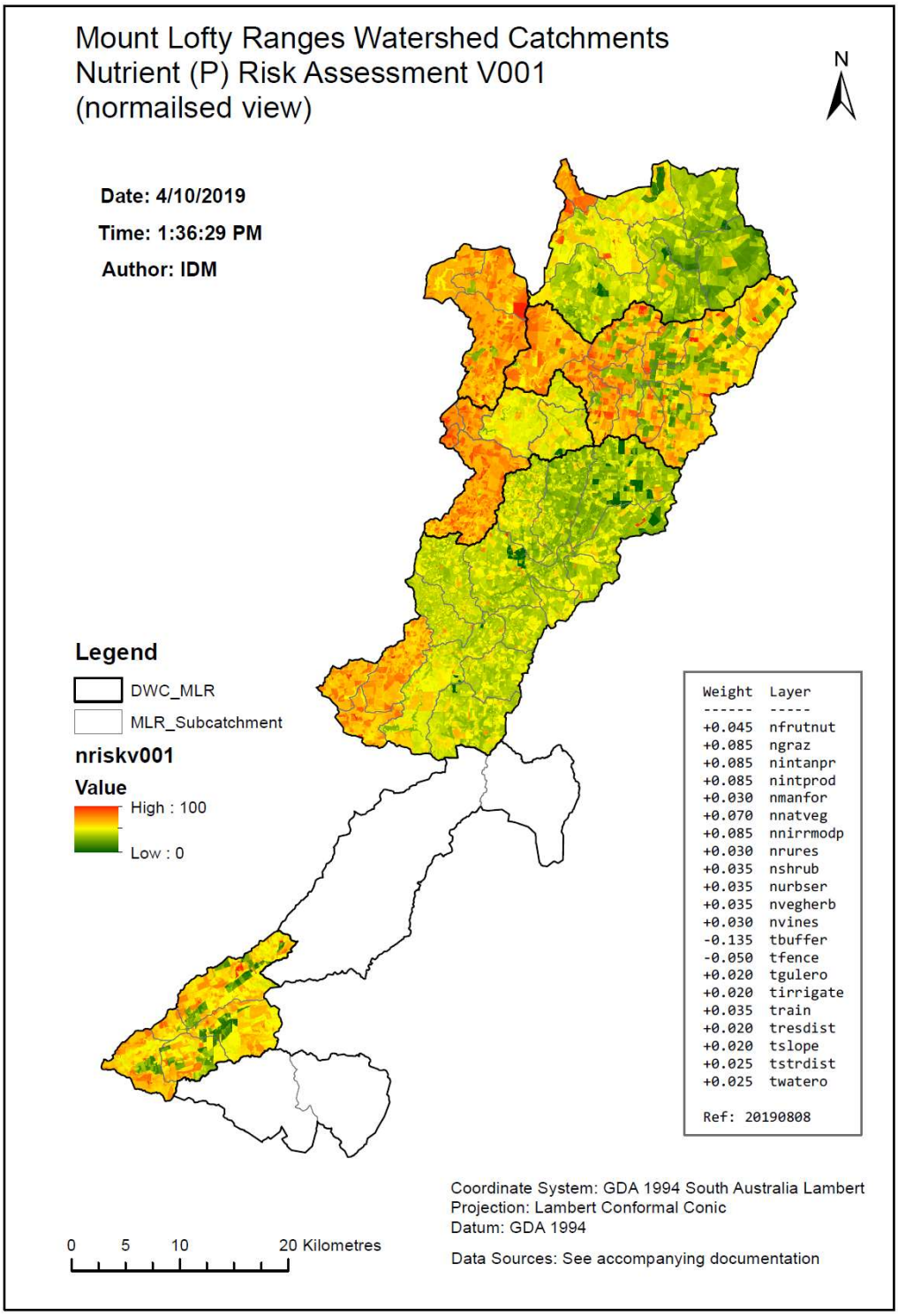


Figure 7.1: Output of the original *sens7* model as applied to nutrients

Two results are presented here to illustrate the application of the modelling strategies developed for pathogens to nutrients (specifically Phosphorus). The first (Figure 7.1 above) shows the output of the *sens7* additive model. As there are more nutrient terms in the model, the weights budget has been distributed proportionately. The details of the variables and their weights can be seen in Appendix I and Appendix H respectively. The underlying risk estimates for each land use type were provided by SAW in the form of a separate MLR_Landusen.xlsx spreadsheet. This was read into the Python application in place of the corresponding pathogens' spreadsheet used previously.

The output in Figure 7.1 is similar to the analogous one for pathogens (Figure 4.4 above). This is to be expected as pathogens and nutrients are afforded comparable treatment in the model (Figure 7.2 below), i.e., each weighted term is added to (or subtracted from) the final risk value.

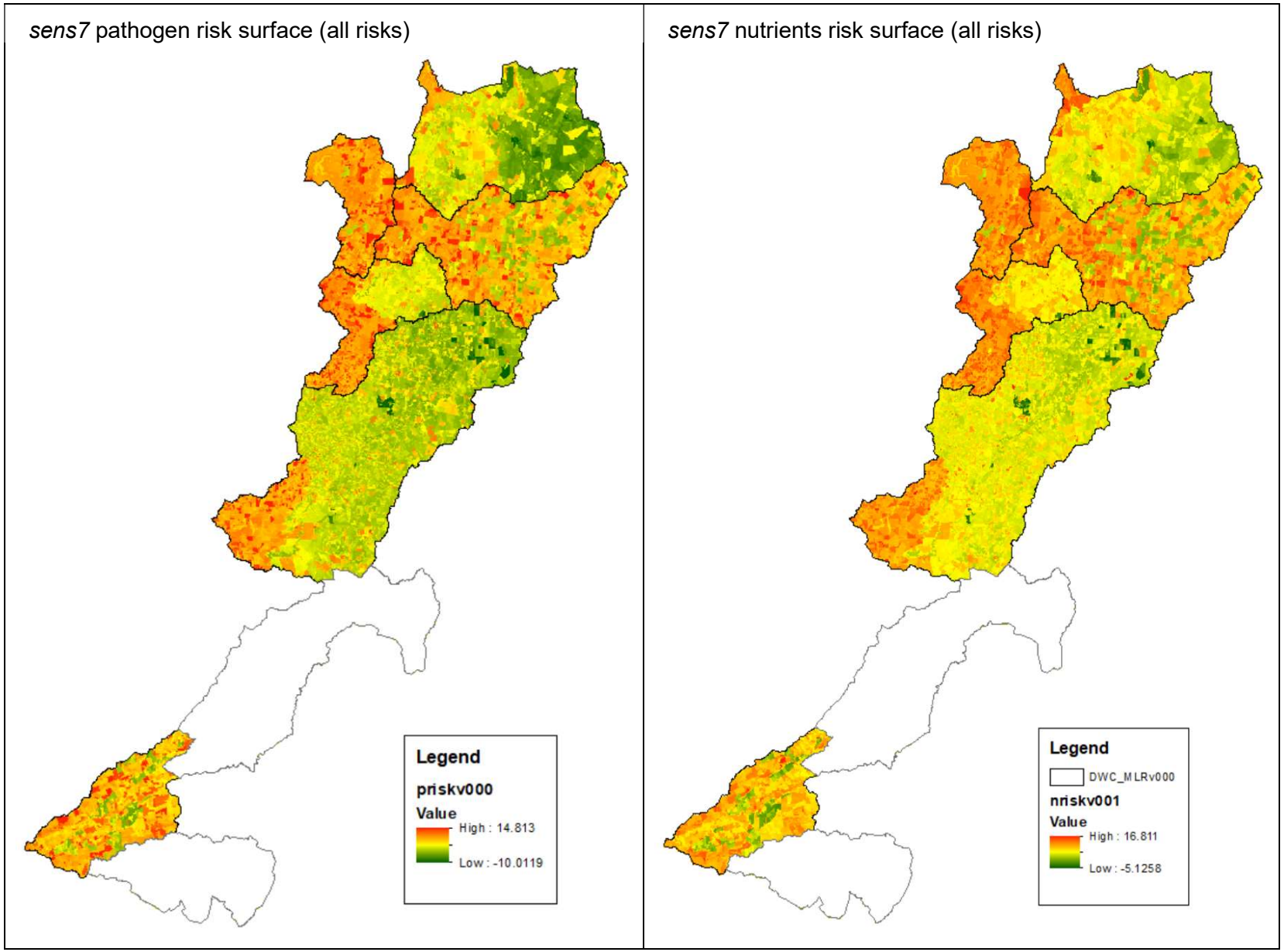


Figure 7.2: Comparison of the pathogen (L) and nutrient (R) outputs for *sens7*

The second outcome of the nutrient study (Figure 7.3 below) shows the impact of the final model development (sens11). This put the focus on monitoring station watersheds with nutrient presence and water flows as the major influences. Again, there is the appearance of generally reduced risk across the watersheds.

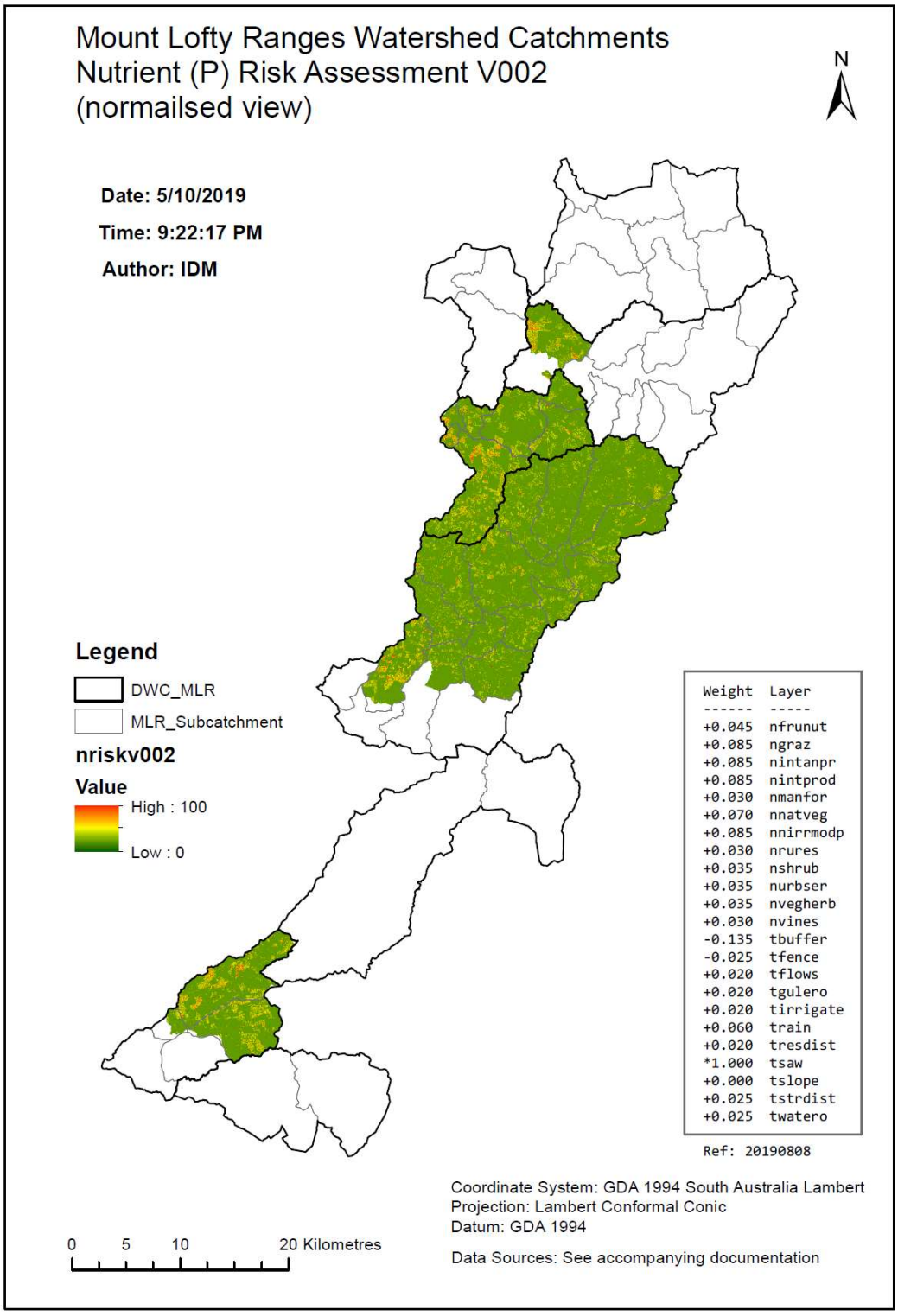


Figure 7.3: The *sens11* output for nutrients

The final output of the pathogens and nutrient studies is shown below (Figure 7.4 below). This comparison clearly shows the differences in watersheds determined for the pathogen and nutrient views. This is due to some monitoring points being in different positions, resulting in different watersheds being calculated. The range of risk values also differs considerably between the two although negative values have been almost eliminated in both. This was achieved largely by trading off fencing against rainfall, with the other transport variables held constant. The final values for the model variables and a description of them can be found in Appendix H and Appendix I respectively.

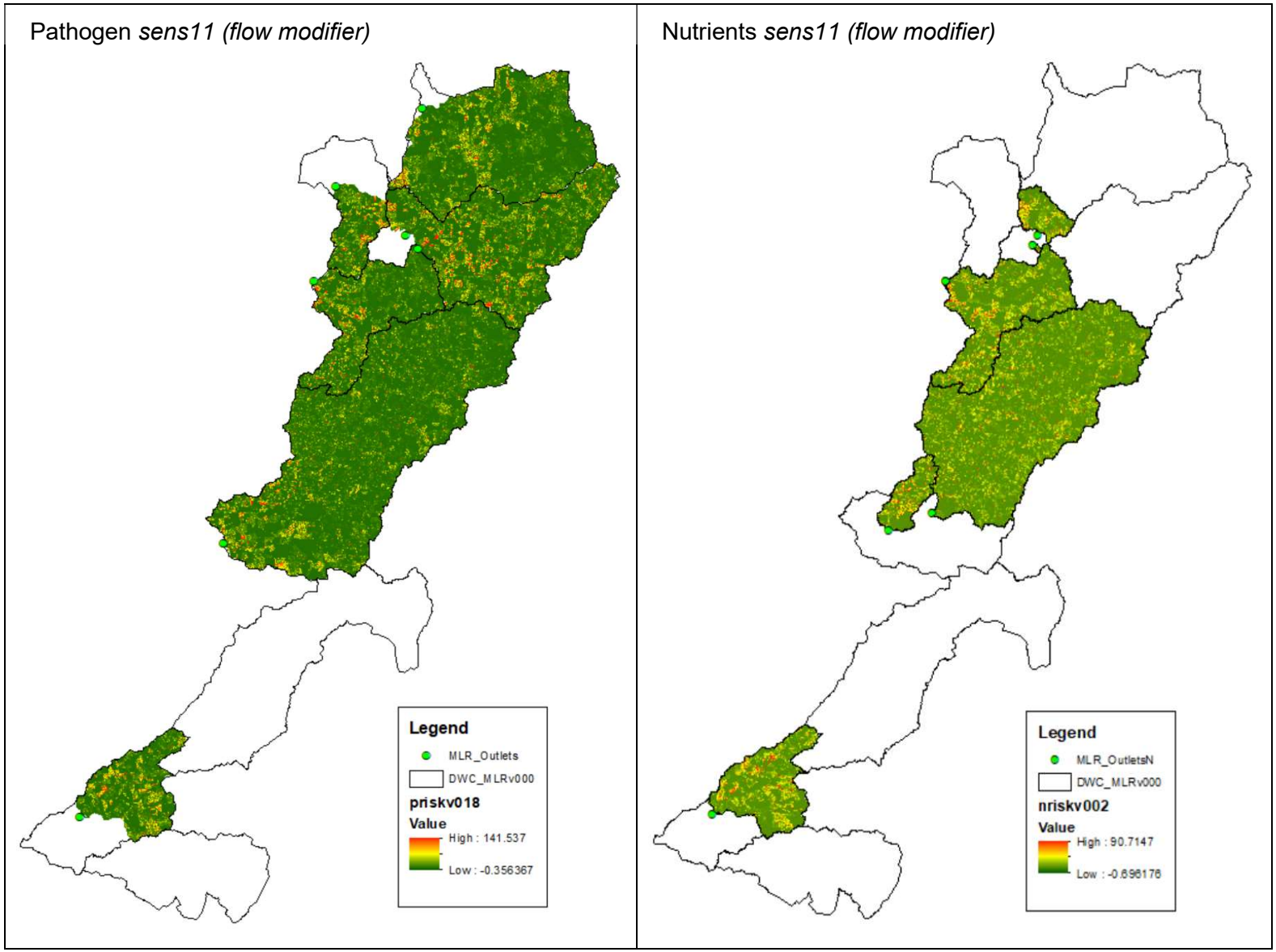


Figure 7.4: Comparison of the sens11 output for pathogens (L) and nutrients (R), illustrating the difference in watersheds associated with the different monitoring stations

7.2 Discussion of the Nutrients Model

This series of experiments investigated how well the techniques developed in the pathogens study could be applied to other pollutants such as nutrients, pesticides or sediments. Surprisingly little effort was required for this to succeed. First, a number of Python support functions developed for pathogens could be applied directly to different pollutants. These have to do with user presentation, data maintenance, model execution, reporting, etc. Second, initial attention to data design and naming conventions meant that the same code could be applied to each pollutant. Differences were external to the Python script in files holding data and the definitions relevant to each pollutant. Once a pollutant choice was received from the user, the resulting parameter was used to guide data file choice, report layout and so on without pollutant-specific code sections being required.

A number of limitations became apparent during the nutrients study. The strategy has generally worked well. However, the approach is predicated on the assumption that only Phosphorus would be modelled. No attempt has been made to represent the more complex behaviours of Nitrogen here. For the purposes of this study, it has been assumed that Phosphorus behaves similarly to pathogens. Being more reactive than Nitrogen, Phosphorus is more likely to be carried off with other contaminants in surface water, as is *Cryptosporidium*. Other behaviours of Phosphorus such as draining into the soil or leaching from non-biological sources remain to be investigated.

One difficulty highlighted by the application of the Python model to nutrients relates to the way contamination monitoring is deployed. The SAW approach for pathogens allocated one monitoring station (or two combined for Anstey Hill) per catchment and assumed that whatever was observed at the station reflected the behaviour of the whole catchment. This may be the case but it would be worthwhile confirming. The need to work with monitoring station watersheds revealed the possibility of variations across catchments. As the results from more monitoring stations become available, watersheds of finer resolution will be able to be defined. This may reveal variations across catchments that are not currently visible, at least in the current system.

Another difficulty raised by the Python activity, concerns the representation of the underlying model. Through the different stages of development here, as a new model was established, the previous one was commented out in the risksUpdate function (Appendix E). It may not be difficult to adapt the behaviour of sediments to the new flow parameter. However, the modelling of Nitrogen's behaviour in different soil types may require an alternative technique for expressing the model in Python and in model selection following the user's pollutant choice. From a programming point of view, this is the core of the transferability problem alluded to by Oliver, DM et al. (2016)

and Porter et al. (2017). It could be addressed by replacing the existing model definitions in risksUpdate with calls to separate functions that model each pollutant's behaviour.

It may be possible in future to use points shapefiles with latitude/longitude positions of pour points (or watershed outlets) to automate the manual raster pour point placement used in this project. This would provide a single process to generate the flow velocity layer. This process was mimicked by the two automated and one manual process used here. For the six points used in each of the pathogen and nutrient experiments, the latitude/longitude data provided did not align with the flow accumulation derived from the DEM. This is a common problem when establishing pour points and is fixed with a standard procedure. In this study, alignment is required as the flow accumulation is necessary for the calculation of flow rates. It is possible, therefore, that the full extent of each watershed would not be established with a completely automated approach.

There were other steps towards the end of the project that were carried out manually in ArcMap simply because there was not sufficient time to complete the Python development. There is scope for considerable enhancement of the Python system to refine how models are defined and to automate standard data maintenance, model execution and reporting.

Nevertheless, what has been achieved thus far significantly improves the turnaround times for data updates and the record-keeping required to support multiple sensitivity studies. As measured by the SAW regression approach there also appears to be some improvement in the model's predictive behaviour. This is discussed further in Chapter 8.

8. MODEL VALIDATION

The previous SAW work only addressed the issue of *Cryptosporidium* contamination of water. Consequently, the focus of the validation work here is also on the pathogen contamination so that some comparisons with the prior work can be made. The same approach has also been applied to the nutrients study.

8.1 Results

SAW applied an infectivity overlay (Eq. 2.1) to the final *sens6* output to reflect the fact that only a proportion of oocysts would be infective. The infectivity measurement or estimate used in the overlay for each catchment is listed in Figure 8.1 (L). Figure 8.1 below compares the SAW *sens6* infectivity view with that generated by Raster Calculator in ArcMap for the *sens11* flow modified model.

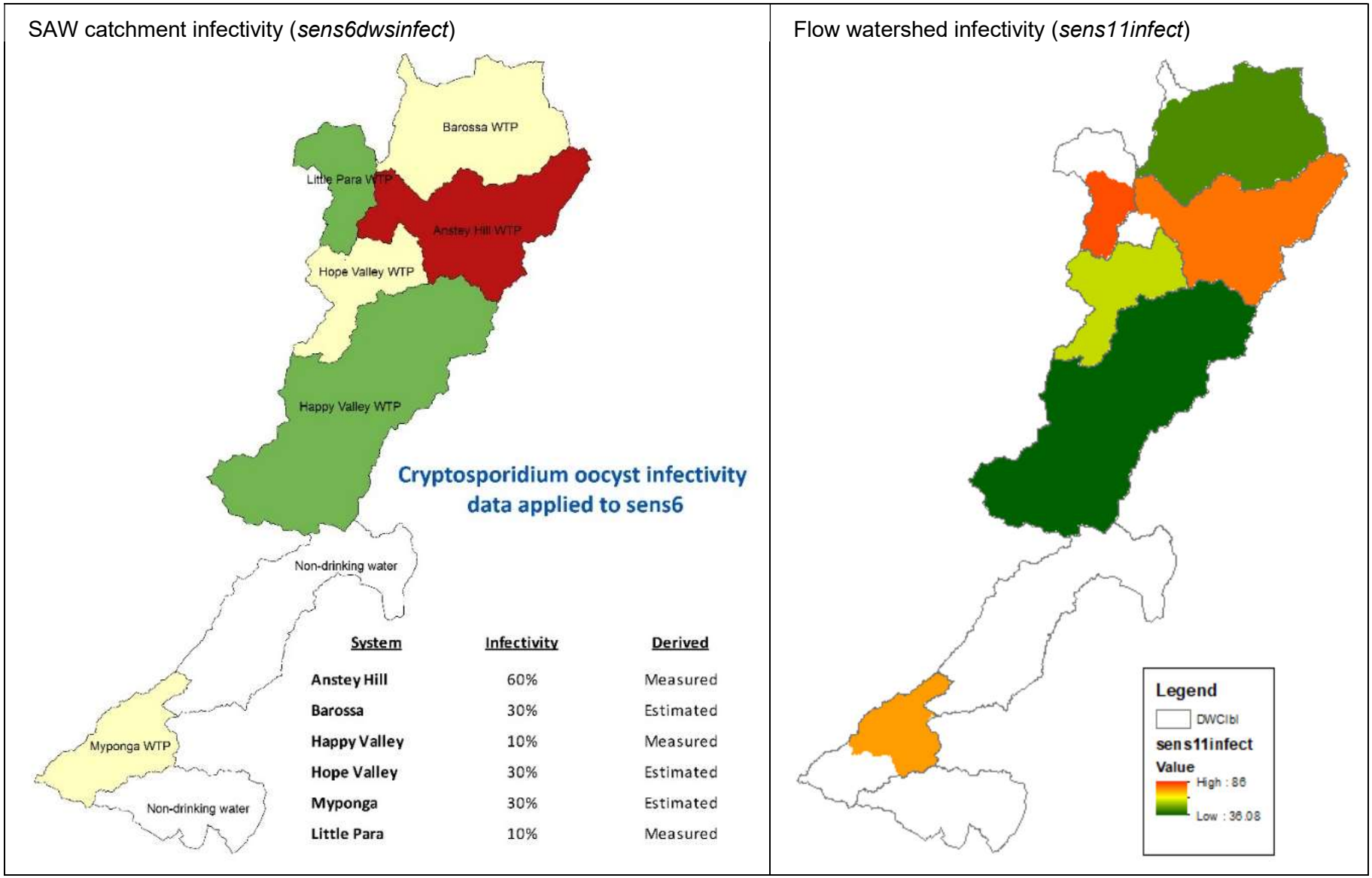


Figure 8.1: Infectivity overlay output for (L) SAW's *sens6* (all risks) (Swaffer 2014, p. 40) and (R) *sens11* (flow modifier)

Differences are clearly visible in the infectivity overlay outputs produced by the two versions in of the model, catchments and watersheds. The differences can be appreciated both visually (Figure 8.1) and numerically (Table 8.1) where the infectivity of the *sens6* version displays a wider range of values.

The result of applying the same infectivity overlay to the catchment (*sens6*) and watershed (*sens11*) scaled risk means is shown in Table 8.1. Note that it has been necessary to adapt to some labelling conventions used in the original SAW report. The values in the *sens6* and *sens11* infectivity columns in Table 8.1 are plotted in Figure 8.2 and Figure 8.3 as "Scaled CRA outputs" on the Y-axis. These are the infectivity overlay values described above. In Table 8.1, they are derived from the adjacent columns of scaled catchment/watershed risk means by applying Eq. 2.1. These scaled means were calculated as described in Section 6.1.6 above.

Table 8.1: *Cryptosporidium* scaled end-of-catchment water quality, catchment/watershed scaled risk means and infectivity derived from the *sens6* (all risks) catchments and *sens11* (flow modifier) watersheds models

WTP System	Scaled water quality data	*sens6* scaled catchment risk means	*sens6* infectivity (Scaled CRA outputs)	*sens11* scaled watershed risk means	*sens11* infectivity (Scaled CRA outputs)
Anstey Hill	100.0	91	84	85	78
Barossa	52.4	8	7	51	44
Happy Valley	62.3	44	36	44	36
Hope Valley	68.5	79	68	65	56
Little Para	85.1	100	82	100	82
Myponga	72.2	100	86	85	73

SAW accumulated end-of-catchment water quality data as a presumptive count of oocysts per 10L (determined by an infectivity assay). These were expressed as log10 values. The maximum of these values, for Anstey Hill, was assigned the scaled value of 100.0. The remaining water quality values were scaled against Anstey Hill's (Swaffer 2014, pp. 38-9). These values are displayed in Table 8.1, Figure 8.2 and Figure 8.3 as "Scaled water quality data" on the X-axis.

The SAW regression strategy applied a least squares regression calculation to the scaled *sens6* Zonal Statistics means and to the infectivity version. The raw risk surface for *sens6* produced an R^2 value of 0.60 and the corresponding value for *sens6infect* was 0.65. Assuming a linear model, the least squares regression analysis of the infectivity numbers revealed (Figure 8.2):

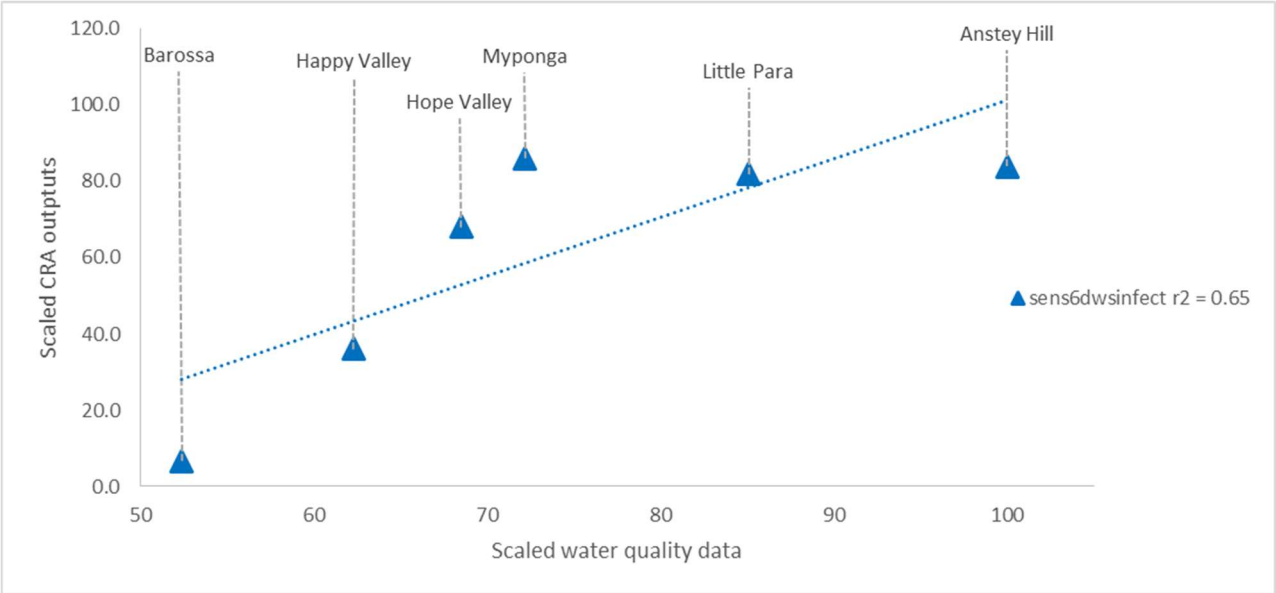


Figure 8.2: SAW scaled *sens6* (all risks) model infectivity risk output (Scaled CRA outputs) compared with observed end-of-catchment pathogen concentrations (Scaled water quality data) (Swaffer 2014, p. 41) (P = 0.05)

The corresponding calculation for the *sens11* (flow modifier) infectivity values produced (Figure 8.3):

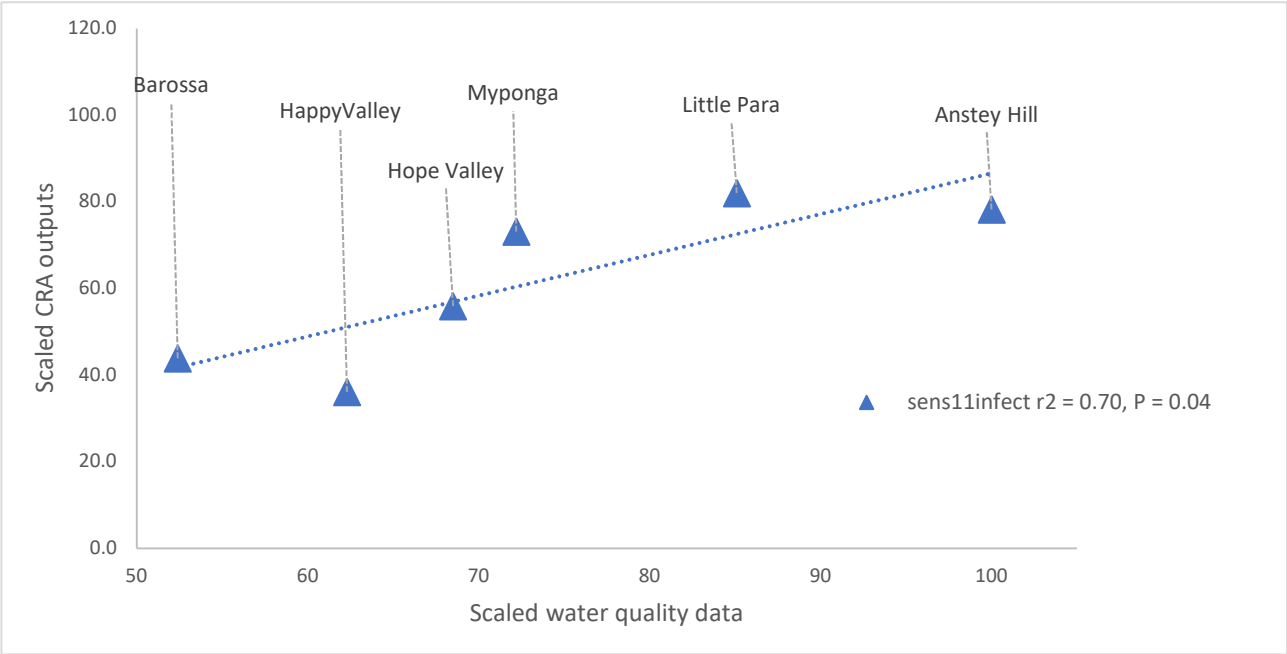


Figure 8.3: Scaled *sens11* (flow modifier) model infectivity risk output (Scaled CRA outputs) compared with end-of-watershed pathogen concentrations (Scaled water quality data)

Finally, the *sens11* (flow modifier) data for nutrients is presented below (Figure 8.4):

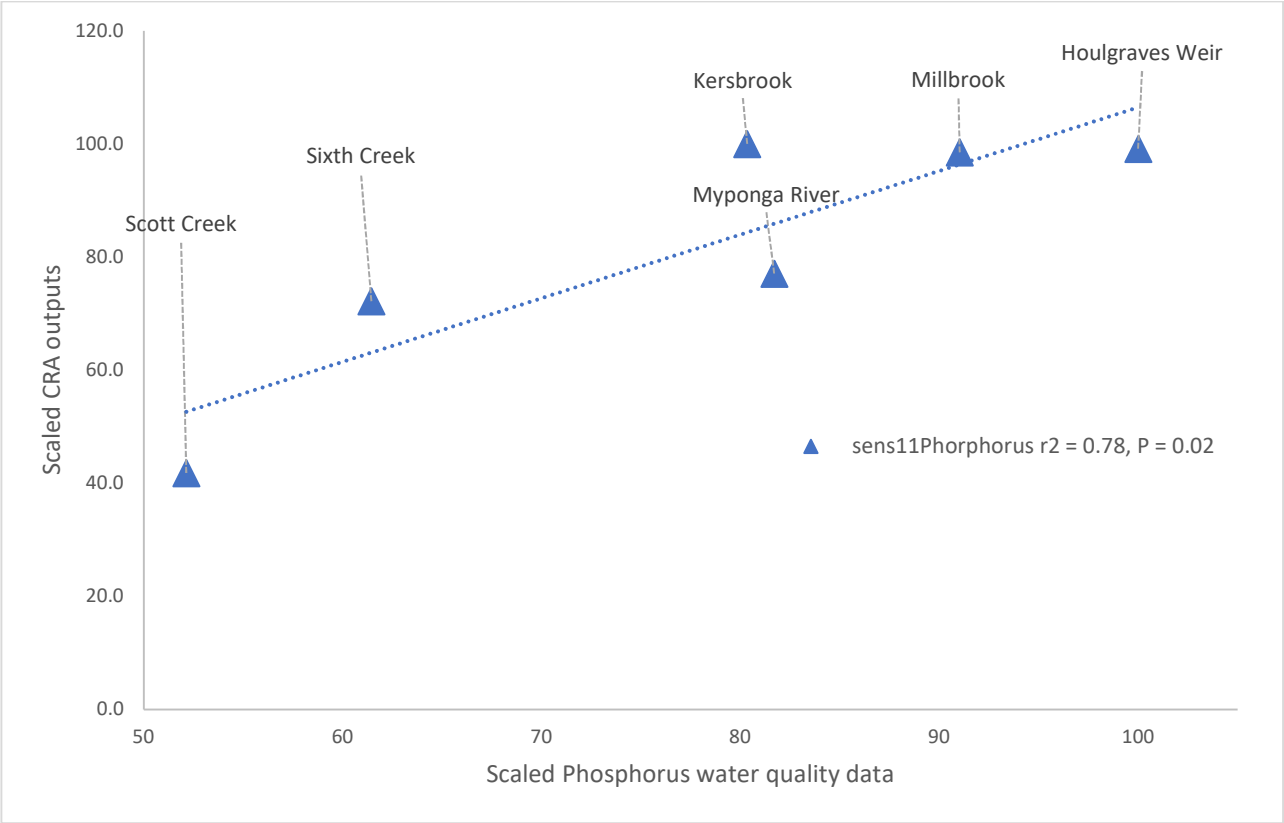


Figure 8.4: Scaled *sens11* (flow modifier) model risk output (Scaled CRA outputs) compared with end-of-watershed flow-weighted nutrients (Total Phosphorus Kg/ML/yr)

The trend lines above are described by the following corresponding equations:

sens6 (all risks) pathogens model:

$$sens6_{infectivity} \text{ risk} = 1.54 * (\text{Pathogen scaled water quality data}) - 53.0 \quad \textbf{(Eq. 8.1)}$$

sens11 (flow modifier) pathogens model:

$$sen11_{infectivity} \text{ risk} = 0.94 * (\text{Pathogen scaled water quality data}) - 7.6 \quad \textbf{(Eq. 8.2)}$$

sens11 (flow modifier) nutrients model:

$$sen11_{Phosphorus} \text{ risk} = 1.12 * (\text{Phosphorus scaled water quality data}) - 6.0 \quad \textbf{(Eq. 8.3)}$$

The *sens11* versions of the model display Y-intercepts closer to the expected mean of zero than the original *sens6* version.

A summary of the regression studies can be found below (Table 8.2). It reveals a steady decline in the R^2 values as the pathogen model is refined. The improvements in the last variation reflect a review of the underlying risk estimate for the tbuffer variable and the weights of tfence and train. The intent of the review was to reduce the generation of negative risk outcomes while avoiding as much as possible modifications to the original SAW settings.

Table 8.2: Summary R^2 values for each development of the model

Pollutant	*sens6* all risks	*sens7* all risks	*sens8* exclude zero source	*sens9* water-sheds	*sens10* flow	*sens11* flow modifier
Pathogens	0.60	0.60	0.58	0.54	0.49	0.61
Pathogen infectivity	0.65					0.70
Nutrients		0.63				0.78

8.2 Discussion of model validation

The practical approach to performing regression analysis has been to extract the Zonal Statistics means for each area (catchment or watershed) and then manually copy them to a spreadsheet for the regression computation. This is tedious and prone to error. It would be a simple matter to capture the statistical means within the Python application and perform the regression calculation there. This could include plotting as required.

The model has gained some more realistic characteristics during its development from the sum of linear terms (*sens6/7*) to the interaction of three groups of these terms (*sens11*). In this latter form, it appears to have gained in its predictive capability when applied to nutrients. This is a qualified position. In this context, "nutrients" relates only to that behaviour of Phosphorus that participates in the biological activities being modelled here.

More formally, there remain significant aspects of the model to be tested to establish if it is linear, or that this is a reasonable assumption (Hastie 2009, p. 236; Rogerson 2015):

- Are the predictive variables independent of each other or is a degree of collinearity being exhibited?
- Do the errors have a normal distribution about the regression line?
- Is the mean of the errors zero and the variance constant (i.e., homoscedasticity is preserved)?
- Do any of the variables express spatial autocorrelation?

Variables have been scaled against their own estimated numerical maximum. The variability in spatial scale between variables should be addressed. These, and other attributes, require an evaluation that is beyond the scope of this project. They have a practical application, however. For example, it seems reasonable to expect that high intensity agricultural activity is correlated with rainfall. Perhaps, also, there is a degree of autocorrelation within the industries associated with a

source variable because of the existence of supporting culture and infrastructure amongst property owners.

Finally, consideration should be given to expanding the number of observation points available to this modelling effort. This would be supported by the new strategy to determine the watersheds associated with monitoring stations. It would open up the study to the examination of non-linear behaviours in higher order models and to the application of automated regression analyses.

9. DISCUSSION AND CONCLUSIONS

The work described here is a response to the Water Services Association of Australia's (WSAA 2015, pp. 13-9) Tier 1 (mandatory) requirements for assessing a drinking water supply system. In this instance, SA Water carried out sanitary surveys to identify pathogen (i.e., *Cryptosporidium*) sources and intensity. The microbial indicator assessment was derived from measurements of *Cryptosporidium* at end-of-catchment monitoring stations above each of the six water treatment plants in the MLR. The vulnerability assessment task was addressed by developing a model to assess the risks of *Cryptosporidium* contamination. This model was established as a Raster Calculation in ArcMap. It was accompanied by an extensive collection of data in the form of spreadsheets, shape files and raster files. Two unpublished reports provided supporting material from the literature to document the sources of the data and, with the results of the sanitary surveys, to justify settings for stocking and infection rates, etc. These reports also outlined the workflow for generating a contamination risk surface for the MLR catchments and for validating this work. This substantial body of work formed the baseline of the current project.

9.1 Limitations of the previous model

At the level of development it had reached, the previous model was constrained in a number of ways:

- the terms were always added (or subtracted), leading to a value for risk derived from the transport terms even if there were no pathogens present
- flowing water (slope was the proxy for this) should be required for the transport of pathogens. In the absence of water flow, a risk was still generated
- substantial negative values for risk could be generated by the model due to the impact of the buffer and fencing variables
- the risk value of each variable and the weighting applied to it were fixed during the execution of the model
- the GIS package was cumbersome to use when sensitivity analyses were performed
- there was no built-in support for record keeping during a series of experiments
- only the risk from pathogens was being modelled
- the DEM for the Myponga catchment contained errors (it is not clear if the slope term used in the *sens6* and earlier versions was derived from this DEM)

The establishment of the pathogens model benefitted from the exercise of considerable professional judgement for the estimation of variable risks and weights. This enabled the development in ArcMap of a comprehensive pathogen risk model for water catchments. However, subsequent tuning of such a model presented difficulties. For example, a problem with conducting a what-if study manually is that a local minimum or maximum may distract the focus of the user.

This could lead to the acceptance of a model that has less predictive power than might otherwise be the case.

A large GIS system provides all the tools needed to develop a modelling system. However, when used for data maintenance, when carrying out sensitivity studies or in comparing the behaviours of different pollutants, those same tools can become unwieldy and intrude on the workflow.

9.2 Responses to limitations of the previous model

This project has addressed the concerns above with the development of a different implementation of the workflow in Python that hides the idiosyncrasies of the GIS package. The goal has been to provide a simple user interface and familiar tools to support data maintenance, sensitivity studies and reporting. This goal has been achieved. Further, enhancements to the underlying model have also been facilitated by the Python system.

The output of this work has been a group of Python scripts that largely implement the original workflow and add the tools described above. Some new functionality still runs under ArcGIS as there was insufficient time to complete the Python development. The main outcomes of the project include the ability to carry out the basic workflow more conveniently, to gain new insights into the data and to extend the model to deal with new pollutants and behaviours.

Most of the SAW workflow has been reduced to simple point and click operations that enable a user to:

- incorporate updated data into shapefiles
- output new model variable raster files from the shapefiles
- generate new MLR risk surfaces to describe the distribution of the risk of pathogen contamination
- create reports

Standard features of spreadsheets have been exploited to support record-keeping during sensitivity studies. Some new analysis capabilities have also been provided in the form of a fencing risk mitigation guide. This technique could be applied more generally to address some of the recommendations for improvement suggested in Swaffer (2014, pp. 55-6). These recommendations included identifying the principal contributor to the risk at a location, e.g., land use, irrigation, distance to watercourse, etc.

A Python script was developed to elaborate the contribution of each model variable to the overall risk. This has led to a better understanding of the behaviour of all the variables in relation to each other. The transport variables appear to exert a greater influence on the output than might have been expected. In particular, the buffers term appears to predominate.

In this project, the negative outcomes generated by the mitigation terms buffers and, to a lesser extent fencing, were addressed by reducing their risk value and weight respectively. Swaffer (2014, p. 54) proposed to avoid the negative risk results by dividing by the mitigation terms rather than subtracting them. This would not work for the fencing variable as it would lead to a divide-by-zero error. The effect of dividing by the buffers value has not been investigated here but see the comments below.

A sequence of developments has been undertaken on the model itself to improve its representation of the natural systems:

- only cells with non-zero pathogen (or nutrients) values contribute to the final risk
- only cells that are members of the contributing area above monitoring stations (called watersheds here) are included in the risk calculation, leading to a more refined measure of risk than averaging across whole catchments
- the slope term has been replaced by a water flow rate that is influenced by upstream flows
- errors in the DEM were discovered and corrected
- the first steps have been taken to generalise this modelling system so it can be applied to other pollutants apart from pathogens

Fundamentally, pathogens (or Phosphorus in the nutrients version) must be present and water must be flowing for any given cell to contribute to the final risk outcome. Expression has been given to this behaviour by multiplying the source group of variables by the transport group and the flow rate term. However, this initial application of a flow variable in the model will benefit from further refinement. While it introduces a useful geographical weighting to the behaviour of water, it is still a static term. There are situations, e.g., in reservoirs, where a static value of zero is not always useful. In fact, variable detention times and multiple extraction points (horizontal and vertical) would result in a variety of risk values for a given reservoir (NH&MRC 2011, pp. 33-4). The problem of static values is discussed further below.

Superficially, the suggestion that the accumulating risk be divided by buffers appears to offer a similar benefit to the use of flow above. However, there is a significant difference. A cell's flow rate risk varies spatially across a watershed depending on the contributing area above it. Conversely, the buffers term is assigned an average risk mitigation amount across whole catchments (or two or three subdivisions of them). By analogy with the flow determination, an understanding of the hierarchy of contributing secondary storages on the stream network may accommodate better the use of a buffer divisor. As oocysts are transported down through the stream network, the impedance contributed by secondary storages could be accumulated.

Infectivity is another term that, like buffers, has been assigned a catchment-wide value. In this case, some values have been derived from measurements, while others are reasonable estimates (Swaffer 2014, pp. 40-1). On the other hand, rainfall varies generally across the MLR. Still other terms vary in relation to a feature such as a reservoir or a stream. Finally, fencing and irrigation are given values specific to a local precinct.

In summary, the current model (excluding the new flows term) provides for a spatial disposition that fixes values for each cell. The next development of the model could consider the impact of geographical weighting, allowing the risk contribution of a cell to be influenced by the values of others around it.

9.3 Further work

The principal recommendation for further work is that a deeper understanding of the data be sought. In particular, where are spatial cross-correlation and autocorrelation expressed and how can these be accommodated in the model? This effort would be aided by a more elaborate data set containing sufficient observations to provide both training and testing data sets. Given the new capability of working with watersheds associated with monitoring stations, the smaller sub-catchments studied in Swaffer et al. (2018) might prove useful here.

The availability of a larger data set would support the application of one or more automated approaches to regression, parameter estimation and selection such as MARS (Multivariate Adaptive Regression Splines) (Friedman 1991; Hastie 2009, pp. 321-7; Milborrow 2019; Rudy 2013) or CART (Classification and Regression Tree) (Wilkes et al. 2011). These strategies would be expected to lead to a more parsimonious model.

This leads to the question of what variables should be available to the regression analysis. The current study introduced a term to model the behaviour of water. It may be appropriate to introduce another factor that alters that behaviour, e.g., some land cover types would be expected to retard water flow, leading to a reduction in oocyte infectivity. Similarly, in addition to pollutants having sources, they could also have behaviours attributed to them that could be modified. For example, Phosphorus flow can be affected by adsorption, plant uptake or leaching from the soil.

The existing model already includes secondary storages (buffers) to retard the flow of *Cryptosporidium*. It currently deducts buffer risks from the total risk. Buffers might instead work as a modifier of the source group although it would not be directly analogous to flow. Flow is a rate modification whereas buffers apply a delay in the movement of *Cryptosporidium* towards the primary storage. This introduces another concept that is missing from the existing system, i.e., time.

The current model avoids having to deal with time by expressing values in terms of, e.g., annual or flow-rated averages. However, the natural systems being modelled may vary continuously over time or they may exhibit discontinuous behaviours. For example, watercourses cease flowing in the hotter months and will not resume their flow for some time after the first rains of the season. Muirhead, Elliott and Monaghan (2011) note that sporadic inputs to streams do not affect median concentration values (such as are used in the SAW model) but they do have a large effect on 95 percentile values. Roser and Ashbolt (2007) address this problem by proposing a qualitative microbial risk assessment that incorporates a consideration of run-off conditions. Further, the sources of pathogens may also behave episodically. Peaks in human health risk appear to be aligned with the breeding cycles of, especially, dairy cattle, rather than peak water flows (Swaffer et al. 2018). These observations suggest that an event-driven model (in this case, based on a Python queue) would have merit.

The risk values of each of the variables and their corresponding weights are currently fixed during the execution of the model. These values could be sampled from distributions characteristic of each variable. With time available to the model, this sampling could reflect the fact that the behaviour of variables changes over time, e.g.:

- ambient temperature changes over a year that are likely to affect the viability of *Cryptosporidium* oocysts (King et al. 2005)
- rainfall variations over time that can be varied by sampling from a distribution
- *Cryptosporidium* oocysts themselves may exhibit a distribution in their viability rates at the time of their dispersal (Jenkins et al. 1997).

It is a relatively simple modification to the Python script to accommodate data and decision-making regarding different pollutants. It is less simple to express models that represent different pollutant behaviours. The behaviour of each pollutant could be captured in its own Python module that would be activated according to the user's choice. For example, it has been assumed for this study that the transport of pathogens or Phosphorus relies on surface water movements only. If Nitrogen is to be considered, aspects of the sub-surface strata relevant to its transport would require description in the model. This task is best defined inside the nutrients' own class definition. The corresponding pollutant object would be created at run time according to the user's choice and without impacting on the main script.

Such modularisation doesn't just provide for distinguishing between pollutant groups. Within a group, the behaviours of the pollutants could also be modelled. So too could a variety of preventative or mitigation measures and barrier management strategies (NH&MRC 2011, pp. 1120-9).

The Python setting creates a rich environment for further development of this model in ways that might not be possible in the native GIS environment.

9.4 Conclusions

ArcGIS provides interfaces to a number of programming languages, including Python. It is now possible to implement, as free-standing Python applications, model systems that were previously wholly contained in ArcGIS. Any such new system is easy to use for maintenance activities and sensitivity studies. It also provides new insights into how the data behave, which in turn can feed back into the design of the model.

What has emerged from this project is the understanding that, in addition to generic transport variables, there are for each pollutant:

- an accompanying model
- source/transport variables
- associated behaviours.

Python's support for modules, functions and classes would be well suited to representing these entities. The existing main line code would retain its roles in user interaction, data maintenance and reporting while the details of each pollutant's data requirements and behaviours could be hidden inside its own class definition.

The information-hiding capabilities of Python present opportunities for enhancement of the model in a modular fashion. It provides for the expression of spatial attributes and behaviours that were not formerly part of the model. This will lead to the creation of a model that is generalisable across pollutant domains and contribute to a deeper understanding of the risks of water contamination in the Mount Lofty Ranges.

APPENDICES

Appendix A: SA Water Plan - GIS-Enabled Water Quality Risk Assessment in a Drinking Water Catchment

Background

The project will be conducted by Ian McDonald in part fulfilment of his candidature for the degree of Master in GIS. The Ian will be supervised by Prof Howard Fallowfield, College of Science and Engineering with the assistance of ….. SAW will provide in kind support through access to facilities, staff and relevant databases.

Project outline

I. <u>Please note:</u>

- number of catchments (1 or more general) can be adjusted, but the desire is to have Ian work on the total of the Mount Lofty Ranges Watershed, as our current CRA is based on it (see attached general info and map; still relevant, but a bit old)
- Focus on risk assessment can be adjusted according to the speed and desire the student wants to undertake one or different risk assessments, using the existing (GIS-based) protocol, e.g., (in priority order):
    - Review and revise / update the current pathogen CRA
    - Conduct the CRA for nutrients (TP, TN) and sediments (TSS) export
    - Conduct the CRA for pesticides

II. <u>Desired outcomes/outputs:</u>

a. CRA outputs similar to the attached
b. Report (see below for suggestion of content)
c. Hand over to SAW of all literature, files, modification of GIS configurations (including new python scripts) in useable format

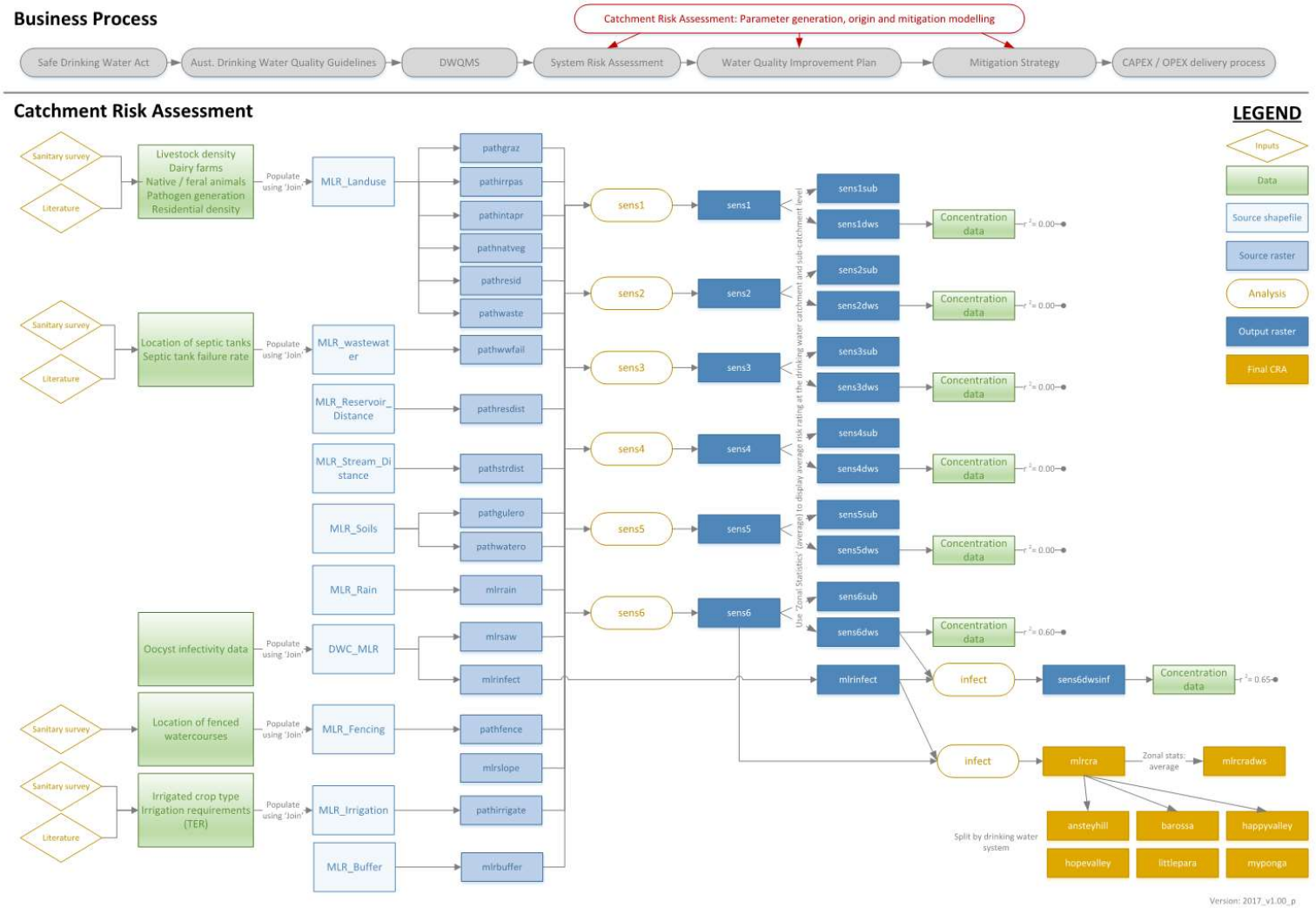
III. <u>Steps would include:</u>

1. Familiarisation with SA Water's GIS-based protocol/method currently used for our existing pathogen Catchment Risk Assessment (CRA) and progress made to date
    a. Review the data which supports the quantification of pathogen risk by various land use categories.
    b. Update the data with new literature (some good papers have come out recently on oocyst shedding in animals which should be incorporated)

2. Convert the step by step procedure into Python script
3. Inform/update/modify existing (pathogen) CRA in GIS for new constituents (e.g., nutrients, sediments),
    a. ‘Parameterisation’ of nutrient exports from various land uses/ land management types via
        i. Targeted literature review and building a literature ‘library’ from which to draw data used to populate the nutrient export from various land use types (SAW also has some land use/nutrient information available)
        ii. Build/adapt a set of shapefiles used to estimate nutrient export. Updating GIS input layers (e.g., through populating a set of excel tables which will be used to populate attributes of shapefiles)
        iii. Selected ground truthing in the field might be needed
    b. Combine the source shapefiles into a single output, ensuring to undertake a sensitivity analyses
    c. Verify the output using end of catchment data (i.e. model outputs, composite sampler data etc
    d. Undertake or recommend verification (water quality monitoring, if significant data gaps are present in the verification data sources
4. Document the step by step instructions for undertaking the risk assessment, from the creation of shapefiles from excel tables, all the way through to the output rasters.

Report could either be focused on just nutrients, or bring both the pathogen and nutrient CRA conceptualisation, literature review, methodology, outputs, and discussion of value into a thesis format

Appendix B: SA Water Catchment Risk Assessment Workflow



(Swaffer 2014, p. 50, used with permission)

Appendix C: SAW Process Map for ArcMap

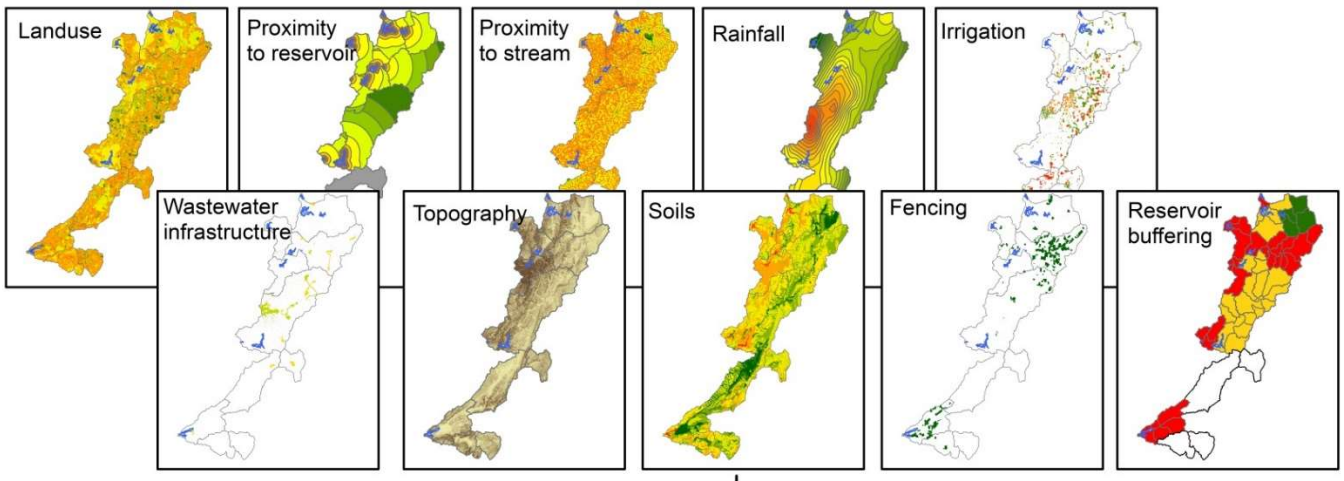
Catchment Risk Assessment process map

version: 2015_v1.00_p

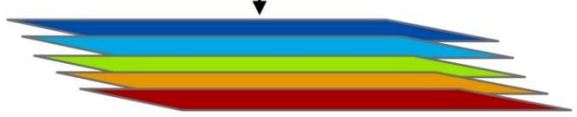
Literature

Pathogen characteristics (prevalence, infectivity)	Agricultural information (Stock type, stocking rates)	Hydrological characteristics (rainfall, proximity to stream)	Landscape information (slope, soil erodability)

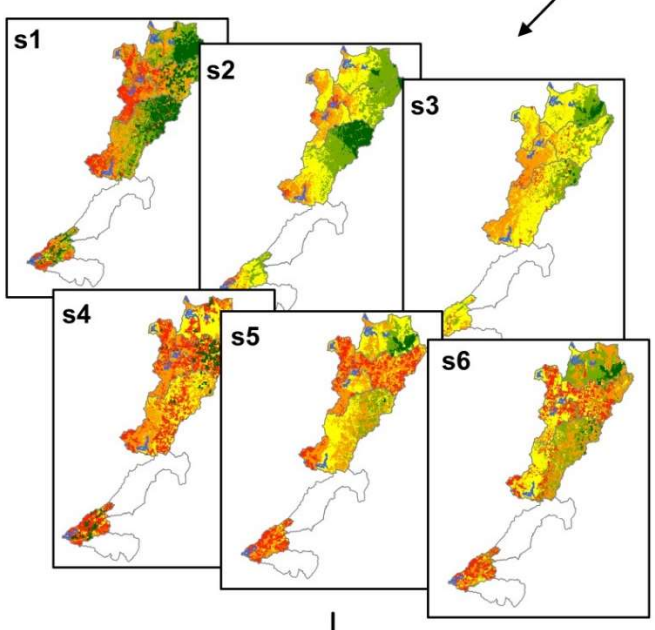
Spatial data



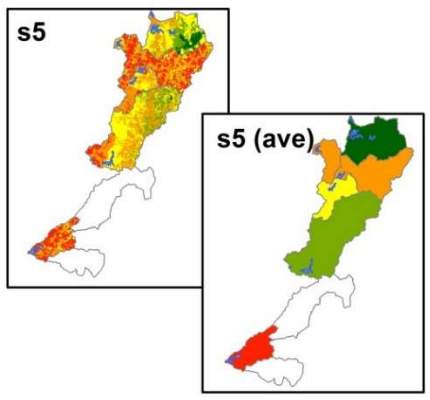
Expert elicitation (weighted layers)



Sensitivity analysis



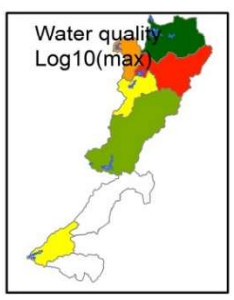
WEIGHTINGS USED	Equal weighted	Reservoir proximity	Hydrology	Fencing	Reservoir storage	Landuse
RISK	**s1**	**s2**	**s3**	**s4**	**s5**	**s6**
Native vegetation	7.1429%	2.5%	2.5%	5.0%	5.0%	5.0%
Residential areas	7.1429%	2.5%	2.5%	10.0%	10.0%	10.0%
Grazing	7.1429%	2.5%	2.5%	25.0%	25.0%	25.0%
Intensive animal	7.1429%	2.5%	2.5%	7.5%	7.5%	7.5%
Irrigated pasture	7.1429%	2.5%	2.5%	15.0%	15.0%	15.0%
Wastewater landuse	7.1429%	2.5%	2.5%	5.0%	5.0%	5.0%
Wastewater failure risk	7.1429%	2.5%	2.5%	2.5%	2.5%	2.5%
Distance to reservoir	7.1429%	25.0%	5.0%	5.0%	5.0%	5.0%
Distance to stream	7.1429%	25.0%	5.0%	5.0%	5.0%	5.0%
Rainfall	7.1429%	10.0%	20.0%	10.0%	10.0%	10.0%
Topography	7.1429%	7.0%	20.0%	2.5%	2.5%	2.5%
Gully erosion	7.1429%	5.0%	6.0%	2.5%	2.5%	2.5%
Water erosion	7.1429%	5.0%	6.0%	2.5%	2.5%	2.5%
Irrigation	7.1429%	5.0%	20.0%	2.5%	2.5%	2.5%
TOTAL	**100%**	**100%**	**100%**	**100.0%**	**100.0%**	**100.0%**
MITIGATION						
Reservoir storage	7.1429%	10.0%	10.0%	10%	25%	10%
Fencing	7.1429%	5.0%	5.0%	50%	5%	5%
TOTAL	**14%**	**15%**	**15%**	**60%**	**30%**	**15%**

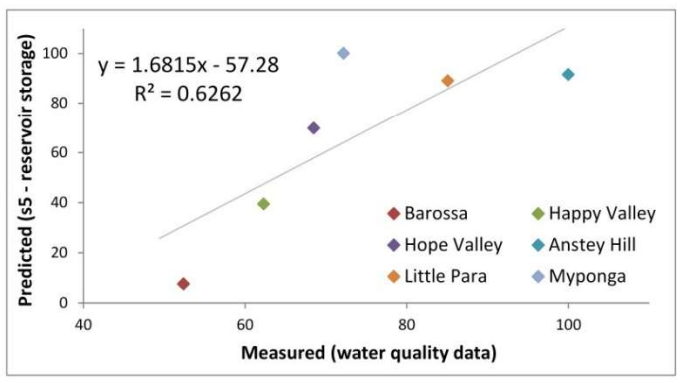


Sensitivity analysis		Average CRA raster value - r = raw, s = scaled												Comparison to WQ data
Description		Barossa		Happy Valley		Hope Valley		Anstey Hill		Little Para		Myponga		
		r	s	r	s	r	s	r	s	r	s	r	s	r^2
s1	Equal weighted	15.96	70.35	18.24	80.41	22.68	100	18.65	82.21	21.18	93.4	19.4	85.55	0.12
s2	Reservoir proximity	43.83	84.72	42.59	82.33	51.74	100	45.49	87.93	50.64	97.89	48.16	93.09	0.09
s3	Hydrology	17.66	61.51	23.41	81.54	28.71	100	22.13	77.08	23.80	82.91	23.16	80.67	0.05
s4	Fencing	11.28	65.83	13.88	80.98	17.14	100	8.41	49.08	16.58	96.77	11.66	68.06	-0.01
s5	Reservoir storage	1.028	7.531	5.37	39.35	9.518	69.69	12.49	91.48	12.15	88.98	13.66	100	**0.63**
s6	Landuse	12.34	64.25	14.79	77.04	17.36	90.4	18.04	93.98	17.69	92.13	19.2	100	0.52

Water quality data

System	Max (pres. 10 L)	Log10(Max)	Log10(Max) (scaled)
Barossa	70	1.85	**52.4**
Happy Valley	157	2.20	**62.3**
Hope Valley	258	2.41	**68.5**
Anstey Hill	3328	3.52	**100.0**
Little Para	990	3.00	**85.1**
Myponga	350	2.54	**72.2**





(SA Water Corporation 2014, p. 19, used with permission)

Appendix D: Variables used in the SAW pathogens risk model (SA Water Corporation 2014, pp. 6-7) and (Swaffer 2014, pp. 21-30) (renamed for this project)

#	Source raster layer	Layer name	Comment
1	SA Water	mlrsaw (tsaw)	Used to delineate drinking water catchments from non-drinking water catchments (such as Finniss River, Hindmarsh River etc)
2	Water Treatment Systems	DWC_SAW	Used to delineate WTP systems
3	Subcatchments	MLR_Subcatchment	Used to delineate subcatchments across the Mount Lofty Ranges (SDEDBA.MLRSubCatchment)
4	Landuse - native vegetation	pathnatveg (pnatveg)	Pathogen risk from native vegetation, based on density and composition of feral animals, manure production, oocyst concentration and infected animal prevalence (Landuse2008_ALUMv6)
5	Landuse - residential areas	pathresid (presid)	Pathogen risk from residential zones, based on dwelling density (Landuse2008_ALUMv6)
6	Landuse - grazing	pathgraz (pgraz)	Pathogen risk from grazing areas, based on stock type and stocking rate, manure production, oocyst concentration and infected animal prevalence (Landuse2008_ALUMv6)
7	Landuse - intensive animal production	pathintapr (pintapr)	Pathogen risk from intensive animal production, based on stock type and stocking rate, manure production, oocyst concentration and infected animal prevalence (Landuse2008_ALUMv6)
8	Landuse - irrigated pasture	pathirrpas (pirrpas)	Pathogen risk from irrigated pasture (assumed to be dairy farms) and based on stocking rate, manure production, oocyst concentration and infected animal prevalence (Landuse2008_ALUMv6)
9	Landuse - waste	pathwaste (pwaste)	Pathogen risk from sewerage treatment plants (landfill removed) based on maximum treatment capacity (ML/day) (Landuse2008_ALUMv6)
10	Wastewater failure	pathwwfail (pwwfail)	Pathogen failure rate from sewerage infrastructure. Unsatisfactory septic tanks or upgrade pending status presented the highest failure rate, council-operated sewer networks presented a moderate rate, with SA Water operated systems the lowest rate
11	Reservoir	pathresdist (tresdist)	Proximity to reservoir. Risk is greater with closer proximity to the water body (SDEBA.Reservoir)
12	Stream	pathstrdist (tstrdist)	Proximity to stream. Risk is greater with closer proximity to stream (SDEBA.MLROrderedStreams)
13	Rain	mlrrain (train)	Rainfall. Risk is greater in higher rainfall areas (MLRAnnualRainfall)
14	Slope	mlrslope (tslope)	Slope. Risk is greater in areas with steeper slopes
15	Soils – gully erosion	pathgulero (tgulero)	Soils. Risk is greater on highly erodible soils (SDEBA.SoilsSA2007)
16	Soils – water erosion	pathwatero (twatero)	Soils. Risk is greater on highly erodible soils (SDEBA.SoilsSA2007)
17	Irrigation	pathirrigate (tirrigate)	An annual irrigation volume (ML/ha) was assigned to the land parcel based on the likely crop type. Risk is greater on parcels with higher irrigation requirements (Landuse2008_ALUMv6)
18	Fencing	pathfencing (tfence)	Pathogen risk is lowered by the presence of a watercourse fence.
19	Reservoir dilution	mlrbuffer (tbuffer)	Pathogen risk is lowered by the presence of upstream storages.

Appendix E: Python script of main application

```
import arcpy
from arcpy import env
import os
from Tkinter import *
import datetime
import ScrolledText as scrolledtext
from arcpy.sa import *
import re
from shutil import copyfile
import numpy as np
import matplotlib.pyplot as plt
import xlrd

MLRDIR = '<User directory>'
VERLEN = 3
DEBUG = True

arcpy.CheckOutExtension("Spatial")
arcpy.env.extent = "MAXOF"
arcpy.env.overwriteOutput = True
arcpy.env.qualifiedFieldNames = False
arcpy.env.cellSize = "FinalRaster\mlrcra"
arcpy.env.workspace = MLRDIR
arcpy.env.scratchWorkspace = MLRDIR

srcSpreadsheetsDir = MLRDIR + "SourceSpreadsheets\\"
srcShapesDir = MLRDIR + "SourceShapefiles\\"
shapesDir = MLRDIR + "Shapes\\"
rastersDir = MLRDIR + "RastersWS\\"
reportsDir = MLRDIR + "Reports\\"
risksDir = MLRDIR + "Risks\\"
mapTemplatesDir = MLRDIR + "MapTemplates\\"

# Read raster names for each shapefile from layers text file and store in shapeVars dictionary
shapeVars = {}
with open(MLRDIR + '/layersListWS.txt','r') as input_file:
    for line in input_file:
        shapeName = line.split(':')[0]
        rasters = line.split(':')[1].strip('\n')
        rastersTuple = tuple(rasters.split(','))
        shapeVars[shapeName] = rastersTuple
input_file.close()
print "shapeVars =", shapeVars, '\n'

OPTIONS = sorted(shapeVars.keys())
if DEBUG:
    print "OPTIONS = ", OPTIONS

root = Tk()

rasSrc = StringVar(root)
rasSrc.set("Select source data")
pSrc = StringVar()       # pollutant source
pSrc.set("p")            # i.e., default choice is Pathogens
dwcVar = StringVar(root)  # set default as for rasSrc?
dwcVar.set("Select catchment")
dwcDict = {}
dwcChoices = []
refNum = StringVar(root)     # Reference number for the risk map legend

# Create the containers for the GUI
frame0 = Frame(root, width=440, height=45, pady=3)
frame1 = Frame(root, width=440, height=60, pady=3)
frame2 = Frame(root, width=400, height=40, pady=3)
frame3 = Frame(root, width=440, height=60, pady=3)
frame4 = Frame(root, width=440, height=40, pady=3)
```

```
frame5 = Frame(root, width=440, height=15, pady=3)

# lay out the containers
root.grid_rowconfigure(1, weight=1)
root.grid_columnconfigure(0, weight=1)
frame0.grid(row=0)              # pollution source radio buttons
frame1.grid(row=1)              # shapefile dropdowns & shape/raster buttons
frame2.grid(row=2)              # Reporting buttons
frame3.grid(row=3)              # Analyse buttons
frame4.grid(row=4)              # message window
frame5.grid(row=5)              # Quit button

# Message window needs to be global to be accessible from different functions
msgWin = scrolledtext.ScrolledText(frame4, width = 50, height = 4)

class BusyManager:
    # Adapted from http://effbot.org/zone/tkinter-busy.htm
    # Viewed on 22 August 2018

    def __init__(self, widget):
        if DEBUG:
            print "Entering BusyManager._init_"
        self.toplevel = widget.winfo_toplevel()
        self.widgets = {}

    def busy(self, widget=None):
        if DEBUG:
            print "Entering BusyManager.busy"

        # attach busy cursor to toplevel, plus all windows
        # that define their own cursor.

        if widget is None:
            w = self.toplevel # myself
        else:
            w = widget

        if not self.widgets.has_key(str(w)):
            try:
                # attach cursor to this widget
                cursor = w.cget("cursor")
                if cursor != "watch":
                    self.widgets[str(w)] = (w, cursor)
                    w.config(cursor = "watch")
            except TclError:
                pass
        for w in w.children.values():
            self.busy(w)

    def notbusy(self):
        if DEBUG:
            print "Entering BusyManager.notbusy"
        # restore cursors
        for w, cursor in self.widgets.values():
            try:
                w.config(cursor=cursor)
            except TclError:
                pass
        self.widgets = {}

def purge(dir, pattern):
    # From: https://stackoverflow.com/questions/1548704/delete-multiple-files-matching-a-pattern/38189275
    # Accessed 13/03/2019
    if DEBUG:
        print "Entering purge"
    for f in os.listdir(dir):
        if re.search(pattern, f) and (f[-5:] != ".lock"):
            os.remove(os.path.join(dir, f))

def printSpreadSheetFieldNames(fName):
```

```
    if DEBUG:
        print "Entering printSpreadSheetFieldNames"
        for i in range(0, len(fName)):
            print fName[i],
        print "\n"

def printSpreadSheetData(spreadSheet):
    if DEBUG:
        print "Entering printSpreadSheetData"
    with arcpy.da.SearchCursor(spreadSheet, '*') as cursor:
        for row in cursor:
            for i in range(0,len(row)):
                print row[i],
            print "\n",
    del cursor

def getCurrentVerName(fList, fName):
    if DEBUG:
        print "\nEntering getCurrentVerName with fName =", fName
    listOfFiles = [f for f in fList if fName in f]
    if len(listOfFiles) > 0:
        latestVer = listOfFiles[-1]
    else:
        latestVer = ""
    return latestVer

def genNewVerName(fList, fName):
    if DEBUG:
        print "\nEntering genNewVerName with:"
        print "fName =", fName
#       print " flist =",  flist
    latestVer = getCurrentVerName(fList, fName)
    if latestVer != "":
        oldFn, oldFtype = os.path.splitext(latestVer)
        oldVerNum = oldFn[-VERLEN:]
        newVerNum = str(int(oldVerNum) + 1).zfill(VERLEN)
        newVer = oldFn.replace(oldVerNum, newVerNum)
    else:
        newVer = fName + str(0).zfill(VERLEN)
    return newVer

def readModel():
    if DEBUG:
        print "Entering readModel"
    pS = pSrc.get()
    if pS == 't':
        pS == 'p'  # Not dealing with transport variables as separate task yet
    inSpreadSheet = MLRDIR + pS + "modelflows.xlsx"
    rcDict = {}

    # Get first sheet of spreadsheet - this is the most recent model
    workbook = xlrd.open_workbook(inSpreadSheet, on_demand=True)
    sheetNames = workbook.sheet_names()
    sheetName = sheetNames[0]
    workbook.release_resources()

    inMemorySheet = "in_memory" + "\\" + "model"

    # Read model spreadsheet into in-memory table
    arcpy.Delete_management(inMemorySheet)
    arcpy.ExcelToTable_conversion(inSpreadSheet, inMemorySheet, sheetName)

    # Build dictionary of model layer name (the key) and corresponding coefficients and operators
    with arcpy.da.SearchCursor(inMemorySheet, '*') as cursor:
        for row in cursor:                          # {rasname: [weight, operator, rasSum]}
            rcDict[row[1]] = [row[2], row[3], 0.0]  # rasSum will store total contribution of
    del cursor                                      # each raster to total risk for later plotting
    arcpy.Delete_management(inMemorySheet)
    refNum.set(rcDict.get('Reference', "None")[1])  # Extract Reference Number for Risk map legend
    del rcDict['Reference']                         # and delete it from the model dictionary
```

```
    if DEBUG:
        print "leaving readModel with rcDict =", rcDict
    return rcDict

def msgOut(txt):
    if DEBUG:
        print "Entering msgOut with txt =", txt
    msgWin.insert(END, txt + "\n")
    msgWin.see(END)
    msgWin.update()

def shapesUpdate():
    if DEBUG:
        print "Entering shapesUpdate"
    inShapesDir = srcShapesDir
    outShapesDir = shapesDir
    arcpy.env.workspace = shapesDir
    arcpy.env.scratchWorkspace = shapesDir
#    outRastersDir = rastersDir
    inSpreadSheetsDir = srcSpreadsheetsDir
    inSheetName = "AttributeTable"
    inMemorySheet = "in_memory" + "\\" + "attribTable"
    inShapeName = rasSrc.get()
    inSpreadSheet = inShapeName + ".xlsx"
    outShapeName = genNewVerName(arcpy.ListFiles("*.shp"), inShapeName + 'v')
    print "shapefiles list =", arcpy.ListFiles("*.shp")
    print "New shapefile name =", outShapeName

    manager = BusyManager(root)
    manager.busy()

    # read spreadsheet into memory
    # https://gis.stackexchange.com/questions/138043/using-arcpy-to-read-from-excel-spreadsheet-to-python-dictionary
    # Accessed 4/6/2018
    arcpy.Delete_management(inMemorySheet)
    arcpy.ExcelToTable_conversion(inSpreadSheetsDir + inSpreadSheet, inMemorySheet, inSheetName)

    # Get field names from spreadsheet (not OID)
    dsc = arcpy.Describe(inMemorySheet)
    fields = dsc.fields
    fieldnames = [field.name for field in fields if field.name != dsc.OIDFieldName]
    # uncomment the following lines to check the spreadsheet headings and contents
    #printSpreadSheetData(inMemorySheet)
    #printSpreadSheetFieldNames(fieldnames)

    # Copy source shapefile to new version of shapefile
    arcpy.FeatureClassToFeatureClass_conversion(inShapesDir + inShapeName + ".shp", outShapesDir, outShapeName
+ ".shp")
    msgOut("Updating copy of shapefile: " + outShapeName)

    # Copy spreadsheet rows over new shapefile attribute table rows
    rowsCur = arcpy.UpdateCursor(outShapesDir + "\\" + outShapeName + ".shp")
    rowsUpd = arcpy.SearchCursor(inMemorySheet)
    rowCur = rowsCur.next()
    rowUpd = rowsUpd.next()
    j = 0
    while rowCur:
        for i in range (2, len(fieldnames)):
            # Note: FID and shape can't be edited.  Hence start from column 2
            #print "fieldnames[" + str(i) + "] =", fieldnames[i] + ":"
            #print "  existing value =", rowCur.getValue(fieldnames[i])
            #print "  new value      =", rowUpd.getValue(fieldnames[i])
            rowCur.setValue(fieldnames[i], rowUpd.getValue(fieldnames[i]))
            rowsCur.updateRow(rowCur)
        rowCur = rowsCur.next()
        rowUpd = rowsUpd.next()
        j = j + 1
        if j%1000 == 0:
            msgOut("Table lines processed: " + str(j))
    del rowsCur
```

```
    del rowsUpd
    arcpy.Delete_management(inMemorySheet)

    arcpy.env.workspace = MLRDIR
    arcpy.env.scratchWorkspace = MLRDIR
    msgOut("End of shapes update processing")
    manager.notbusy()

def rastersUpdate():
    if DEBUG:
        print "Entering rastersUpdate"
    inShapeName = rasSrc.get()      # i.e., the shapefile that holds the field(s) from which to generate raster(s)
    print "Source shapefile for raster generation =", inShapeName
    manager = BusyManager(root)
    manager.busy()
    msgOut("Beginning rasters update processing")
    arcpy.env.workspace = shapesDir
    arcpy.env.scratchWorkspace = shapesDir
    currentShapeVer = getCurrentVerName(arcpy.ListFiles("*.shp"), inShapeName + 'v')
    srcShape = shapesDir + currentShapeVer
    arcpy.env.workspace = rastersDir
    arcpy.env.scratchWorkspace = rastersDir

    # Select rasters for the pollutant source (pSrc):
    # p = pathogen, n = nutrient, c = pesticide, s = sediment, t = transport
    pSrcRasters = []
    pS = pSrc.get()
    for r in shapeVars[inShapeName]:  # inShapeName is the key into the shapeVars dictionary built from layersList.txt
        if r[0] == pS:              # r is the raster name derived from the field name in the shapeVars entry
            pSrcRasters.append(r)
    fullRastersList = arcpy.ListRasters('*')
    for nR in pSrcRasters:
        msgOut("Processing raster: " + nR)
        outRaster = rastersDir + genNewVerName(fullRastersList, nR + 'v')
        arcpy.FeatureToRaster_conversion(srcShape, nR, outRaster)  # add new raster as next version of existing one

    arcpy.env.workspace = MLRDIR
    arcpy.env.scratchWorkspace = MLRDIR
    msgOut("End of rasters update processing")
    manager.notbusy()

def risksUpdate():      # Needs ArcGIS v10.6.1
    if DEBUG:
        print "Entering risksUpdate"

    def coeff(lyr):
        if DEBUG:
            print "Entering coeff with lyr =", lyr
        # Note: Only dealing with the +/- operator.  Assuming all coefficients will be multiplied
        if rcDict[lyr][1] == '+':
            c = float(rcDict[lyr][0])
        else:
            c = -float(rcDict[lyr][0])
        return c

    manager = BusyManager(root)
    manager.busy()
    msgOut("Beginning risks update processing")
    inRasDir = rastersDir
    outRasDir = risksDir
    rcDict = readModel()

    # Apply raster calculations
    msgOut("Beginning raster processing.")
    msgOut("It will take a few minutes\nto process the following rasters:")
    arcpy.env.workspace = rastersDir
    arcpy.env.scratchWorkspace = rastersDir
    pollutantSrc = pSrc.get()
    fullRastersList = arcpy.ListRasters('*')
    pollutantSrcSubset = [r for r in fullRastersList if r[0] == pollutantSrc or (r[0] == 't')]
```

```
    print "pollutantSrcSubset =", pollutantSrcSubset

    # Build lists of the current transport and source raster names
    # Set aside saw and flows names
    tRasters = []
    sRasters = []
    for rK in rcDict.keys():
        print "\nrK =", rK
        if 'tsaw' in rK:
            tsawName = getCurrentVerName(pollutantSrcSubset, rK)
        elif 'tflows' in rK:
            tflowsName = getCurrentVerName(pollutantSrcSubset, rK)
        elif rK[0] == 't':
            tRasters.append(getCurrentVerName(pollutantSrcSubset, rK))
        else:
            sRasters.append(getCurrentVerName(pollutantSrcSubset, rK))
    print "tsawName =", tsawName                # Put messages out here
    print "tflowsName =", tflowsName
    print "tRasters =", tRasters
    print "sRasters =", sRasters
    msgOut(tflowsName)
    for t in tRasters:
        msgOut(t)
    for s in sRasters:
        msgOut(s)

    tName = tRasters[0][:-(VERLEN + 1)]
    print"tName =", tName
    sName = sRasters[0][:-(VERLEN + 1)]
    print"sName =", sName
    tSumRasters = coeff(tName) * arcpy.Raster(rastersDir + tRasters[0])
    sSumRasters = coeff(sName) * arcpy.Raster(rastersDir + sRasters[0])
    for t in tRasters[1:]:
        print "t =", t          # message out here
        tName = t[:-(VERLEN + 1)]
        tSumRasters = tSumRasters + coeff(tName) * arcpy.Raster(rastersDir + t)
    for s in sRasters[1:]:
        sName = s[:-(VERLEN + 1)]
        print "sName =", sName
        sSumRasters = sSumRasters + coeff(sName) * arcpy.Raster(rastersDir + s)
#   sumRasters = tSumRasters + sSumRasters
#   sumRasters = Con(sSumRasters > 0, sSumRasters + tSumRasters, 0)   # No pollutants = no risk

    sumRasters = Con(sSumRasters > 0, sSumRasters, 0) \
            * tSumRasters * coeff('tflowsw') * arcpy.Raster(rastersDir + tflowsName)  # no flows ...
    # Remove non-drinking water catchments and save risk raster
    # Actually we don't have to any more because everything is tied to watersheds
    # Note: the first attempt produced a limited colour range whereas the Con approach was brighter
    # sumRaster = sumRaster * arcpy.Raster(rastersDir + tsawCurrent) # original SAW approach
    # sumRaster = arcpy.sa.Con(arcpy.Raster(rastersDir + tsawCurrent), sumRaster, 0)

    # Generate name for next version of risk raster and save
    arcpy.env.workspace = risksDir
    arcpy.env.scratchWorkspace = risksDir
    pS = pSrc.get()
    pSrcRiskRasters = arcpy.ListRasters(pS + "*")
    nextpSrcRR = genNewVerName(pSrcRiskRasters, pS + "riskv")
    sumRasters.save(outRasDir + nextpSrcRR)
    tmpRas = arcpy.Raster(nextpSrcRR)
    print "New risk raster min, mean and max =", tmpRas.minimum, tmpRas.mean, tmpRas.maximum
    print "rcDict is now", rcDict

    arcpy.env.workspace = MLRDIR
    arcpy.env.scratchWorkspace = MLRDIR
    msgOut(nextpSrcRR + " saved\nEnd of risks update processing")
    manager.notbusy()

def generateMXD(inRas, outMXD, tmpltFile, tmpltLayer, titleStr, symbology):
    if DEBUG:
        print "Entering generateMXD with inRas, outMXD =", inRas, outMXD
```

```
        print "Temporary .mxd is", mapTemplatesDir + tmpltFile

    # https://gis.stackexchange.com/questions/82335/
    #         how-to-programmatically-change-raster-symbology-after-reclassification
    # Viewed 4/7/2018
    templateSymbology = mapTemplatesDir + symbology
    mxd = arcpy.mapping.MapDocument(mapTemplatesDir + tmpltFile)
    df = arcpy.mapping.ListDataFrames(mxd)[0]
    addLayer = arcpy.mapping.Layer(inRas)

    # Build table of raster layers and their weightings for risk surface
    inRasWeights = ("prisk" in inRas) or ("nrisk" in inRas)
    if inRasWeights:
        rcDict = {}
        rcDict = readModel()
        rasterWeights = "Weight  Layer\n------  -----\n"
        for key, val in sorted(rcDict.items()):
            if key != "tsaww":
                c = "{0:.3f}".format(float(val[0]))
                rasterWeights = rasterWeights + val[1] + c + "  " + key + "\n"
        rasterWeights = rasterWeights + "   \nRef: " + refNum.get()

    # Place new layer underneath catchment polygons
    arcpy.mapping.AddLayer(df, addLayer, "BOTTOM")
    reclass_raster_lyr = arcpy.mapping.ListLayers(mxd)[tmpltLayer]

    # Apply symbology from .lyr template file and save map
    arcpy.ApplySymbologyFromLayer_management(reclass_raster_lyr, templateSymbology)
    arcpy.RefreshActiveView()
    for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT_ELEMENT"):
        if elm.text == "Title":
            elm.text = titleStr
            elm.elementPositionX = 2.5
            elm.elementPositionY = 26.0
        elif elm.text == "Author:":
            elm.text = ("Author: " + "IDM")
            #elm.text = ("Author: " + os.getenv('username'))
        elif elm.name == "rasterweights":
            if inRasWeights:
                elm.text = rasterWeights
    legend = arcpy.mapping.ListLayoutElements(mxd, "LEGEND_ELEMENT", "Legend")[0]
    legend.elementPositionX = 2.5
    legend.elementPositionY = 10.0

    mxd.saveACopy(outMXD +".mxd")
    del mxd, addLayer

def reportHeading(catchment, pS, rptTypeVer):
    if pS == 'p':
        pSrcText = "Pathogen"
    elif pS == 'n':
        pSrcText = "Nutrient (P)"
    elif pS == 'c':
        pSrcText = "Pesticide"
    elif pS == 's':
        pSrcText = "Sediment"
    else:
        pSrcText = "Transport"
    if "Mean" in rptTypeVer:
        scaling = "\n(scaled view)"
    else:
        scaling = "\n(normailsed view)"
    return "Mount Lofty Ranges Watershed " + catchment + "\n" + pSrcText + rptTypeVer + scaling

def reportsUpdate():
    if DEBUG:
        print "Entering reportsUpdate"

    def generatePDF(outDir, cType):
        # Read back the map with new symbology
```

```
    # http://desktop.arcgis.com/en/arcmap/10.3/analyze/arcpy-mapping/textelement-class.htm
    # Viewed 4/7/2018
    if DEBUG:
        print "Entering generatePDF with mxd name =", outDir + cType + ".mxd"
    mxd = arcpy.mapping.MapDocument(outDir + cType + ".mxd")
    arcpy.mapping.ExportToPDF(mxd, outDir + cType + ".pdf")
    del mxd

def normaliseRaster(inRas, outDir, outRas):
    # convert reals to integers in range 0 - 100
    if DEBUG:
        print "Entering normaliseRaster with inRas =", inRas
        inRasTmp = arcpy.Raster(inRas)
        print "inRasTmp minmum and maximum =", inRasTmp.minimum, inRasTmp.maximum
    nmlRaster = (arcpy.Raster(inRas) - arcpy.Raster(inRas).minimum) /    \
            (arcpy.Raster(inRas).maximum - arcpy.Raster(inRas).minimum) * 100 + 0.5
    arcpy.Int_3d(nmlRaster, outDir + outRas)

def scaleRaster(inRas, outDir, outRas):
    # convert reals to integers in range 0 - 100
    if DEBUG:
        print "Entering scaleRaster with inRas =", inRas
        inRasTmp = arcpy.Raster(inRas)
        inRasMin = inRasTmp.minimum
        inRasMean = inRasTmp.mean
        inRasMax = inRasTmp.maximum
        print "inRasTmp minmum, mean and maximum =", inRasMin, inRasMean, inRasMax
    scaledRaster = inRasTmp / inRasMax * 100 + 0.5
    arcpy.Int_3d(scaledRaster, outDir + outRas)

def genZonalStats(zData, zField, fN, zStat, template, tmpltLayer, titleStr, symbology):
    if DEBUG:
        print "Entering genZonalStats witn fN =", fN
    msgOut("Beginning " + fN + " Zonal Stats processing")
    outZonalStatistics = ZonalStatistics(zData, zField, inValueRaster, zStat)
    outZonalStatistics.save(reportsDir + fN)

    # Generate scaled DWC risk surface, remove non-DWC regions and save PDF
#     normaliseRaster(fN, reportsDir, fN + "n")
    scaleRaster(fN, reportsDir, fN + "n")
    fNExclNon = arcpy.sa.Con(arcpy.Raster(rastersDir + "tsawwv000"), arcpy.Raster(reportsDir + fN + "n"),)
    fNExclNon.save(reportsDir + fN)

    if (zField == "System") and (zStat == "MEAN"):       # Add MEAN values to label of each WTP system
        print "Entering ZonalStatisticsAsTable"
        outZStable = ZonalStatisticsAsTable(srcShapesDir + "DWC_MLR.shp", "System", reportsDir + fN,
                                    reportsDir + "dwcZSMeanTable.dbf", "DATA", "MEAN")
        outZStableUnscaled = ZonalStatisticsAsTable(srcShapesDir + "DWC_MLR.shp", "System", inValueRaster,
                                    reportsDir + "dwcZSMeanTableUnscaled.dbf", "DATA", "MEAN")

        # Build dictionary of Zonal Stats mean values with WTP name as the key
        # The value is formatted as a string for incorporation into layout labels
        zonalStatsDBF = r"dwcZSMeanTable.dbf"
        zonalStatsFields = ['System', 'ZONE_CODE', 'COUNT', 'AREA', 'MEAN']
#         zonalStatsDict = {r[0]:" (" + str(int(r[4])) + ")" for r in arcpy.da.SearchCursor(zonalStatsDBF, zonalStatsFields)}
        zonalStatsDict = {r[0]:"/" + str(int(r[4])) for r in arcpy.da.SearchCursor(zonalStatsDBF, zonalStatsFields)}

        # Create version of DWC_MLR shapefile annotated with Zonal Statistics table MEAN values
        inShapeName = "DWC_MLR.shp"
        outShapeName = "DWClbl.shp"
        arcpy.FeatureClassToFeatureClass_conversion(srcShapesDir + inShapeName, reportsDir, outShapeName)
        outFieldNames = [f.name for f in arcpy.ListFields(outShapeName)]
        with arcpy.da.UpdateCursor(reportsDir + outShapeName, outFieldNames) as cursor:
            for row in cursor:
                if row[2] != "Non-drinking water":
                    meanValue = zonalStatsDict[row[2]]
                    row[2] = row[2] + meanValue
                cursor.updateRow(row)
        del cursor
```

```
            # Generate labelled mxd template
            mxd = arcpy.mapping.MapDocument(mapTemplatesDir + "dwcTemplate.mxd")
            df = arcpy.mapping.ListDataFrames(mxd)[0]

            # Remove the existing layer in the template but keep the layout
            for lyr in arcpy.mapping.ListLayers(mxd, "", df):
                if lyr.name == "DWC_MLR":
                    arcpy.mapping.RemoveLayer(df, lyr)
            newDWCLayer = arcpy.mapping.Layer(reportsDir + "DWClbl.shp")
            arcpy.mapping.AddLayer(df, newDWCLayer)
            reclass_raster_lyr = arcpy.mapping.ListLayers(mxd)[0]
            reclass_raster_lyr.showLabels = True
            arcpy.ApplySymbologyFromLayer_management(reclass_raster_lyr, "dwcMeanLabelledTemplate.lyr")
            arcpy.RefreshActiveView()
            mxd.saveACopy(mapTemplatesDir + "dwcLabelledTemplate.mxd")
            del mxd
            generateMXD(reportsDir + fN, reportsDir + fN, "dwcLabelledTemplate.mxd", tmpltLayer, titleStr, symbology)
        else:
            generateMXD(reportsDir + fN, reportsDir + fN, template, tmpltLayer, titleStr, symbology)
        generatePDF(reportsDir, fN)
        arcpy.Delete_management(fN + "n")
#       arcpy.Delete_management(fN + ".mxd")
#       arcpy.Delete_management(fN)

    manager = BusyManager(root)
    manager.busy()
    msgOut("Beginning standard reports processing")
    inRasDir = MLRDIR + "Risks\\"
#   reportsDir = MLRDIR + "Reports\\"

    # Get latest risk raster for the pollutant source
    # Note: ListRasters can only list rasters in the current directory
    #       so change workspace and scratchworkspace for this step, then restore them
    arcpy.env.workspace = inRasDir
    arcpy.env.scratchWorkspace = inRasDir
    pS = pSrc.get()
    pSrcRiskRasters = arcpy.ListRasters(pS + "*")
    firstRR = pSrcRiskRasters[0]
    currentRiskRaster = getCurrentVerName(pSrcRiskRasters, firstRR[:-VERLEN])
    inValueRaster = inRasDir + currentRiskRaster
    arcpy.env.workspace = reportsDir
    arcpy.env.scratchWorkspace = reportsDir

    # Generate normalised DWC risk surface, remove non-DWC regions and save PDF
    msgOut("Generating risk surface MXD and PDF files")
    normaliseRaster(inValueRaster, reportsDir, currentRiskRaster + "tmp")
    rrExclNon = arcpy.sa.Con(arcpy.Raster(rastersDir + "tsawwv000"), arcpy.Raster(reportsDir + currentRiskRaster +
"tmp"))
#   rrExclNon = arcpy.sa.Con(arcpy.Raster(rastersDir + "gsawv000"), arcpy.Raster(risksDir + currentRiskRaster))
    rrExclNon.save(reportsDir + currentRiskRaster)
    tmpltLayer = 2
    titleStr = reportHeading("Catchments", pS, " Risk Assessment V" + currentRiskRaster[-3:])
    generateMXD(reportsDir + currentRiskRaster, reportsDir + currentRiskRaster,
                "risksTemplate.mxd", tmpltLayer, titleStr, "rasterSymbologyTemplate.lyr")
    generatePDF(reportsDir, currentRiskRaster)
    msgOut("Saved risk surface MXD and PDF files")

    # Uncomment the following to get minimum and maximum values for the risk raster
#   inRasTmp = arcpy.Raster(currentRiskRaster + "tmp")
#   print "Normalised risk raster minmum and maximum =", inRasTmp.minimum, inRasTmp.maximum
    arcpy.Delete_management(currentRiskRaster + "tmp")

    # Generate normalised DWC Zonal Stats Mean view for System variable
    msgOut("Generating Zonal Statistics MEAN output")
    dwcPDFs = arcpy.ListFiles(pS +"dwc*.pdf")
    dwcNextMean = genNewVerName(dwcPDFs, pS + "dwcmeanv")
    inZoneData = srcShapesDir + "DWC_MLR.shp"
    zoneField = "System"
    stat = "MEAN"
    tmpltLayer = 1
```

```
    #   **** Build titleStr as above (as a separate function? ***
    titleStr = reportHeading("Catchments", pS, " Mean Zonal Statistics V" + dwcNextMean[-3:])
    genZonalStats(inZoneData, zoneField, dwcNextMean, stat, "dwcTemplate.mxd", tmpltLayer, titleStr,
                "rasterSymbologyTemplate.lyr")
#    outZStable = ZonalStatisticsAsTable(inZoneData, zoneField, inValueRaster, reportsDir + "dwcZSMeanTable.dbf",
"DATA", "MEAN")

    # Generate normalised sub-DWC Zonal Stats Mean view for SUB_NAME variable
    subPDFs = arcpy.ListFiles(pS +"sub*.pdf")
    subNextMean = genNewVerName(subPDFs, pS + "submeanv")
    inZoneData = srcShapesDir + "MLR_Subcatchment.shp"
    zoneField = "SUB_NAME"
    stat = "Mean"
    tmpltLayer = 2
    # Check title string function call below
    titleStr = "Mount Lofty Ranges Watershed Sub-catchments\nPathogen Mean Zonal Statistics V" + subNextMean[-3:] +
"\n(normalised view)"
#    titleStr = reportHeading("Sub-catchments", pS, " Mean Zonal Statistics V" + subNextMean[-3:])
##    # Generate normalised sub-DWC Zonal Stats Maximum view for SUB_NAME variable
##    subNextMax = subNextMean[:4] + "max" + subNextMean[-4:]
##    inZoneData = srcShapesDir + "MLR_Subcatchment.shp"
##    zoneField = "SUB_NAME"
##    stat = "Maximum"
##    tmpltLayer = 2
    # Check title string function call below
##    titleStr = "Mount Lofty Ranges Watershed Sub-catchments\nPathogen Maximum Zonal Statistics V" + subNextMax[-
3:] + "\n(normalised view)"
#    titleStr = reportHeading("Sub-catchments", pS, " Maximum Zonal Statistics V" + subNextMax[-3:])
##    genZonalStats(inZoneData, zoneField, subNextMax, stat, "dwsubcTemplate.mxd", tmpltLayer, titleStr
##                "rasterSymbologyTemplate.lyr")

    arcpy.env.workspace = MLRDIR
    arcpy.env.scratchWorkspace = MLRDIR
    msgOut("End of standard reports processing")
    manager.notbusy()

def createUniqueValuesMap(currentRR):
    if DEBUG:
        print "Entering createUniqueValuesMap"
    # Create a map file with values reclassified to unique for histogram statistics generation
    histRas = arcpy.Raster(reportsDir + currentRR)
    histRas.save(reportsDir + "histRas")
    tmpltLayer = 2
    titleStr = "Unique values raster\nfor histogram statistics"
    generateMXD(reportsDir + "histRas", reportsDir + "histRas",
                "risksTemplate.mxd", tmpltLayer, titleStr, "uniqueValuesTemplate.lyr")

def selectDwcShape(ctchmntType, ctchmntChoice, whereVar, dwcChoiceUpper):
    if DEBUG:
        print "Entering selectDwcShape"
    srcShapeFile = srcShapesDir + ctchmntType
    arcpy.MakeFeatureLayer_management(srcShapeFile, "lyr")
    whereClause = whereVar + " = '" + dwcChoiceUpper + "'"
    arcpy.SelectLayerByAttribute_management("lyr", "NEW_SELECTION", whereClause)
    arcpy.FeatureClassToFeatureClass_conversion("lyr", reportsDir, "selectShape")

def detailedStatistics():
    if DEBUG:
        print "Entering detailedStatistics"
    msgOut("Unused for now")

def detailedComparison():
    if DEBUG:
        print "Entering detailedComparison"
    msgOut("rasShares code goes here when a bug in Python/PyScripter is resolved")

def detailedAnalysis():          # Specific to pathogens for now
    if DEBUG:
        print "Entering detailedAnalysis"
```

```
manager = BusyManager(root)
manager.busy()
msgOut("Beginning of detailed analysis processing")
arcpy.env.workspace = reportsDir
arcpy.env.scratchWorkspace = reportsDir

# Need to build in error checking to force sub-DWC choice before proceeding
dwcChoice = dwcVar.get()      # From dwcMenuButton in initGUI below
print "dwcChoice =", dwcChoice
dwcChoiceUpper = dwcChoice.upper()

# Create copy of current risk raster with intergers in range 0 - 100 rather than reals
pS = pSrc.get()
pSrcRiskRasters = arcpy.ListRasters(pS + "*")
firstRR = pSrcRiskRasters[0]
currentRiskRaster = getCurrentVerName(pSrcRiskRasters, firstRR[:-VERLEN])
createUniqueValuesMap(currentRiskRaster)

msgOut("Generating high risk regions for sub-DWC " + dwcChoice)
selectDwcShape("MLR_Subcatchment.shp", dwcChoiceUpper, "SUB_NAME", dwcChoiceUpper)
inMask = "selectShape.shp"

# Create raster of the top 10% of risk values for this sub-DWC
inRisk = risksDir + "priskv000"          # *** in development.  Version number is hard coded ***
cutRisk = ExtractByMask(inRisk, inMask)
cutRisk.save(reportsDir + "cutRisk")
cutRisk = arcpy.Raster("cutRisk")
cutRiskTop = Con(cutRisk > 0.6 * cutRisk.maximum, 1, 0)   # Get dwc high risk values
cutRiskTop.save(reportsDir + "cutRiskTop")

# Create raster of fencing for this sub-DWC
inFence = "tfencev000"                  # *** in development. pfence must be converted to tfence ***
cutFence = ExtractByMask(inFence, inMask)       # Also, get it from rasters
cutFence.save(reportsDir + "cutFence")

# Create streams shapefile for this sub-DWC
inStreams = srcShapesDir + "MLR_Streams.shp"
arcpy.Clip_analysis(inStreams, inMask, reportsDir + "cutStreams")

# Separate out later to generalised generateMXD above?
# https://community.esri.com/thread/204754-arcpyapplysymbologyfromlayermanagement
#        -does-not-define-symbology-within-stand-alone-script
symbology = {"cutStreams" : mapTemplatesDir + "cutStreamsTemplate.lyr",
             "selectShape" : mapTemplatesDir + "selectShapeTemplate.lyr",
             "cutFence" : mapTemplatesDir + "cutFenceTemplate.lyr",
              "cutRiskTop" : mapTemplatesDir + "cutRisktopTemplate.lyr"}
layerList = ["selectShape.shp", "cutRiskTop", "cutFence", "cutStreams.shp"]

# Add detailed sub-DWC layers to template already loaded with scale bar, north arrow and legend
mxd = arcpy.mapping.MapDocument(mapTemplatesDir + "detailedsubdwcTemplate.mxd")
df = arcpy.mapping.ListDataFrames(mxd)[0]
for lL in layerList:
    print "Adding layer " + lL + " to mxd template"
    addLayer = arcpy.mapping.Layer(lL)
    arcpy.mapping.AddLayer(df, addLayer)

# Apply symbology in template .lyr files to layers
for l in arcpy.mapping.ListLayers(mxd):
    print "Apply symbology for layer name:", l.name
    arcpy.ApplySymbologyFromLayer_management(l, symbology[l.name])
arcpy.RefreshTOC()
arcpy.RefreshActiveView()
for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT_ELEMENT"):
    print "elm.text =", elm.text
    if elm.text == "Title":
        elm.text = dwcChoice + " High Risk Regions\nRelative to Streams and Fencing"
mxd.saveACopy(reportsDir + "subDWC.mxd")

# Read back mxd with new symbology applied and generate layout PDF
mxd = arcpy.mapping.MapDocument(reportsDir + "subDWC.mxd")
```

```
    arcpy.mapping.ExportToPDF(mxd, reportsDir + dwcChoice + ".pdf")

    del mxd, addLayer
    purge(reportsDir, "histRasTbl")
#   purge(reportsDir, "selectShape")
    purge(reportsDir, "histogramStats")
    arcpy.Delete_management("histRas.mxd")
    arcpy.Delete_management("histRas")

    arcpy.env.workspace = MLRDIR
    arcpy.env.scratchWorkspace = MLRDIR
    msgOut("End of detailed analysis processing")
    manager.notbusy()

def quit():
    if DEBUG:
        print "Entering quit"
    root.destroy()

def readDWCChoices():
    if DEBUG:
        print "Entering readDWCChoices"
#   inSpreadSheet = MLRDIR + "histogramChoices.xlsx"
    pS = pSrc.get()
    menuDict = {}
    if (pS == 'p') or (pS == 'g'):
        sheetName = "pathogens"
    elif pS == 'n':
        sheetName = "nutrients"
    elif pS == 'c':
        sheetName = "pesticides"
    else:
        sheetName = "sediments"
#   inMemorySheet = "in_memory" + "\\" + "histogram"

    # Read menu and submenu names from histogramChoices text file
    menus = []
    with open(MLRDIR + '/DWC.txt','r') as input_file:
        for line in input_file:
            menus.append([subMenu.strip('\n') for subMenu in line.split(',')])
    input_file.close()
    return menus

def initGUI(root):
    if DEBUG:
        print "Entering initGUI"
    root.geometry("470x760")
    root.title("MLR Risk Management Decision Support")

    # Create pollutant source radiobutton widgets for the top frame
    pollutantSrcLabel = Label(frame0, font = 'None 11 underline', text = "\nPollutant source", fg = "blue")
    pathogen = Radiobutton(frame0, text = "Pathogen", padx = 60, variable = pSrc, value = "p")
    nutrient = Radiobutton(frame0, text = "Nutrient", padx = 60, variable = pSrc, value = "n")
    pesticide = Radiobutton(frame0, text = "Pesticide", padx = 60, variable = pSrc, value = "c")
    sediment = Radiobutton(frame0, text = "Sediment", padx = 60, variable = pSrc, value = "s")
    transport = Radiobutton(frame0, text = "Transport", padx = 60, variable = pSrc, value = "t")

    # Lay out radiobutton widgets in the top frame
    pollutantSrcLabel.grid(row=0, column = 0, pady = 10)
    pathogen.grid(row = 1, column = 0, sticky = 'w')
    nutrient.grid(row = 2, column = 0, sticky = 'w')
    pesticide.grid(row = 3, column = 0, sticky = 'w')
    sediment.grid(row = 4, column = 0, sticky = 'w')
    transport.grid(row = 5, column = 0, sticky = 'w')
    if DEBUG:
        print "pSrc = ", pSrc.get()

    # Create and lay out frame1 data source dropdown widget and update buttons
    dataSrcLbl = Label(frame1, font = 'None 11 underline', text = "Data Management", fg = "blue")
    dataSrcLbl.grid(row = 0, column = 0)          #, padx = (32,)
```

```
dataSrcLb2 = Label(frame1,
      font = (None, 10),
      text = "Select shapefile for Shapes or Rasters updates", fg = "blue")
dataSrcLb2.grid(row = 1, column = 0)          #, padx = (32,)
dataSrc = OptionMenu(frame1, rasSrc, *OPTIONS)
dataSrc.config(width=21)
dataSrc.grid(row = 2, column = 0, pady = 10)                #, sticky = "w"

stdRptsLbl = Label(frame2, font = 'None 11 underline', text = "\nStandard Reports", fg = "blue")
stdRptsLbl.grid(row = 0, column = 0, padx = (32,))
stdRptsLb2 = Label(frame2, font = 'None 10', text = "Risk surface and Zonal Statistics", fg = "blue")
stdRptsLb2.grid(row = 1, column = 0, padx = (32,))

# Frame 3 catchment choice and statistics analysis
dwcMenuButton = Menubutton(frame3, textvariable = dwcVar, indicatoron = True, borderwidth = 2, relief=RAISED)
dwcMenuButton.config(width = 29)
dwcMenuButton.grid(row = 2, column = 0, padx = (0, 0), pady = 0)     #, sticky = 'w'
dwcMenu = Menu(dwcMenuButton, tearoff = False)
dwcMenuButton.configure(menu = dwcMenu)
for item in readDWCChoices():
    menu = Menu(dwcMenu, tearoff=False)
    dwcMenu.add_cascade(label = item[0], menu = menu)
    i = 0
    for value in item[1:]:
        if value == '*':
            i += 1
            menu.add_separator()
        else:
            if i == 0:
                dwcDict[value] = "dwc"
                lbl = value + " (DWC)"
            elif i == 2:
                dwcDict[value] = "wtp"
                lbl = value + " (WTP)"
            else:
                dwcDict[value] = "sub"
                lbl = value
            menu.add_radiobutton(value = value, label = lbl, variable = dwcVar)
analyseLbl = Label(frame3, font = 'None 11 underline', text = "\nAnalysis", fg = "blue")
analyseLbl.grid(row = 0, column = 0, padx = (32,))
analyseLb2 = Label(frame3,
      font = (None, 10),
      text = "Select relevant catchment type before\n" + \
           "detailed statistics or analysis (sub-DWC only) reports\n", fg = "blue")
analyseLb2.grid(row = 1, column = 0, padx = (32,))

# Create and lay out frame1-3 update and reports buttons
buttonShapes = Button(frame1, text = "Shapes", command = shapesUpdate)
buttonShapes.config(width = 8)
buttonShapes.grid(row = 3, column = 0, padx = (0, 102), pady = 0)          #, sticky = 'w'
buttonRasters = Button(frame1, text = "Rasters", command = rastersUpdate)
buttonRasters.config(width = 8)
buttonRasters.grid(row = 3, column = 0, padx = (100, 0), pady = 0)         #, sticky = 'e'
buttonRisks = Button(frame2, text = "Risks", command = risksUpdate)
buttonRisks.config(width = 8)
buttonRisks.grid(row = 3, column = 0, padx = (0, 100), pady = 10)         #, padx = (15, 10), sticky = 'w'
buttonReports = Button(frame2, text = "Reports", command = reportsUpdate)
buttonReports.config(width = 8)
buttonReports.grid(row = 3, column = 0, padx = (100, 0), pady = 10)         #, padx = (15, 5), sticky = 'e'
buttonStatistics = Button(frame3, text = "Statistics", command = detailedStatistics)
buttonStatistics.config(width = 8)
buttonStatistics.grid(row = 3, column = 0, padx = (0,150), pady = 10)         #, padx = (15, 5)
buttonCompare = Button(frame3, text = "Compare", command = detailedComparison)
buttonCompare.config(width = 8)
buttonCompare.grid(row = 3, column = 0, padx = (150, 150), pady = 10)         #, padx = (15, 5)
buttonAnalyse = Button(frame3, text = "Analyse", command = detailedAnalysis)
buttonAnalyse.config(width = 8)
buttonAnalyse.grid(row = 3, column = 0, padx = (150, 0), pady = 10)         #, padx = (15, 5)

# Lay out frame5 message window (created above as a global)
```

```
    msgWin.grid(row = 0, column = 0, pady = (10,10))

    # Create and lay out frame4 Quit button widget
    buttonQuit = Button(frame4, text = "Quit", command = quit)
    buttonQuit.config(width = 9)
    buttonQuit.grid(row = 2, column = 0, pady = (10, 10))

def main():
    initGUI(root)
    root.mainloop()

if __name__ == '__main__':
    main()
```

Appendix F: Python script for total risk contribution of each variable

```
# Needed 64-bit environment to avoid arcpy error
import arcpy
import numpy as np
import matplotlib.pyplot as plt
import time as time
from matplotlib import style
import xlrd

MLRDIR = r'<User directory>/Project/'
RASDIR = MLRDIR + 'Rasters/'
SHRDIR = MLRDIR + 'Utility/Share'

style.use('seaborn-whitegrid')

rasDict = {}

def readModel():

    rDict = {}

    # Get first sheet of spreadsheet - this is the most recent model
    workbook = xlrd.open_workbook('pmodel.xlsx', on_demand=True)
    sheetNames = workbook.sheet_names()
    sheetName = sheetNames[0]
    print('sheetName =', sheetName)
    workbook.release_resources()
    inMemorySheet = "in_memory" + "\\" + "model"

    # Read model spreadsheet into in-memory table
    arcpy.Delete_management(inMemorySheet)
    arcpy.ExcelToTable_conversion('pmodel.xlsx', inMemorySheet, sheetName)

    # Build dictionary of model layer name (the key) and corresponding coefficients and operators
    with arcpy.da.SearchCursor(inMemorySheet, '*') as cursor:
        for row in cursor:
            key = row[1]
            rDict[key] = [row[2], row[3]]
            print('rDict[' + key + '] =', rDict[row[1]])
    del cursor
    arcpy.Delete_management(inMemorySheet)
    del rDict['Reference']
    return rDict

arcpy.env.workspace = RASDIR
arcpy.env.scratchWorkspace = RASDIR
fullRastersList = arcpy.ListRasters('*')
rasSubset = [r for r in fullRastersList if (r[0] == 'p') or (r[0] == 't')]
print(rasSubset)
sh = np.zeros(len(rasSubset), dtype = float)
arcpy.env.workspace = SHRDIR
arcpy.env.scratchWorkspace = SHRDIR

rasDict = readModel()
print('rasDict =', rasDict)
sh = []
i = 0
for r in rasSubset:
    start = time.time()
    inRas = arcpy.Raster(RASDIR + r)
    rNameRoot = inRas.name[:-4]
    print(rNameRoot, rasDict[rNameRoot][0], rasDict[rNameRoot][1])
    if rasDict[rNameRoot][1] == '+':
        weight = float(rasDict[rNameRoot][0])
    elif rasDict[rNameRoot][1] == '*':
        weight = 0.0
    else:
        weight = -float(rasDict[rNameRoot][0])
```

```
    inRasArray = arcpy.RasterToNumPyArray(inRas, nodata_to_value=0)
    weightedArray = inRasArray * weight
    sum = np.sum(weightedArray)
    sh.append(sum/10e6)
    i += 1
    end = time.time()
    print(inRas.name + ' sum =', sum, 'at time', end - start)
print(sh)
plt.rcParams.update({'font.size': 15})
plt.figure(figsize = (10,10.8))
plt.xlabel('Variable rasters')
plt.ylabel('Total risk/1,000,000')
labels = sorted(rasDict.keys())
x = np.arange(i)
plt.bar(x, height = sh, align = 'center')
plt.xticks(x,labels, rotation = 'vertical')
plt.margins(x = 0)
plt.show()
```

Appendix G: Python scripts for flow rate replacement of slope

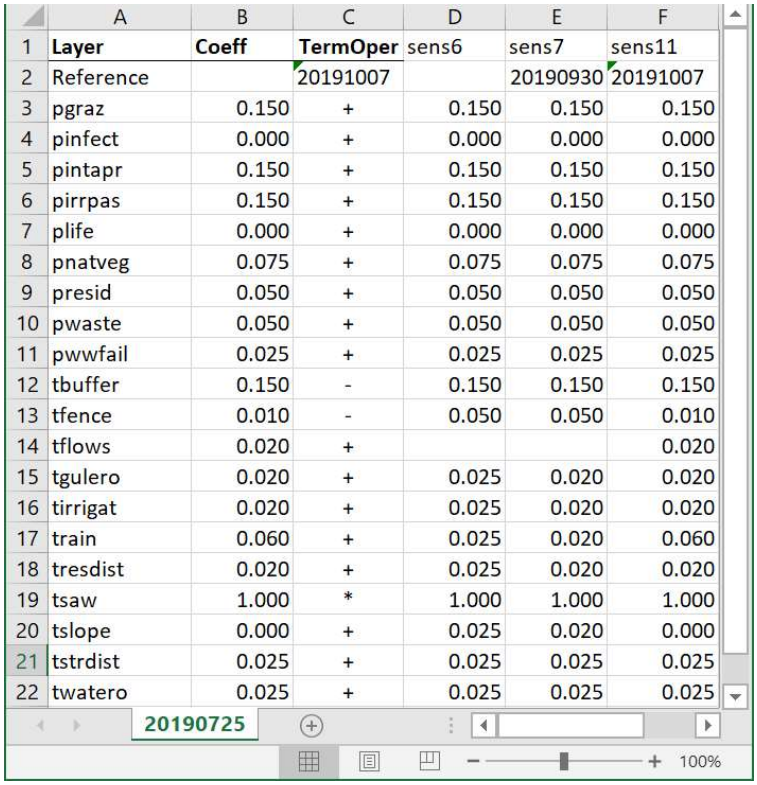
```
"""

catchmentFlowsAccumulation

Use catchment shapes to clip out catchment DEMS and fill any sinks.
Calculate flow directions.
Calculate flow accumulation prior to determining the position of the
watershed outlet (pour point)
"""

import arcpy
from arcpy.sa import *

FLOWSDIR = <flows directory path>

arcpy.env.workspace = FLOWSDIR
arcpy.env.scratchWorkspace = FLOWSDIR
arcpy.env.overwriteOutput = True
arcpy.env.qualifiedFieldNames = False
arcpy.CheckOutExtension("Spatial")

mlrDEM = FLOWSDIR + "mlrdem"
mlrCatchments = FLOWSDIR + "DWC_MLR.shp"
fieldNames = [f.name for f in arcpy.ListFields(mlrCatchments) if f.type != 'OID']
print "Catchment field names =", fieldNames

# https://gis.stackexchange.com/questions/6775/using-integer-variable-in-where-clause-of-arcpy-script
# https://learn.arcgis.com/en/projects/predict-floods-with-unit-hydrographs/
# Accessed 20190731
# https://gis.stackexchange.com/questions/27457/including-variable-in-where-clause-of-arcpy-select-analysis
# Accessed 20190820
arcpy.MakeFeatureLayer_management(mlrCatchments, "lyr")
mlrCatchmentRows = arcpy.SearchCursor(mlrCatchments)
for row in mlrCatchmentRows:
    wtpVal = row.getValue("System")
    if wtpVal != 'Non-drinking water':
        print "System =", wtpVal
        cName = wtpVal.split()[0].lower() # need raster name suitable for each WTP System
        print "cName =", cName
        arcpy.SelectLayerByAttribute_management("lyr", "NEW_SELECTION", ' "System"= ' + "'%s'" % wtpVal)
        arcpy.FeatureClassToFeatureClass_conversion("lyr", FLOWSDIR, cName)
        arcpy.Clip_management(mlrDEM, "", cName + "dem", cName + ".shp", "", "ClippingGeometry",
"MAINTAIN_EXTENT")
        cDEMfilled = Fill(cName + "dem")
        cDEMfilledName = FLOWSDIR + cName + "fil"
        cDEMfilled.save(cDEMfilledName)
        cFlowDirection = FlowDirection(cDEMfilledName)
        cFlowDirectionName = FLOWSDIR + cName + "fld"
        cFlowDirection.save(cFlowDirectionName)
        cFlowAccumulation = FlowAccumulation(cFlowDirectionName)
        cFlowAccumulationName = FLOWSDIR + cName + "acc"
        cFlowAccumulation.save(cFlowAccumulationName)

'''
At this point, we need to pause for an interactive step in ArcMap:
 - measure the distance between points in the MLR_Outlets shapefile and their
   nearest flow accumulation cell
 - using this measure (rounded up to a suitable value), run the Snap Pour Point
   tool to generate a new raster with pour points on their flow accumulation
   paths.
Do this for each catchment. Then the watersheds can be determined above the
pour points.  This is done in the next script (catchmentFlowsVelocity)
which ultimately calculates the velocity field risk surface.
'''
print "Finished"
```

```
"""

catchmentFlowsVelocity

Calculate the slope area term after delineating the watershed
"""
import arcpy
from arcpy.sa import *

FLOWSDIR = <flows directory path>

arcpy.env.workspace = FLOWSDIR
arcpy.env.scratchWorkspace = FLOWSDIR
arcpy.env.overwriteOutput = True
arcpy.env.qualifiedFieldNames = False
arcpy.CheckOutExtension("Spatial")

mlrDEM = FLOWSDIR + "mlrdem"
mlrCatchments = FLOWSDIR + "DWC_MLR.shp"
fieldNames = [f.name for f in arcpy.ListFields(mlrCatchments) if f.type != 'OID']
print "Catchment field names =", fieldNames

# https://gis.stackexchange.com/questions/6775/using-integer-variable-in-where-clause-of-arcpy-script
# https://learn.arcgis.com/en/projects/predict-floods-with-unit-hydrographs/
# Accessed 20190731
# https://gis.stackexchange.com/questions/27457/including-variable-in-where-clause-of-arcpy-select-analysis
# Accessed 20190820
arcpy.MakeFeatureLayer_management(mlrCatchments, "lyr")
mlrCatchmentRows = arcpy.SearchCursor(mlrCatchments)
for row in mlrCatchmentRows:
    wtpVal = row.getValue("System")
    if wtpVal != 'Non-drinking water':
        print "System =", wtpVal
        cName = wtpVal.split()[0].lower() # need raster name suitable for each WTP System
        print "cName =", cName
        fNameStem = FLOWSDIR + cName
        cDEMfilledName = fNameStem + "fil"
        cFlowsWatershedName = fNameStem + "wsh"
        cFlowsWatershed = Watershed(fNameStem + "fld", fNameStem + "snp")
        cFlowsWatershed.save(cFlowsWatershedName)
        cFlowAccumulationName = fNameStem + "acc"
        cFlowsSlope = Slope(cDEMfilledName, "PERCENT_RISE", 1)
        cFlowsSlopeName = fNameStem + "slo"
        cFlowsSlope.save(cFlowsSlopeName)
        cFlowsSlopeArea = SquareRoot(cFlowsSlopeName) * SquareRoot(cFlowAccumulationName)
        cFlowsSlopeAreaName = fNameStem + "sla"
        cFlowsSlopeArea.save(cFlowsSlopeAreaName)
        arcpy.env.mask = cFlowsWatershedName
        slopeAreaTmp = arcpy.Raster(cFlowsSlopeAreaName)
        slopeAreaMean = slopeAreaTmp.mean
        cFlowsVelocityUnlimitedName = fNameStem + "unl"
        cFlowsVelocityUnlimited = 0.1 * (slopeAreaTmp / slopeAreaMean)
        cFlowsVelocityUnlimited.save(cFlowsVelocityUnlimitedName)
print "Finished"
```

```
"""
After determining the maximum valocity in the unlimited velocity rasters,
use this value to scale the velocity rasters to a value of 100.
"""
import arcpy
from arcpy.sa import *

FLOWSDIR = <flows directory path>

arcpy.env.workspace = FLOWSDIR
arcpy.env.scratchWorkspace = FLOWSDIR
arcpy.env.overwriteOutput = True
arcpy.env.qualifiedFieldNames = False
arcpy.CheckOutExtension("Spatial")

MAXV = 228.345   # This needs to be revised after each change in underlying data
            # Could find it automatically with an initial pass over the unl files

watersheds = ['ansteygk', 'barossa', 'happy', 'hope', 'little', 'myponga']
for w in watersheds:

   # What follows must be calculated relative to all watersheds.
   # Save the maximum for all six and use the highest value to be 100%
   # Scale everything against this
   print "Watershed =", w
   velocityUnlimitedTmp = arcpy.Raster(w + 'unl')
   velocityUnlimitedMax = MAXV
   velocityScaledRiskName = w + "vsc"
   velocityScaledRisk = velocityUnlimitedTmp / velocityUnlimitedMax * 100.0
   velocityScaledRisk.save(velocityScaledRiskName)
print "Finished"
```

Appendix H: Final Model configurations

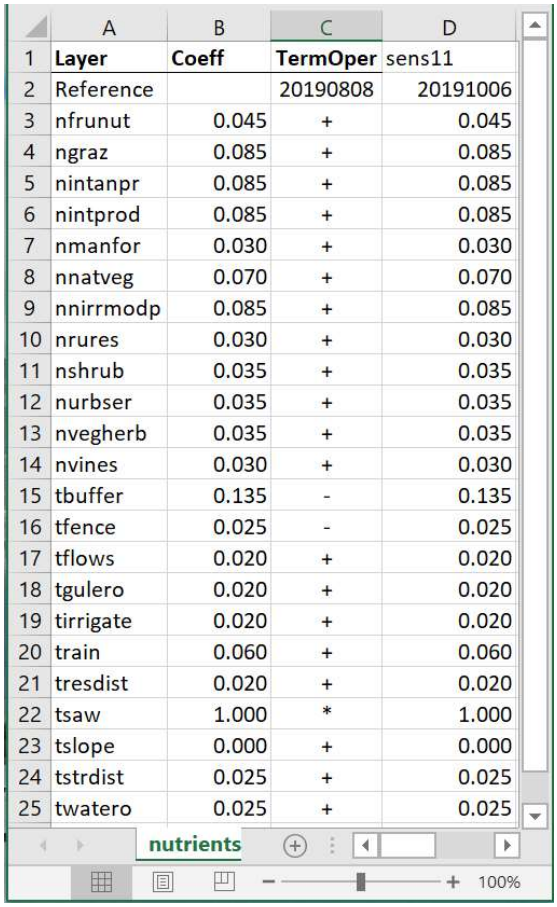
H.1: Pathogens



	A	B	C	D	E	F
1	**Layer**	**Coeff**	**TermOper**	sens6	sens7	sens11
2	Reference		20191007		20190930	20191007
3	pgraz	0.150	+	0.150	0.150	0.150
4	pinfect	0.000	+	0.000	0.000	0.000
5	pintapr	0.150	+	0.150	0.150	0.150
6	pirrpas	0.150	+	0.150	0.150	0.150
7	plife	0.000	+	0.000	0.000	0.000
8	pnatveg	0.075	+	0.075	0.075	0.075
9	presid	0.050	+	0.050	0.050	0.050
10	pwaste	0.050	+	0.050	0.050	0.050
11	pwwfail	0.025	+	0.025	0.025	0.025
12	tbuffer	0.150	-	0.150	0.150	0.150
13	tfence	0.010	-	0.050	0.050	0.010
14	tflows	0.020	+			0.020
15	tgulero	0.020	+	0.025	0.020	0.020
16	tirrigat	0.020	+	0.025	0.020	0.020
17	train	0.060	+	0.025	0.020	0.060
18	tresdist	0.020	+	0.025	0.020	0.020
19	tsaw	1.000	*	1.000	1.000	1.000
20	tslope	0.000	+	0.025	0.020	0.000
21	tstrdist	0.025	+	0.025	0.025	0.025
22	twatero	0.025	+	0.025	0.025	0.025

20190725 100%

H.2: Nutrients



	A	B	C	D
1	**Layer**	**Coeff**	**TermOper**	sens11
2	Reference		20190808	20191006
3	nfrunut	0.045	+	0.045
4	ngraz	0.085	+	0.085
5	nintanpr	0.085	+	0.085
6	nintprod	0.085	+	0.085
7	nmanfor	0.030	+	0.030
8	nnatveg	0.070	+	0.070
9	nnirrmodp	0.085	+	0.085
10	nrures	0.030	+	0.030
11	nshrub	0.035	+	0.035
12	nurbser	0.035	+	0.035
13	nvegherb	0.035	+	0.035
14	nvines	0.030	+	0.030
15	tbuffer	0.135	-	0.135
16	tfence	0.025	-	0.025
17	tflows	0.020	+	0.020
18	tgulero	0.020	+	0.020
19	tirrigate	0.020	+	0.020
20	train	0.060	+	0.060
21	tresdist	0.020	+	0.020
22	tsaw	1.000	*	1.000
23	tslope	0.000	+	0.000
24	tstrdist	0.025	+	0.025
25	twatero	0.025	+	0.025

nutrients 100%

Appendix I: Nutrient-specific variables added for the SAW nutrients study

Note: These variables were used to hold values of the risk of Phosphorus contamination in surface water run-off from each land use type.

#	Layer name	Land use
1	ngraz	Grazing
2	nintanpr	Intensive animal production
3	nintprod	Intensive production
4	nnirrmodp	Irrigated modified pasture
5	nmanfor	Managed forest
6	nnatveg	Native vegetation
7	nfrutnut	Olives, tree fruits & tree nuts
8	nrures	Rural residential
9	nshrub	Shrub berries and fruits
10	nurbsur	Urban services
11	nvegherb	Vegetables and herbs, flowers & bulbs
12	nvines	Vine fruits
13	nwater	Water

REFERENCES

AHC 2013, *Guide for Watershed (Primary Production) Zone*, Adelaide Hills Council, viewed 2 November 2017, <http://www.ahc.sa.gov.au/ahc-resident/Documents/Guide%20for%20watershed%20primary%20production%20zone.pdf>.

—— 2017, *Water Management Plan*, Adelaide Hills Council, viewed 9 November 2019, <https://www.ahc.sa.gov.au/ahc-council/Documents/Reports-Strategies-Policies-Plans/Strategies-Plans/COUNCIL-PLAN-Final-Water-Management-2017.pdf>.

Aljassim, M 2018, *Understanding the Impact of Land Use on Microbial Water Quality to Support Decisions for a Future Land Use Plan*, ProQuest Dissertations Publishing.

Alonso Fernández, JR, García Nieto, PJ, Díaz Muñiz, C & Álvarez Antón, JC 2014, 'Modeling eutrophication and risk prevention in a reservoir in the Northwest of Spain by using multivariate adaptive regression splines analysis', *Ecological Engineering*, vol. 68, pp. 80-9.

AMLRNRMB 2017, *Best practice land management guidelines for small grazing properties*, Adelaide and Mount Lofty Ranges Natural Resources Management Board, Adelaide, South Australia.

Australian and New Zealand Environment and Conservation Council & Agriculture and Resource Management Council of Australia and New Zealand 2000, *Australian and New Zealand guidelines for fresh and marine water quality. Volume 1, The Guidelines*, Australian and New Zealand Environment and Conservation Council and Agriculture and Resource Management Council of Australia and New Zealand, Canberra.

Australian Government 2019, *Commonwealth Environmental Water Office*, viewed 10 November 2019, <https://www.environment.gov.au/water/cewo>.

Bradley, J & Billington, K 2002, *Land status data mapping for the Mount Lofty Ranges watershed*, Environment Protection Authority, Adelaide, South Australia.

Bryan, BA, Kandulu, J, Deere, DA, White, M, Frizenschaf, J & Crossman, ND 2009, 'Adaptive management for mitigating Cryptosporidium risk in source water: A case study in an agricultural catchment in South Australia', *Journal of Environmental Management*, vol. 90, no. 10, pp. 3122-34.

Bukhari, Z & Smith, H 1997, 'Cryptosporidium parvum: Oocyst excretion and viability patterns in experimentally infected lambs', *Epidemiology and infection*, vol. 119, no. 1, pp. 105-8.

Chen, Q, Mei, K, Dahlgren, RA, Wang, T, Gong, J & Zhang, M 2016, 'Impacts of land use and population density on seasonal surface water quality using a modified geographically weighted regression', *Science of the Total Environment*, vol. 572, no. C, pp. 450-66.

Coffey, R, Cummins, E, Bhreathnach, N, Flaherty, V & Cormican, M 2010, 'Development of a pathogen transport model for Irish catchments using SWAT', *Agricultural Water Management*, vol. 97, no. 1, pp. 101-11.

Connor, J, Summers, D, Regan, C, Abbott, H, Frizenschaf, J & van der Linden, L 2019, 'The economics of riparian plantings for carbon and water quality benefits in the Mount Lofty Ranges', *Goyder Institute for Water Research Technical Report Series*, no. 19/06.

Cox, GM, Gibbons, JM, Wood, ATA, Craigon, J, Ramsden, SJ & Crout, NMJ 2006, 'Towards the systematic simplification of mechanistic models', *Ecological Modelling*, vol. 198, no. 1-2, pp. 240-6.

Cox, JW, Bald, M, Burch, M, Chittleborough, D, Cuddy, S, Deane, D, Fleming, N, Frizenschaf, J, Green, G & Halfyard, R 2013, 'Water Allocation Planning and Water Quality Improvements Scoping Study–Discussion paper', *Goyder Institute for Water Research Technical Report Series*, no. 13/8.

Cox, JW, Oliver, DP, Fleming, NK & Anderson, JS 2012, 'Off-site transport of nutrients and sediment from three main land-uses in the Mt Lofty Ranges, South Australia', *Agricultural Water Management*, vol. 106, pp. 50-9.

Daniels, CB & Good, K 2015, 'Building resilience to natural, climate and anthropocentric change in the Adelaide and Mount Lofty Ranges region: a natural resources management board perspective', *Transactions of the Royal Society of South Australia*, vol. 139, no. 1, pp. 83-96.

Darnault, CJG, Peng, Z, Yu, C, Li, B, Jacobson, AR & Baveye, PC 2017, 'Movement of Cryptosporidium parvum Oocysts through Soils without Preferential Pathways: Exploratory Test', *Frontiers in Environmental Science*, vol. 5, no. 39.

Department for Health and Ageing 2016, *Safe Drinking Water Act Annual Report 2015-2016*, Government of South Australia.

DEW 2019, *SA Land cover models (2010 - 2015)*, viewed 18 November 2019, <http://www.waterconnect.sa.gov.au/Content/Downloads/DEWNR/LANDSCAPE_SA_Landcover_2010_2015_ML.zip>.

Dooley, T, Ellis, K, Miles, C & Hughes, B 2011a, *Land activity impacts on water quality: risk to aquatic ecosystems and potable water supply*, Rural Solutions SA, Adelaide, South Australia.

—— 2011b, *APPENDICES to the report: Land activity impacts on water quality: risk to aquatic ecosystems and potable water supply.*, Rural Solutions SA, Adelaide, South Australia.

EPA 2000, *The State of Health of the Mount Lofty Ranges Catchments from a water quality perspective*, Government of South Australia, Environment Protection Agency Department for Environment and Heritage, viewed 3 November 2017, <report.epa.sa.gov.au/files/477406_mtlofty.pdf>.

—— 2004, *Mount Lofty Ranges Watershed Protection Office Status Report June 2003*, Government of South Australia, Environment Protection Authority, Adelaide, South Australia.

—— 2005, *Mount Lofty Ranges Watershed*, Government of South Australia, Environment Protection Agency, viewed 16 November 2019, <https://www.epa.sa.gov.au/files/8432_watershed_mlr.pdf>.

—— 2007, *Protecting drinking water quality into the future — priority areas and land use compatibility in Adelaide's Mount Lofty Ranges Watershed*, Government of South Australia, Environment Protection Agency, Adelaide, South Australia.

—— 2019, *Mount Lofty Ranges*, Government of South Australia, Environment Protection Agency, viewed 17 November 2019, <http://www.epa.sa.gov.au/environmental_info/water_quality/programs/mount_lofty_ranges>.

ESRI 2018, *ArcGIS Desktop*, 10.6.1 edn, ESRI Inc., Redlands, California.

—— 2019, *Predict Floods with Unit Hydrographs*, viewed 31 July 2019, <https://learn.arcgis.com/en/projects/predict-floods-with-unit-hydrographs/>.

Federal Register of Legislation 2019, *Water Act 2007*, viewed 7 September 2019, <https://www.legislation.gov.au/Details/C2020C00058>.

Ferguson, C, Husman, AMdR, Altavilla, N, Deere, D & Ashbolt, N 2003, 'Fate and Transport of Surface Water Pathogens in Watersheds', *Critical Reviews in Environmental Science and Technology*, vol. 33, no. 3, pp. 299-361.

Fleming, N, Cox, J, He, Y, Thomas, S & Frizenschaf, J 2010a, *Analysis of Constituent Concentrations in the Mount Lofty Ranges for Modelling Purposes*, eWater Cooperative Research Centre, Adelaide, South Australia.

—— 2010b, *Analysis of Total Suspended Sediment and Total Nutrient Concentration data in the Mount Lofty Ranges to derive event mean concentrations*, ISBN 978-1-921543-32-6, eWater Cooperative Research Centre, Adelaide, South Australia.

Fotheringham, S, Brunsdon, C & Charlton, M 2002, *Geographically Weighted Regression*, John Wiley & Sons, Ltd, Chichester, England.

Friedman, JH 1991, 'Multivariate adaptive regression splines', *The annals of statistics*, pp. 1-67.

Government of South Australia 2019a, *Environment Protection Act 1993*, Adelaide, South Australia.

—— 2019b, *Natural Resources Management Act 2004*, Adelaide, South Australia.

—— 2019c, *Development Act 1993*, Adelaide, South Australia.

Hastie, T 2009, *The elements of statistical learning : data mining, inference and prediction*, 2nd edn, Springer, New York, N.Y.

Hobson, P, Fabris, R, Develter, E, Linden, L, Burch, M & Brookes, J 2010, 'Reservoir Inflow Monitoring for Improved Management of Treated Water Quality—A South Australian Experience', *An International Journal - Published for the European Water Resources Association (EWRA)*, vol. 24, no. 14, pp. 4161-74.

Jenkins, MB, Anguish, LJ, Bowman, DD, Walker, MJ & Ghiorse, WC 1997, 'Assessment of a dye permeability assay for determination of inactivation rates of Cryptosporidium parvum oocysts', *Applied and Environmental Microbiology*, vol. 63, no. 10, p. 3844.

John, M & Flehr, M 2011, *State of the District Report*, Adelaide Hills Council, Stirling, South Australia.

Kandulu, J & Bryan, B 2009, 'Evaluating alternatives for mitigating Cryptosporidium risk and generating environmental service benefits in water supply catchments', *IDEAS Working Paper Series from RePEc*.

Keane, R 2019, *Digital Elevation Model corrections*, Flinders University (Adelaide), Personal communication.

King, B, Keegan, AR, Monis, PT & Saint, CP 2005, 'Environmental Temperature Controls Cryptosporidium Oocyst Metabolic Rate and Associated Retention of Infectivity', *Applied and Environmental Microbiology*, vol. 71, no. 7, p. 3848.

King, B & Monis, P 2007, 'Critical processes affecting Cryptosporidium oocyst survival in the environment', *Parasitology*, vol. 134, no. 3, pp. 309-23.

Maidment, DR, Olivera, F, Calver, A, Eatherall, A & Fraczek, W 1996, 'Unit hydrograph derived from a spatially distributed velocity field', *Hydrological Processes*, vol. 10, no. 6, pp. 831-44.

McGrane, S, Tetzlaff, D & Soulsby, C 2014, 'Application of a linear regression model to assess the influence of urbanised areas and grazing pastures on the microbiological quality of rural streams', *An International Journal Devoted to Progress in the Use of Monitoring Data in Assessing Environmental Risks to Man and the Environment*, vol. 186, no. 11, pp. 7141-55.

McLaren, C 1997, *Dry sheep equivalents for comparing different classes of livestock*, Department of Primary Industries, Victoria, Australia.

Medda, S & Bhar, KK 2019, 'Comparison of single-site and multi-site stochastic models for streamflow generation.(Original Article)(Report)', *Applied Water Science*, vol. 9, no. 3.

Meineri, E, Deville, AS, Grémillet, D, Gauthier-Clerc, M & Béchet, A 2015, 'Combining correlative and mechanistic habitat suitability models to improve ecological compensation', *Biological Reviews*, vol. 90, no. 1, pp. 314-29.

Microsoft Corporation 2016, *Excel*, Redmond, Washigton.

Milborrow, S 2019, *Notes on the earth package*, viewed 11 Oct 2019, <https://cran.r-project.org/web/packages/earth/index.html>.

Miller, R, Guice, J & Deere, D 2009, *Risk Assessment for Drinking Water Sources*, Water Quality Research Australia Limited, Adelaide, South Australia.

Muirhead, RW, Elliott, AH & Monaghan, RM 2011, 'A model framework to assess the effect of dairy farms and wild fowl on microbial water quality during base-flow conditions', *Water Research*, vol. 45, no. 9, pp. 2863-74.

Muñoz, J & Felicísimo, ÁM 2004, 'Comparison of statistical methods commonly used in predictive modelling', *Journal of Vegetation Science*, vol. 15, no. 2, pp. 285-92.

Natural Resources AMLR 2019a, *Adelaide and Mount Lofty Ranges NRM Board*, viewed 10 November 2019, <https://www.naturalresources.sa.gov.au/adelaidemtloftyranges/about-us/about-the-board>.

Neill, AJ, Tetzlaff, D, Strachan, NJC, Hough, RL, Avery, LM, Watson, H & Soulsby, C 2018, 'Using spatial-stream-network models and long-term data to understand and predict dynamics of faecal contamination in a mixed land-use catchment', *Science of the Total Environment*, vol. 612, pp. 840-52.

NH&MRC 2009, *Australian Guidelines for Water Recycling Stormwater Harvesting and Reuse*, Canberra, ACT, Australia.

—— 2011, *Australian Drinking Water Guidelines 6 2011 v3.5 updated May 2019*, Canberra, ACT, Australia.

—— 2019, *National Health and Medical Research Council*, viewed 10 November 2019, <https://www.nhmrc.gov.au/about-us/who-we-are>.

Oliver, DM, Porter, KDH, Pachepsky, YA, Muirhead, RW, Reaney, SM, Coffey, R, Kay, D, Milledge, DG, Hong, E, Anthony, SG, Page, T, Bloodworth, JW, Mellander, P-E, Carbonneau, PE, McGrane, SJ & Quilliam, RS 2016, 'Predicting microbial water quality with models: Over-arching questions for managing risk in agricultural catchments', *Science of the Total Environment*, vol. 544, pp. 39-47.

Oliver, DP, Kookana, RS, Anderson, JS, Cox, J, Waller, N & Smith, L 2012b, 'The off-site transport of pesticide loads from two land uses in relation to hydrological events in the Mt. Lofty Ranges, South Australia', *Agricultural Water Management*, vol. 106, pp. 70-7.

Oliver, DP, Kookana, RS, Anderson, JS, Cox, JW, Fleming, N, Waller, N & Smith, L 2012a, 'Off-site transport of pesticides from two horticultural land uses in the Mt. Lofty Ranges, South Australia', *Agricultural Water Management*, vol. 106, pp. 60-9.

Oliver, DP, Kookana, RS, Anderson, JS, Cox, JW, Waller, N & Smith, LH 2012c, 'Off-site transport of pesticides in dissolved and particulate forms from two land uses in the Mt. Lofty Ranges, South Australia', *Agricultural Water Management*, vol. 106, pp. 78-85.

Packer, S 2010a, *Mount Lofty Ranges Watershed Water Resource Risk Report*, Environment Protection Authority, Adelaide, South Australia.

—— 2010b, *Modelling of Risk Solutions*, Environment Protection Authority, Adelaide, South Australia.

Pisaniello, JD 2010, 'The need for 'adequate' farm dam safety management accountability to avoid dam failure emergencies', *Australian Journal of Emergency Management*, vol. 25, no. 3, pp. 31-8.

Porter, KDH, Reaney, SM, Quilliam, RS, Burgess, C & Oliver, DM 2017, 'Predicting diffuse microbial pollution risk across catchments: The performance of SCIMAP and recommendations for future development', *Science of the Total Environment*, vol. 609, pp. 456-65.

Python Software Foundation 2019, *Python*, viewed 25 September 2018, <https://www.python.org/>.

Rogerson, PA 2015, *Statistical Methods for Geography: A Student's Guide, 4th Edition*, Sage, London.

Roser, D & Ashbolt, N 2007, *Source Water Quality Assessment and the Management of Pathogens in Surface Catchments and Aquifers*, Cooperative Research Centre for Water Quality and Treatment, Salisbury, South Australia.

Rudy, J 2013, *Py-earth documentation*, viewed 11 Oct 2019, <https://contrib.scikit-learn.org/py-earth/>.

SA Water 2019a, *About SA Water*, viewed 12 Oct 2019, <https://www.sawater.com.au/about-us/about-sa-water>.

—— 2019b, *Major pipelines*, viewed 10 November 2019, <https://www.sawater.com.au/community-and-environment/our-water-and-sewerage-systems/our-networks/major-pipelines>.

SA Water Corporation 2014, 'Catchment Risk Assessment: Protozoa', SA Water Corporation, Adelaide, South Australia.

Spies, B & Woodgate, P 2005, *Salinity mapping methods in the Australian context*, Australian Government, Department of Agriculture, Water and the Environment.

Swaffer, B 2014, 'Catchment Risk Assessment: Detailed Method', SA Water Corporation, Adelaide, South Australia.

Swaffer, B, Abbott, H, King, B, van Der Linden, L & Monis, P 2018, 'Understanding human infectious Cryptosporidium risk in drinking water supply catchments', *Water Research*, vol. 138, pp. 282-92.

Swaffer, B, Vial, HM, King, BJ, Daly, R, Frizenschaf, J & Monis, PT 2014, 'Investigating source water Cryptosporidium concentration, species and infectivity rates during rainfall-runoff in a multi-use catchment', *Water Research*, vol. 67, pp. 310-20.

Water Research Australia 2019, *Disability-adjusted life years*, viewed 15 November 2019, <https://www.waterra.com.au/publications/document-search/?download=539>.

Whitehead, PG, Leckie, H, Rankinen, K, Butterfield, D, Futter, MN & Bussi, G 2016, 'An INCA model for pathogens in rivers and catchments: Model structure, sensitivity analysis and application to the River Thames catchment, UK', *Science of the Total Environment*, vol. 572, pp. 1601-10.

Wilkes, G, Edge, TA, Gannon, VPJ, Jokinen, C, Lyautey, E, Neumann, NF, Ruecker, N, Scott, A, Sunohara, M, Topp, E & Lapen, DR 2011, 'Associations among pathogenic bacteria, parasites, and environmental and land use factors in multiple mixed-use watersheds', *Water Research*, vol. 45, no. 18, pp. 5807-25.

Willoughby, N, Thompson, D, Royal, M & Miles, M 2018, *South Australian Land Cover Layers: an introduction and summary statistics*, South Australian Department of Environment and Water, Adelaide, South Australia.

WSAA 2015, *Health-based targets for drinking water safety*, Water Services Association of Australia Limited, Docklands, Victoria, Australia.

Zahedi, A, Monis, P, Gofton, AW, Oskam, CL, Ball, A, Bath, A, Bartkow, M, Robertson, I & Ryan, U 2018, 'Cryptosporidium species and subtypes in animals inhabiting drinking water catchments in three states across Australia', *Water Research*, vol. 134, pp. 327-40.