

**OPTIMAL PATH / NEUTRAL NETWORK APPROACHES TO
MODELING OF FOREST ROAD DESIGN FOR USE IN
AUTOMATED GIS SYSTEMS**

by

IONUT ANDREI ARON

B.Sc., University TRANSILVANIA, Brasov, ROMANIA, 1992

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF FORESTRY

in

THE FACULTY OF GRADUATE STUDIES

(Faculty of Forestry)

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

September 2003

© Ionut Andrei ARON, 2003

Library Authorization

In presenting this thesis in partial fulfillment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Ionut ARON

Name of Author (please print)

Dec 9, 2003

Date

Title of Thesis: Optimal path / neural network
approaches to modelling road design for use
in automated GIS systems

Degree: Masters of Forestry Year: 2003

Abstract

A model that integrates the use of feed-forward Artificial Neural Networks and GIS techniques was developed for preliminary road design in forestry. The model was built and tested with GIS data from Malcolm Knapp Research Forest (approximately 5000 ha).

Artificial Neural Networks having a variety of architectures were trained using this data, in conjunction with a several different learning parameters. Once the neural networks were trained and tested, the knowledge was used to predict based on new data. The results of these predictions consisted in a numerical representation of the “likelihood” of a given cell to contain a road. The resulting values were represented in GIS format as a new cost – surface. With the addition of this new surface, an optimal path function was run to trace the location of the road.

The Artificial Neural Network approach was compared to the classical “Slope – Curvature” approach, traditionally used before in preliminary forest road design with GIS. In all cases, the Artificial Neural Network approach yields better results, both in terms of road length and spatial autocorrelation coefficient.

The results of this study suggest that the technique used may be a reasonable method for determining the preliminary road location in forested areas. The roads generated with the aid of Artificial Neural Networks have a higher spatial autocorrelation with the original road and are also shorter than the ones generated with the Slope – Curvature cost surface. The route selected is not perfect from an engineering point of view; however it still can provide good information for the forest road-planner. The route selected by the computer should be modified based on the field inspection and the knowledge and experience of the specialist.

Table of Contents

Abstract	ii
List of Tables	v
List of figures	vi
Acknowledgements	vii
1. Introduction	1
1.1. Background and Rationale	1
1.2. Objective of study	5
2. Previous research	6
2.1. Road modeling	6
2.2. Artificial Intelligence in Natural Resources	15
3. Study site	20
4. Data collection and preparation	22
4.1. Data collection	22
4.2. Data standardization	24
4.3. Generating the DEM and the raster datasets	26
4.4. Artificial Neural Network processing	28
5. Analytical Methods	30
5.1. Training and testing the Artificial Neural Network	30
5.2. Predicting on new data	37
5.3. Creating the cost surface	37
5.4. The optimal path algorithm	38
5.5. Model calibration and testing	42
6. Results and Discussion	46
7. Conclusions and Recommendations	52
8. References	55
9. Appendices	58

List of tables

Table 1	Existing and derived datasets used in modeling the forest roads	24
Table 2	Roads modeled and criteria utilized to select the study datasets	34
Table 3	Input parameters for the neural network	36
Table 4	Cut-angle values for road H50	51
Table 5	Road lengths table for all roads modeled	53
Table 6	Stream crossings for all roads modeled	54
Table 7	Spatial autocorrelation coefficient values for all roads modeled	55
Table 8	Spatial autocorrelation comparison table for the 5 roads modeled	68

List of figures

Fig. 1	An example of an Artificial Neural Network simple node	19
Fig. 2	An example of a simple Artificial Neural Network	20
Fig. 3	Malcolm Knapp Research Forest Overview Map	22
Fig. 4	Location map for all roads modeled	35
Fig. 5	Architecture of the Neural Network used in modeling the roads	38
Fig. 6	The algorithm for calculating the accumulative cost	42
Fig. 7	The algorithm for calculating the cost distance grid	43
Fig. 8	Back-link positions in the back-link grid	43
Fig. 9	The algorithm for calculating the cost allocation grid	44
Fig. 10	An example of high spatial autocorrelation	46
Fig. 11	An example of low spatial autocorrelation	46
Fig. 12	Map of road H50	63
Fig. 13	Map of road H90	64
Fig. 14	Map of road H110	65
Fig. 15	Map of road A42	66
Fig. 16	Map of road L00	67

Acknowledgments

I would like to acknowledge with gratitude several people and organizations, which contributed greatly to this research project and to my graduate education. First I would like to acknowledge my advisor, Dr. Michael Meitner, whose advice and suggestions in how to prepare the thesis project and compile the data have been of great help.

Next I would like to thank the members of my committee, Dr. John Nelson, Mr. Paul Lawson and Dr. Steven Sheppard for their useful advice and assistance throughout this study.

I am also very grateful to the Malcolm Knapp Research Forest for providing the GIS dataset used in this analysis and much more other information related to this project.

1. Introduction

1.1. Background and Rationale

The forest industry must be increasingly competitive on a global basis, driving the need to continually reduce costs. In light of this, forest companies strive to operate in a cost effective manner while balancing this with ecological and environmental concerns. One major component related to both of these concerns is the construction and use of forest road networks. Forest managers are concerned with how forest-related activities can be conducted in the most economical way by planning an appropriate forest road network that takes into consideration timber transportation and road construction, as well as their environmental impacts.

Approaches and methods to assist in the development and identification of current and future roading requirements continue to be a much needed area of research (Murray, 1998). One reason for this is the expectation that more comprehensive management planning will be carried out given the availability of more detailed and complete spatial information associated with today's forest regions. Another related reason is to ensure that the environmental and economical impact of forest management practices can be estimated. Thus, analytical methods and techniques are essential components of the management planning process associated with forest harvesting and transportation operations.

The planning of a road network to access timber harvest sites is a difficult and time consuming task. Traditional methods for designing a forest road system consisted largely of aerial photo interpretation and field reconnaissance. More recently, forest engineers have used large-scale contour maps to select preliminary routes using dividers, a process known as route projection. The engineer then calculates the total construction, transport and road maintenance cost for each alternative. This is both an expensive and time-consuming task given that the forest management plan typically covers thousands of hectares and tens of thousands of timber entry points over a planning period of many years. Sessions and Liu (1993) have discussed the need for tools and techniques to assist in the development of road network systems in forest management, particularly within the context of accessing timber harvest sites.

Designing and evaluating several alternative road networks is generally impractical, because of the complexity and time-consuming nature of the task. Planners are thus typically restricted to reviewing only one or two alternative networks, often only superficially (Walker and Lougheed, 1989). Therefore there is a clear need for an automated method to assist forest managers and specialists in planning their forest road network.

The introduction of Geographic Information Systems (GIS) represents an important step forward in forest planning and modeling. This technology provides a new means of integrating information in a way that helps us understand and address forest management problems. GIS helps us to organize data and to

understand their spatial relationship, providing a basis for making more precise and intelligent decisions.

There are several GIS tools that use raster GIS and Digital Terrain Models (DTMs) such as ESRI' s Spatial Analyst, ESRI' s ArcGRID toolbox, both of which have predefined functions that can help the specialist in solving the problem of preliminary road design. These of-the-shelf tools are based on the use of shortest path algorithms. These algorithms attempt to minimize cumulative cost along a path between two or more cells in a two dimensional array by calculating the accumulative cost over a surface and by compensating for the actual distance that must be travelled taking into account horizontal and vertical factors influencing the total cost of moving from one location to another. The path with the least cumulative cost is then considered as the optimal path for the road.

One disadvantage of this approach is that the optimal path calculated by the computer does not always coincide with the path that would be chosen by the specialist in the field. A number of possible explanations for this discrepancy exist, such as:

- the underlying data is of insufficient accuracy or reliability to support the analysis;
- the knowledge of the specialist is not well represented in the optimal path model;
- the mathematical combination of all factors involved in road planning does not address the interdependence of these factors, which means

that a road location may be a nonlinear and non-separable function of the combination of all these factors.

My thesis addresses some of these limitations by developing a model that integrates the use of traditional GIS cost – analysis tools with the use of neural networks that have the ability to “learn” and to generalize from limited information.

The goal of this model is not to replace detailed ground survey and engineering plans but more intelligently direct and limit the possibilities that need to be considered, as well as highlighting road locations that are not obvious but might be feasible. This model indicates where such survey should be undertaken, thus reducing the time and cost of determining the final road layout. Moreover, by allowing the user to vary the input parameters and constraints, the model permits considerable flexibility in the evaluation of alternative road networks.

1.2. Objective of Study

The objective of this thesis is to determine whether artificial neural networks combined with GIS techniques might be useful in preliminary design of roads in forestry. To do this, a computer model will be developed by integrating GIS cost – analysis tools such as shortest path algorithms with artificial neural networks that have the ability to “learn” and generalize from limited information.

2. Previous Research

2.1. Road modeling

A key component of integrated forest management planning in Canada is a road plan that shows the network of forest roads to be constructed, maintained and decommissioned over a period of time to allow access to stands scheduled for harvesting and management. To aid the forest management planner, a number of decision support systems have been developed in recent years that take advantage of Geographic Information Systems, Database Management Systems and advanced analytical methods (e.g. Network, Pegger, Roadview). Some of these decision support systems deal with the entire road network for an extensive area (e.g. Network) while others focus on individual roads or small clusters of roads over one or more periods of time (Pegger, Roadview)

Tan (1992) identifies two primary steps in the typical procedure of automated forest road locating: network formulating and road locating. These steps apply to both the more extensive road networks as well as for individual roads or small cluster of roads. These steps are:

- 1 – Network formulating

- 2 – Road locating

Network formulating is concerned with determining the nodes and links contained in the network. Hogan (1973) identifies links as the existing roads, roads to be constructed or other modes of transportation. Nodes are connected by links and they are the points through which the resources of a management unit must pass

for transportation to markets. The formulation of a network must be performed with care so as to include all feasible routes.

Road locating is concerned with finding the most appropriate route for a road or a road network. For locating a road between a pair of points (origin – destination), heuristic algorithms¹ and shortest path algorithms (Minamikata 1984) can be applied in determining the road route of the least cost; while the dynamic programming technique of Douglas and Henderson (1987) can locate the road route with the greatest benefit. The Traveling Salesman Algorithm (Wijngaard and Reinders 1985) and Minimum Spanning Tree model (Dijkstra, 1984) are useful for finding a road network connecting a certain number of points (e.g. centers of forest areas) with the minimum cost of road construction. These algorithms require some expert instructions to enable these points to be connected by the road network.

When locating forest roads, it is common that the points to be connected are unknown except for the given starting points. In such cases it would be increasingly difficult to locate a forest road that would minimize the total cost of terrain transportation, road transportation, road construction and maintenance. Some algorithms are reported in the literature (e.g. Kobayashi 1984, Nieuwenhuis 1986) for estimating the optimum solution when locating forest roads. The road locating method presented by Kobayashi (1984) uses a heuristic rule that begins with a given starting point and moves iteratively to a

¹ A rule of thumb, simplification, or educated guess that reduces or limits the search for solutions in domains that are difficult and poorly understood. Unlike algorithms, heuristics do not guarantee optimal, or even feasible, solutions and are often used with no theoretical guarantee (www.hyperdictionary.com)

neighbouring node to achieve the greatest benefit to cost ratio at each iteration, until a satisfactory road system is found. In this method, if the routes between non-neighbouring nodes are not found before road locating, then it is difficult to cross over extreme terrain. Alternatively, Niuewenhuis (1986) initially defines the proposed routes for road construction between any pair of the nodes by the least cost path method, and then evaluates the alternative road routes by using the rule of maximum ratio of cell coverage to cost. In this method, timber transportation and forest stands are not considered and thus the cost benefit analysis and spatial analysis concerning the road are limited.

The NETWORK program (Sessions, 1988) addresses the problem of optimization of logging transportation over multiple periods of time. The model answers the question: "Which products should be hauled to which mills in order to maximize discounted net revenue?" The inputs for NETWORK consists of two lists of information: one list is of existing, proposed or alternative links; the other list is of origins where material will enter the network, destinations where material will leave the network, the volume the activity will generate and the year in which the activity will occur. The solution identifies the combination of routes that will be used to provide the minimum total cost or maximum value. NETWORK can optimally solve problems involving only variable costs but uses a *heuristic*, to find good , feasible solutions for cases with fixed and variable costs (Sessions and Sessions, 1988). Examples of variable costs are: transportation costs, road surface rock replacement, etc, while fixed costs might be road construction and maintenance costs such as: ditch and culvert cleanout, reconstruction costs for

existing roads, etc. NETWORK calculates the minimum cost or maximum value network by using a shortest path algorithm to solve the variable cost problem proposed by Ford (Ford and Fulkerson, 1956). Results for network planning problems that involve variable costs will be optimal. For networks with variable and fixed costs, the heuristic rules do not guarantee an optimal solution, but the solution will be feasible. NETWORK uses an empirical formulation of a network, in which the user defines nodes and links manually, and therefore several routes might be missing due to errors in the formulation of the network. Moreover, the model doesn't take into account the slope of the terrain, which is a decisive factor in locating the forest roads, due to limitations in the maximum gradients that transport vehicles can negotiate.

In 1993, Sessions and Liu used an improved version of NETWORK to develop a model for preliminary planning of forest road systems using Digital Terrain Models (DTM). The method identifies feasible road segments, evaluates their variable and fixed costs and then determines the optimal set of road segments to be used and the year in which the roads are to be constructed. The solution procedure has three steps:

- identification of possible road segments on the DTM and calculation of the costs for each road segment,
- network analysis to find the heuristic combination of road segments which minimizes the sum of the construction, maintenance and transport costs, and

- display the results of the analysis on the digital terrain model or contour map so the transportation planner can review and verify the results in the field

The method takes into account the maximum gradients that transport vehicles can negotiate, but environmental factors such as unstable slopes or riparian management areas are not considered. After the road links have been generated from the digital elevation model, a heuristic algorithm (Network II) is used in order to identify the set of road segments, which minimize the sum of construction, transport and road maintenance costs. This algorithm solves an iterative sequence of shortest path problems, at each iteration converting the fixed costs into equivalent variable costs.

The method assumes that there are no existing roads. However, existing roads can be incorporated in the model by identifying them by the closest grid point and adding them to the list of feasible road segments with zero construction costs. The model does not take into account soil information or slope stability information in the creation of feasible road segments. With GIS, this information is becoming more accessible and could be used as a cell attribute to determine:

- If a road segment is permissible
- How soil and ground slope would affect cut and fill slopes, and
- How existing vegetation would affect road construction costs

The previous models address the problem of large-scale road design (individual roads or small number of roads in a limited area). To address the problem of road network design at a smaller scale, R.M. Newnham (1995) developed a computer model named ROADPLAN. The model divides the area into cells, assigns them harvesting priorities and calculates a "roadability" factor based on the number of pixels classified as water or muskeg. Further, the cells are grouped together in cutblocks based on maximum allowable area and minimum allowable volume. Although ROADPLAN considers possible linkages between a potential node and all cells on the existing network, the links that are finally selected are all straight lines. Topography is not considered either in adjusting the construction costs or in limiting the maximum slope for the forest road.

CARPP (Computer Aided Road Projection Program) developed by Anderson and Nelson (2003) considers topography, stream crossings and other road design parameters (such as distance between switchbacks and junction angles) by assigning limits and penalties to each link. Penalties add cost to undesirable links so that the algorithm will favour other links, and limits unacceptable links, thereby restricting the network to only feasible links. However, the solutions are not optimal and there is variation caused by the order of landings and spatial randomness of the node/link set.

All of the existing models that use GIS techniques are limited in terms of road location accuracy due to limitations in GIS data accuracy. With the advent of high-resolution DTM data, computer-aided road design systems have become

more attractive, as they provide quick evaluation of alternatives in a more systematic manner. One method for gathering high-resolution elevation data is with the use of airborne laser mapping technology such as LIDAR (Light Detection And Ranging). LIDAR elevation data has been found to be accurate to within 15 cm (Ahamed et al. 2000).

The model developed by Coulter (Coulter et al, 2001) uses LIDAR data to design forest road segments, by estimating accurately and quickly the cut and fill earthwork that occurs during forest road construction. The method does not calculate the road layout but it uses user-defined centreline points and then searches alternate road locations to minimize earthwork while satisfying user-specified constraints and design criteria.

The method may significantly decrease road design time and effort, both in the field and in the office. An engineer in the field equipped with a Global Positioning System (GPS) unit would need to make far fewer field measurements in order to complete road design and earthwork calculations, meaning a greater number of alternatives could be examined. If an area is covered by a high resolution DTM, a field engineer would need only a series of road centre points in order to calculate fairly accurate cut and fill volumes.

At this point, the method applies to straight road segments and does not deal with horizontal or vertical curves. Additionally, this model also does not take in-sloping, out-sloping, curve widening or turnouts into considerations.

“Off – the – Shelf” software, such as RoadEng, AutoCAD, Pegger and RoadView, ArcVIEW and ArcINFO (ESRI 1992) can be used in forest road planning and design.

RoadEng from SoftTree as well as AutoCAD rely on survey data collected in the field to generate terrain models and very detailed engineered road location and construction plans. To use these programs, a road location has to be chosen in advance by the specialist, and accurate surveys must be carried out in the field.

PEGGER and ROADVIEW developed by Rogers (2001) are ArcVIEW extensions that assists in the initial road planning by automating the route projection process, also known as “pegging”. The PEGGER program is a tool for quickly identifying possible route location alternatives given grades specified by the user. The results are then displayed in ROADVIEW, a road visualisation package for ArcVIEW.

This tool does not evaluate additional environmental and economic constraints that must be considered by the professional forester such as soil types, hydrology, property lines and slope classes. Moreover, PEGGER can be used to develop only *one* road segment at the time and it cannot deal with more complex road networks.

ArcINFO is a much more powerful GIS package developed by ESRI that can be used in forest road planning. ArcGRID is a raster or cell-based geoprocessing toolbox that is integrated with ArcINFO. In ArcGRID, a planning area can be divided into individual grids. Each grid could be a section of road

depending on its surroundings (slope, earthwork, etc). There are three functions that can be used for road location: COSTDISTANCE, PATHDISTANCE and COSTPATH. These functions are based on use of a shortest path algorithm. COSTDISTANCE and PATHDISTANCE are similar functions, but the PATHDISTANCE function considers horizontal and vertical factors. Horizontal factors determine the difficulty on moving from one cell to another while accounting for the horizontal elements that may affect the movement. The vertical factors determine the difficulty of moving from one cell to another while accounting for the vertical elements that may affect the movement. To determine this vertical factor, the slope between the FROM cell and the TO cell is calculated from the values defined in the input grid.

Using one of these functions, one can find the least cumulative cost from a cell (landing) to a source cell (existing roads). The path from a cell to a source cell is found by the COSTPATH function.

If we consider only the slope factor, these functions can be used directly for preliminary planning of forest roads. However, in forest road planning, considering only the grade factor is not adequate. We must take into account some other important factors such as hydrology, soil stability, ecological factors, earthwork, etc. The goal of this study is to provide improvements to the way that cost-benefit analysis and spatial analysis are performed.

2.2. Artificial Intelligence in Natural Resources

Geographic Information Systems as an aid to the decision making process provide excellent capabilities for understanding and processing spatial data, but very often this is overlooked in favour of the “presentation” of data. With the development of Artificial Intelligence (AI) based technologies in recent years, new opportunities have emerged to enhance the tools used to process spatial data (Deadman and Gimblet, 1995). Artificial Intelligence techniques have tremendous capabilities to process real-world situation and simulate how the human brain process spatial data and therefore combined with GIS can be successfully used in making intelligent decisions in problems with a spatial component.

Traditionally, Artificial Intelligence techniques have been used in areas such as: robotics, natural language recognition, image recognition, detection of medical phenomena, stock market predictions, etc but there are a few applications in Forestry as well, such as: landscape planning, growth and yield modeling, forest stand age calculation from satellite imagery, estimation of probability for landslides, etc. This wide gamut of applications has conducted to the development of a large variety of new techniques and methods that have been incorporated in desktop applications and thus making Artificial Intelligence increasingly available to researchers and specialists. These commercially available software packages include Artificial Neural Networks (ANN), Genetic Algorithms, Expert Systems, Cellular Automata etc.

Since GIS systems are data intensive and many different types of data (e.g. remote sensing, topography, hydrology, etc) are used by specialists in the decision making process, AI may prove useful in capturing expertise and representing knowledge as an aid to decision making. In light of these applications and in conjunction with the suitability of the structure of the input data, ANNs were chosen as the method to be investigated for applicability to the road planning problem.

An artificial neural network, sometimes referred to as simply a Neural Net, differs from other computer algorithms in that it is not rule-based. Neural networks are applicable in virtually every situation in which a relationship (or pattern) between the input variables and output variables exists, even when that relationship is very complex and not easy to articulate in common statistical terms. Neural nets are used in natural resource application development in areas such landscape planning, forestry, geology and can be trained to respond to a variety of input variables, of spatial or non-spatial form.

An artificial neural network is trained to recognize and generalize the pattern or the relationship between a set of inputs and outputs and consists of many simple processing units (or nodes) connected together.

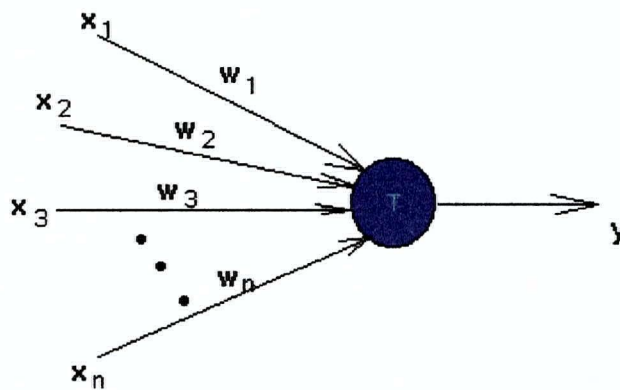


Fig. 1 An example of an Artificial Neural Network simple node

- x_1, x_2, \dots, x_n are inputs. These could be real numbers or Boolean values depending on the problem.
- y is the output and is Boolean.
- w_1, w_2, \dots, w_n are weights of the edges and are real valued.
- T is the threshold and is real valued.

The output y is 1 if the net input which is

$$w_1 x_1 + w_2 x_2 + \dots + w_n x_n$$

is greater than the threshold T . Otherwise the output is 0.

The behaviour of each node is very simple, but together a collection of nodes can exhibit many sophisticated behaviours and be used to solve complex tasks.

The knowledge in a NN is stored as the strength of the interconnection weights (a numeric parameter), which is modified through a process called learning, using a training algorithm. The algorithmic function in conjunction with a learning rule is used to modify the weights in the network in an orderly fashion.

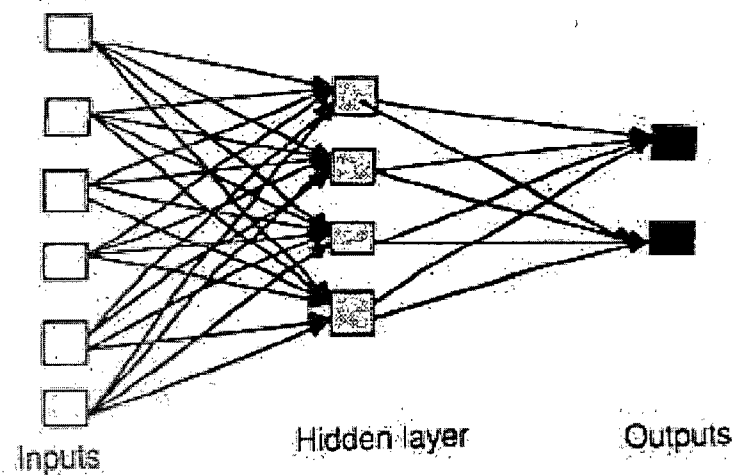


Fig. 2 An example of a simple Artificial Neural Network

The most common type of artificial neural network consists of three groups, or layers of units. The first layer is called the input layer and contains one node (or neuron) for each of the inputs in the training data file. The last layer is called the output layer and contains one neuron for each of the network outputs. Between the input and output layers are an arbitrary number of hidden layers each containing an arbitrary number of neurons. Each neuron is connected to every other neuron in adjacent layers by a set of weights (see fig.2). The weights define the 'strength' of the flow of information from one layer to the next through

the network. Each weight can take on any positive or negative value. "Training" a neural network is simply the process of determining an appropriate set of weights so that the network accurately approximates the input/output relationship of the training data.

Unlike most computer applications, an Artificial Neural Network is not "programmed", rather it is "taught" to give an acceptable answer to a particular problem. Input and output values are given to the network, initial weights are assigned and then the NN repeatedly adjust those interconnection weights until it can successfully produce output values that match the original values. This weighted matrix of interconnections allows the neural network to learn and apply this knowledge to new datasets (Easson and Barr, 1996). In Chapter 4.4 "Artificial Neural Network processing", I will discuss into more detail the parameters utilized for the road location application.

Although there are simple commercial NN tools available that appear as simple to use as a spreadsheet, making the best use of these techniques require some understanding of the underlying algorithms. Therefore it is important to consider carefully how to map features of the problem to input/output layers, what (if any) hidden layers to have in between, what learning method to use and how to set parameters and initialize the weights.

3. Study site

The University of British Columbia Malcolm Knapp Research Forest was established in 1949 as a facility for research, education and demonstration in the practice of Forestry. The forest is situated about 60 kilometres East of Vancouver, British Columbia, at a latitude 49° 18' N, longitude 122° 35' W.

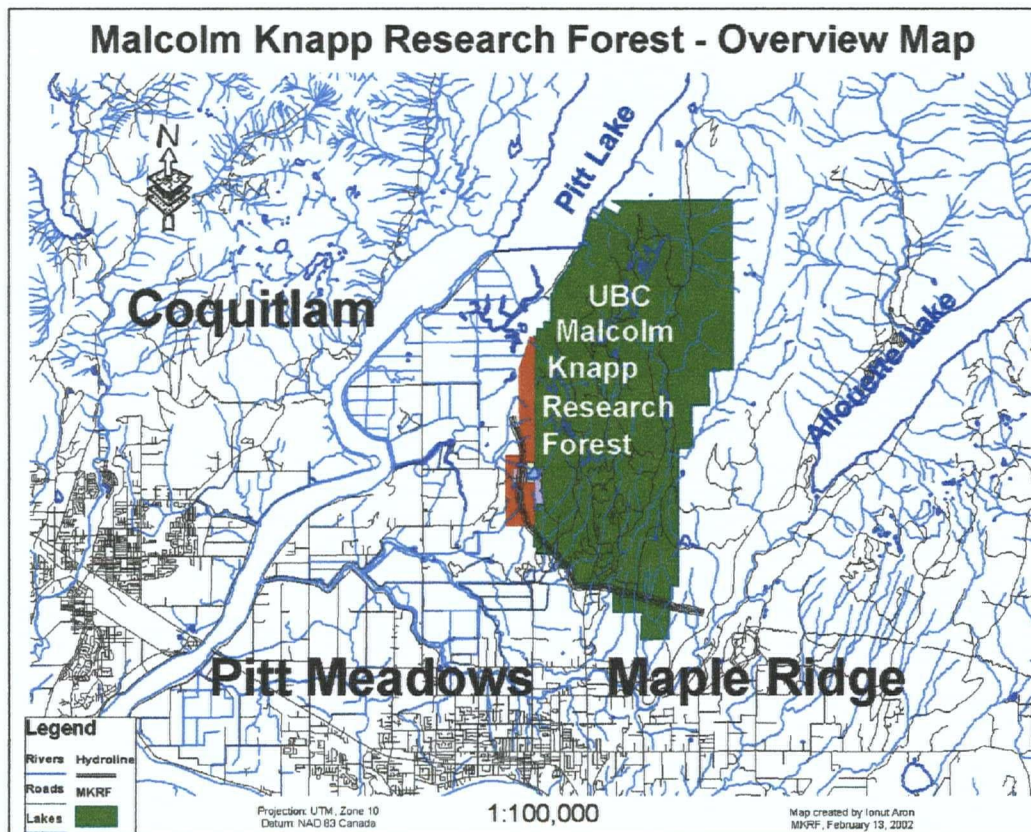


Fig. 3 Malcolm Knapp Research Forest Overview Map

The forest is 5,157 hectares in size, with an average width (from west to east) of 4 km and an average length (from south to north) of 13 km. The area is typical lower coastal topography – mountain lakes, steep slopes and rock outcrops in the North and more gentle slopes of glacial till in the South. Elevation range from

sea level to 1025 meters. For more detailed information, please visit www.forestry.ubc.ca/resfor/mkrf.html. The Research Forest has an extensive road network (over 100 km) covering the entire land-base, in a variety of terrain conditions and with various degrees of complexity. This, combined with an extensive GIS dataset, makes the Research Forest an excellent candidate for developing and testing the road model.

4. Data collection and preparation

4.1. Data collection

The dataset contains ArcView and Arc/Info coverages as well as 0.5 m resolution ortho-photos needed for the project (Table 1).

Table 1 Existing and derived datasets use din modeling the forest roads

Dataset	Existing	Derived
Contour lines	√	
Lakes	√	
Forest cover	√	
Soil Bearing Capacity	√	
Streams	√	
Roads	√	
Openings	√	
Slope		√
Side Slopes		√
Curvature		√

The actual contour lines have a 20 m interval and so new contour lines (5 m interval) were digitized using an Arc/Info procedure in order to provide a more

accurate source for Digital Elevation Model creation, necessary for a good representation of the terrain in a road-modeling project of this level of detail.

Two coverages were digitized, each using a different method:

- 5 m contour lines were digitized manually from 1:10,000 MYLAR maps using the TRACE module from ArcINFO 7.2. This coverage is extremely important for the entire model, since it provides the basis for the creation of a highly accurate Digital Terrain Model (DTM), which is further used to generate slope and curvature grids. The MYLAR maps were created from 1:20,000 air photo interpretation done by hand over 15 years ago and are very accurate compared to similar information in use elsewhere in BC (Lawson, pers.com.)

- Ecology maps were digitized from geo-referenced scanned maps (Klinka 1975) using the on-screen digitizing procedure.

After digitizing, elevation data was added to the contour lines and soil information to the ecology maps. The soil data was further processed in order to calculate the soil bearing capacity for the entire area. For more information on soil bearing capacity calculation, please see Appendix 3.

4.2. Data standardization

For this project, a set of GIS standards were developed and implemented. These standards refer to the data format, projection, storage precision, registration, file names, metadata and topology. Below is a short description of each of these standards.

Format

Spatial data is stored in ARC/INFO coverage format.

Storage Projection and Datum

The coverages are projected in the Universal Transverse Mercator (UTM) Zone 10, and the coordinates are in meters. The datum is North American Datum 83 (NAD 83 Canada).

Storage Precision

Coverages are created in single precision coordinates (preserving about 7 decimals). This is sufficient for the dataset used in the analysis, as UTM projection coordinates can be stored in single precision with meter accuracy.

Data registration

All the existing datasets were subject to affine transformation (a warping process that preserves ratio of distance along the lines) in order to align them with the new 0.5 m color ortho-photo to ensure dataset consistency and accuracy.

Coverage Names

Each coverage's name is composed of a descriptor referring to its thematic content or what kind of information it represents. The next (optional) character '_' is simply a separator. The rest of the name refers to the road to be modeled. This convention is extended to the raster datasets as well. Some examples of coverage and grid names formed according to the naming scheme would be: 'road_h50', 'lake_h110', 'slope_a42', etc.

Metadata

Every coverage has metadata associated to it. Metadata is "data about data", providing information such as: who is responsible for the dataset, date of creation, data source, spatial extent, accuracy, etc.

Clean Topology

All coverages are topologically clean and correct and all polygon boundaries meet exactly.

4.3. Generating the DEM and the Raster datasets

The model is raster based which means that a Digital Elevation Model (DEM) was created as a basis for the elevation information as well as for generating other raster coverages used in the analysis, such as slope, curvature, etc. A DEM provides a digital representation of a portion of Earth over a two dimensional surface. The basic data for a DEM is based on terrain elevation observations that are derived generally from one of three sources: digitized contours, photogrammetric data capture (including aerial photography and digital satellite imagery), and surveying. DEMs derived from hypsography (topographic relief) or contours are perhaps the most common of all sources.

For this project a Triangulated Irregular Network (TIN) was created from 5m contour lines, streams, lakes and roads, the last three integrated in the model as break lines. Break lines represent a discontinuity in the in the slope of the surface and are used for a better representation of the surface and to improve the display and analysis of TIN.

All the grids used in the model have a 5 m resolution that enables them to better model the terrain conditions and also relates to the road width, which in most cases is 5 meters as well. Increasing and/or decreasing this resolution would decrease the model' s accuracy by introducing interpolation and generalization errors.

From the new TIN, a raster DEM was created as well as other grids, such as:

- slope
- side slopes
- curvature

Slope identifies the steepest downhill slope for a location on a surface. The Slope command takes an input surface raster and calculates an output raster containing the slope at each cell. The lower the slope value, the flatter the terrain is and the higher the slope value, the steeper the terrain (ArcGIS Desktop Help).

Side slopes represent the slope from one cell to all neighboring cells.

Curvature represents the fragmentation of the terrain and it's calculated at each cell center. A positive curvature indicates that the surface is upwardly convex at that cell while negative curvature indicates that the surface is upwardly concave at that cell. A value of zero indicates that the surface is flat (ArcGIS Desktop Help).

4.4. Artificial Neural Network Processing

For an ANN to be used as a tool to interpret spatial data, the map information must be converted into patterns. These patterns consist of a value for each input theme at a given location. During the training of the ANN, the patterns must also contain the value of the accepted output value for each location. Once the trained ANN has produced an interpretive result, the result must be converted back into GIS format for the production of an output map, further used in the analysis.

The following GIS layers were used to collect data from:

- slope
- side – slopes (slope to all neighboring cells)
- lakes
- streams
- soil bearing capacity
- curvature
- roads

The steps required to translate the information, for each input and output from ArcINFO to a PC-based ANN are outlined below. These steps begin after the data has been automated and is in the form of ArcINFO coverages. For a detailed list of commands used, please see Appendix 3.

- (1) Create the Triangulated Irregular Network (TIN) from 5 m contour lines, including the streams, lakes and roads as break lines.
- (2) Generate the slope and side slope grids from the TIN. The raster resolution chosen was 5 m.
- (3) Create a grid of points that will be used to collect the information from all other layers. The points were equally distributed at 5m interval, coinciding with the centroids of the cells.
- (4) Buffer the lakes, streams and roads as follows:
 - lakes : 10m buffer
 - streams: 20m buffer
 - roads: 5m buffer
- (5) Concatenate the information from all the grids and polygon coverages into the point coverage created at Step 3. In this step, a custom made Arc Macro Language (AML) program was used in order to combine all the attributes from all layers into a single GIS database. The process was done for the entire study area, and the AML program ran for 6 days on a Pentium III – 900 Mhz before the data collection was completed. The result is a database with over two million records and 15 variables, database that was imported and stored in Microsoft Access format.

5. Analytical Methods

5.1. Training and testing the Artificial Neural Network

Five case-studies were undertaken to investigate the applicability of the model to preliminary forest road design (see Table 3). Each study had an area of approximately one square kilometer, covering a variety of relief conditions and road configurations. After running the model, the resulting roads were compared to the existing roads on the ground and a spatial autocorrelation coefficient was calculated in each case.

Table 2 Roads modeled and criteria utilized to select the study datasets

Study	Road name	Length (m)	Road complexity	Terrain conditions	Training dataset
1	H50	1,065	Simple	Steep (over 30°)	Local
2	H110	1,151	Medium		Local
3	H90	1,554	Complex		Local
4	A42	715	Simple	Flat (0-15°)	Local
5	L00	3,406	Complex	Medium (15° - 30°)	External

For each road that was modeled, a smaller dataset (40000 – 65000 records) was clipped from the database created in step 5 to be used in analysis. The smaller dataset clipped in the previous step was then exported in ASCII

format, using MS Access Export function, in order to be compatible with QwikNet Professional Neural Network software (<http://qwiknet.home.comcast.net>).

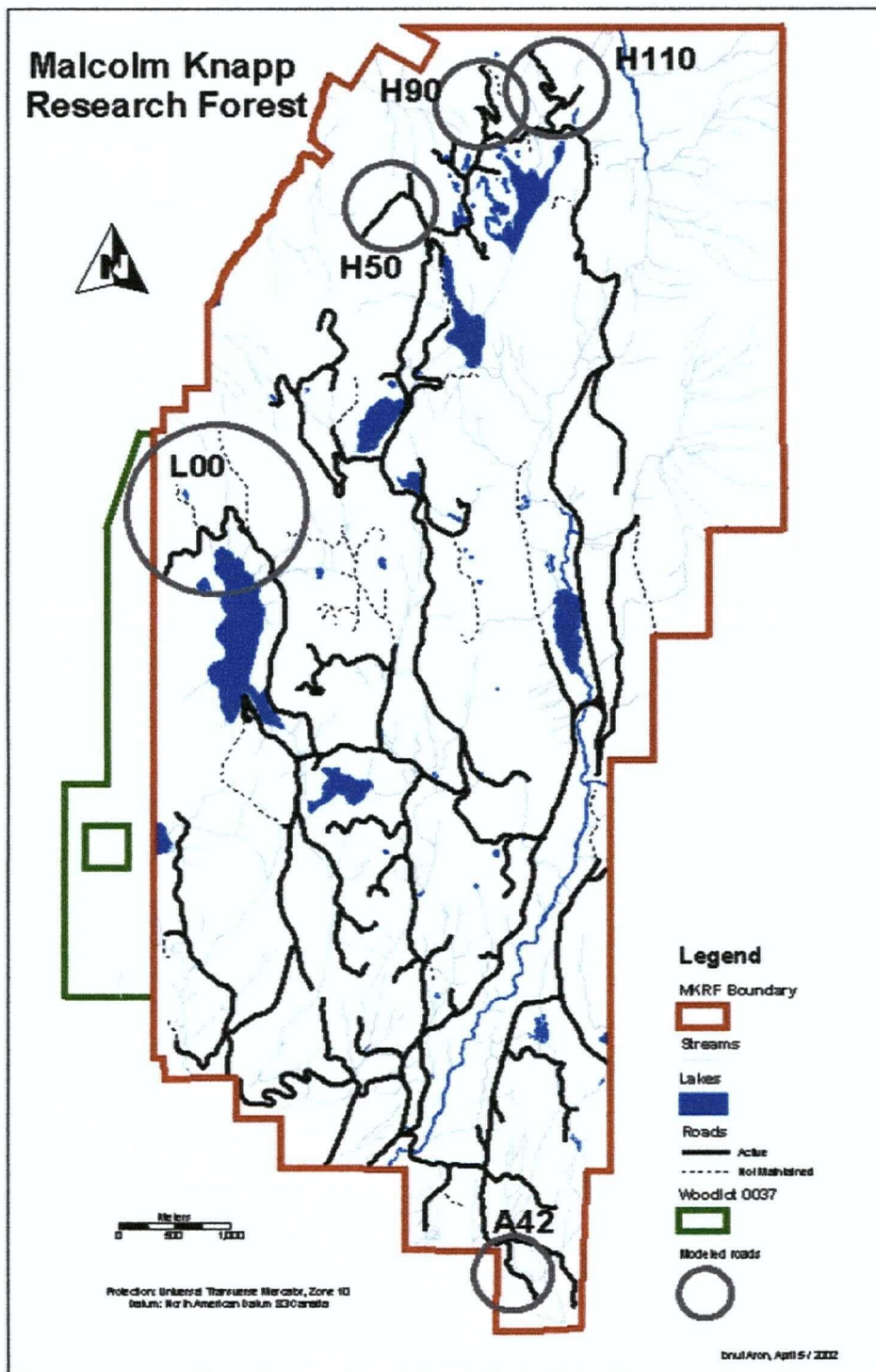


Fig.4 Location map for all roads modeled

A two layer back – propagation ANN with 5 neurons in each hidden layer was used for this project. The neural network implemented is a feed-forward multi-layer perceptron, which simply means that it is made up of several layers and the data propagates forward through the network from input to output. The first layer is called the input layer and contains one node for each of the inputs in the training data file. The last layer is called the output layer and contains one neuron for each of the network outputs. Between the input and output layers there is one hidden layer, containing five neurons. Each neuron is connected to every other neuron in adjacent layers by a set of weights. The weights define the ‘strength’ of the flow of information from one layer to the next through the network. Each weight can take on any positive or negative value. “Training” a neural network is simply the process of determining an appropriate set of weights so that the network accurately approximates the input/output relationship of the training data.

When a grid cell is processed, the values associated with that cell are scaled automatically by the software and become the values of the input nodes. Data scaling is essential for neural networks to be able to properly learn from training data. The type of activation function chosen for the output layer determines the range that the training data must be scaled for. These scaled values are then passed to the nodes in the hidden layer connected to the input nodes. However, rather than being passed directly, they are first multiplied by the weight associated with the connections between the nodes. The receiving node then applies a sigmoid function to the sum of values they receive and the result is

passed to the nodes in the output layer. The following parameters were considered as input for the neural network:

Table3. Input parameters for the neural network

	Variable name	Description									
1.	Slope	Slope for the cell that is processed (percents)									
2.	Slope_NO	<div>Variables 2 to 9 represent side slopes in percents as show in the diagram below:</div> <table><tr><td>Slope_NE</td><td>Slope_NO</td><td>Slope_NW</td></tr><tr><td>Slope_WE</td><td>Slope</td><td>Slope_EA</td></tr><tr><td>Slope_SW</td><td>Slope_SO</td><td>Slope_SE</td></tr></table>	Slope_NE	Slope_NO	Slope_NW	Slope_WE	Slope	Slope_EA	Slope_SW	Slope_SO	Slope_SE
Slope_NE	Slope_NO		Slope_NW								
Slope_WE	Slope		Slope_EA								
Slope_SW	Slope_SO		Slope_SE								
3.	Slope_NW										
4.	Slope_WE										
5.	Slope_SW										
6.	Slope_SO										
7.	Slope_SE										
8.	Slope_EA										
9.	Slope_NE										
10.	Lake	<div>Presence or absence of a lake</div> <div><div>- 1 if a lake is present</div><div>- 0 if there is no lake</div></div>									
11	Stream	<div>Presence or absence of a stream</div> <div><div>- 1 if a stream is present</div><div>- 0 if there is no stream</div></div>									
12	QF	Soil bearing capacity									
13	Curvature	Terrain curvature									
14	Road	<div>Presence or absence of a road</div> <div><div>- 1 if a road is present</div><div>- 0 if there is no road</div></div>									

The final neural network architecture is shown in the figure below:

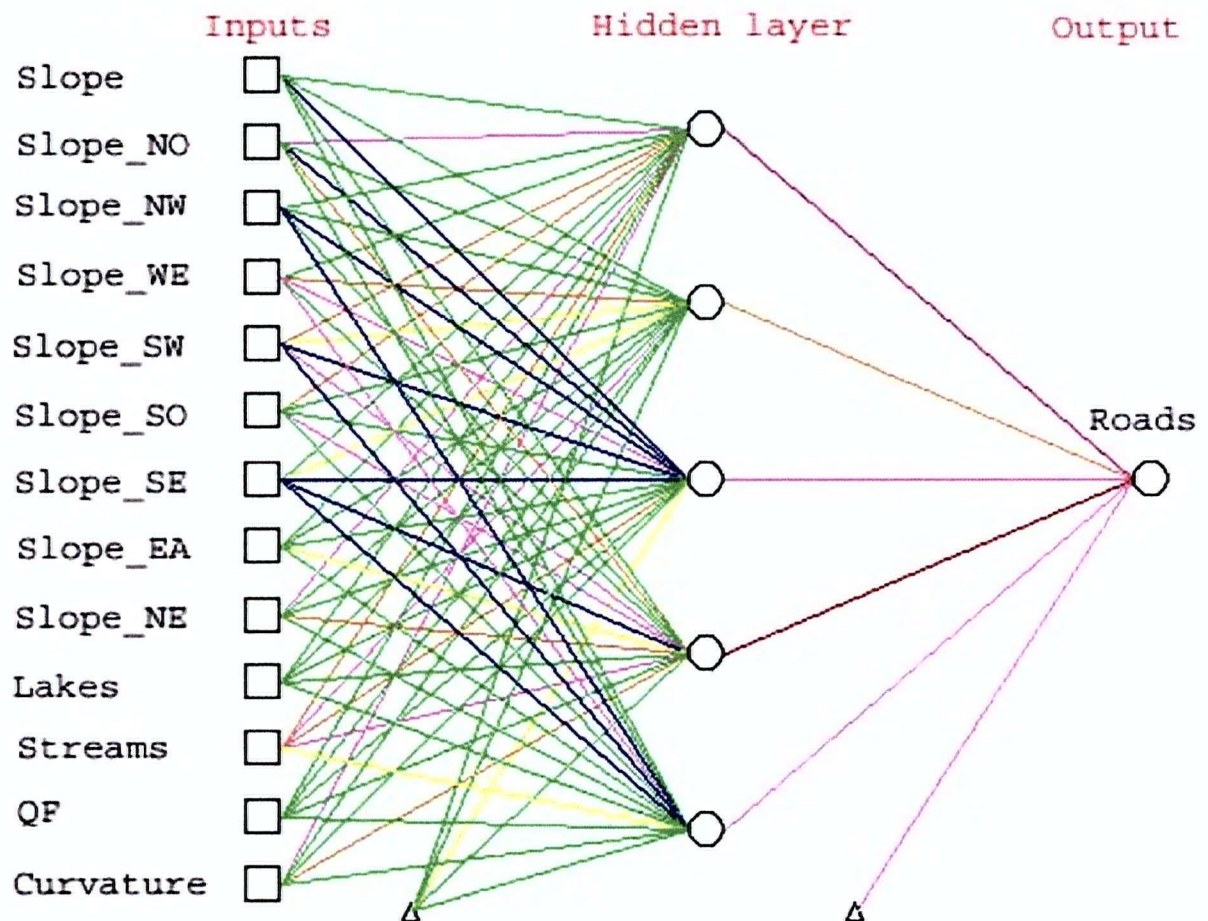


Fig.5 Architecture of the Neural Network used in modeling the roads

Once the architecture and learning method were chosen, and the training parameters were set, the training session started with sample data (approximately 70% of the entire dataset), so the internal weights can start to adjust and learn patterns and trends in the data. Initially, the weights assigned to each connection are set to random values. The weights are then adjusted based upon the training data by undergoing many iterations of a training function, such

as the back-propagation. The back-propagation is the most commonly used training algorithm for neural networks. The weights (w) are updated as follows:

$$\Delta w_{ij}(t) = -\eta \frac{\delta E(t)}{\delta w_{ij}(t)} + \alpha \Delta w_{ij}(t-1) \quad [1]$$

where η is the learning rate and α is the momentum.

The learning rate parameter controls the rate at which the neural network learns and the momentum parameter controls the influence of the last weight change on the current weight update. Higher values for these parameters lead to faster learning but this can cause the network to be unstable and stop learning.

The relationship learned by the NN must be tested before being used in predictions, in order to determine the quality of the knowledge. Testing of the trained NN requires that inputs, but not the outputs, for another portion of the training data be presented to the NN. The accepted output is compared to the output produced by the NN to see if the NN's output is acceptable. If the output produced by the trained NN is correct within accepted error ranges (RMS error < 0.1), the NN can be used with new data. The testing dataset was represented by the remaining 30% of the dataset.

5.2. Predicting on new data

After the net is trained, it was used to predict based on input data. The results of these predictions consisted in a numerical representation of the “likelihood” of a given cell to contain a road. The resulting values range from 0 to 1, with 0 representing cells with the lowest likelihood to contain a road (for example a cell that is on a lake) and 1 representing cells with the highest likelihood to contain a road (for example a cell that is on an existing road). The resulting values were represented in a GIS format as a new cost surface.

5.3. Creating the cost surface

To calculate the cost value for each cell, a new field was added to the database and the cost was calculated using the equation [1]:

$$\text{Cost} = \frac{1}{\text{Likelihood}} \quad [3]$$

In other words, the cost of having a road on a specific cell is inversely proportional to the likelihood of that cell to contain a road (the higher the likelihood, the smaller the cost and vice versa). This new variable was utilized in producing the cost surface on which the ArcINFO optimal path function was run.

5.4. The optimal path algorithm

With the addition of this new resultant cost surface, an optimal path function was run to trace the optimal path between two or more cells. Instead of calculating the actual distance (Euclidean distance) from one point to another, the function determines the shortest weighted distance (or accumulated travel cost) from each cell to the nearest cell in the set of source cells. The weighted distance functions apply “distance” not in geographic units but in cost units.

The optimal path function will return either the least cost path between the source(s) and destination(s) or a corridor of the least cost paths, respectively. In our case, the source cells represent the existing road network and the destination cells represent landings.

Tracing the optimal path is a two-step process:

- Run the PATHDISTANCE function
- Run the COSTPATH function

The PATHDISTANCE function calculates the accumulative cost over a cost surface, and also compensates for the horizontal and vertical factors influencing the total cost of moving from one location to another. The results from PATHDISTANCE function are used as input in COSTPATH and CORRIDOR grid functions. These functions return either the least cost path between the source(s)

and destination(s) or a corridor of the least cost paths, respectively. In our case, the source cells represent the existing road network and the destination cells represent landings.

The PATHDISTANCE function calculates the shortest weighted distance (or accumulated travel cost) from each cell to the nearest cell in the set of source cells. The function creates three output grids:

1. the cost-distance grid
2. the cost-backlink grid
3. the cost-allocation grid

1. The first step in calculating the **cost-distance grid** is to identify the source cells. Then the cost to travel to each neighbour that adjoins a source cell is determined. Next, each of the neighbour cells is ordered from least costly to most costly in a list. The cell location with the least cost is then removed from the list. Finally, the least-accumulative cost to each of the neighbours of the cell that was just removed from the list is determined.

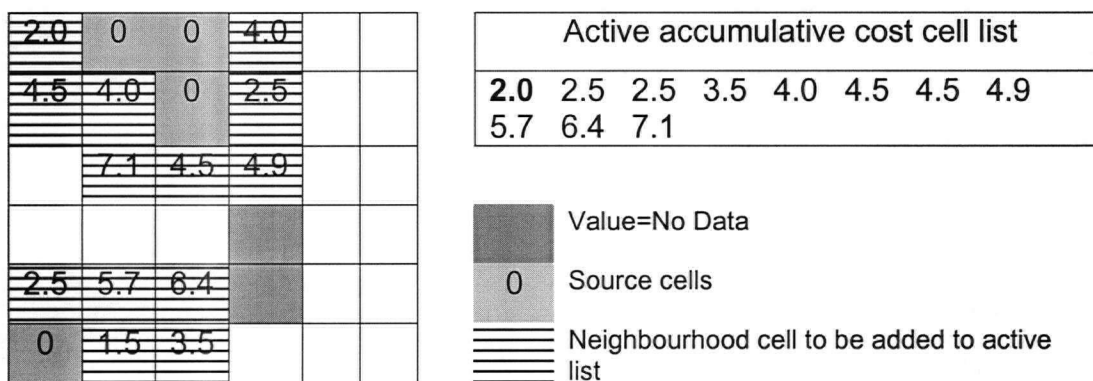


Fig 6. The algorithm for calculating the accumulative cost

The process is repeated until all cells on the grid have been assigned an accumulative cost. The grid does not show which source to return to or how to get there. The accumulative values are based on the cost unit specified on the cost surface.

2. The **cost- backlink grid** with a value range from 0 to 8 that can be used to reconstruct the route to the source. Each value (0 through 8) identifies which neighbouring cell to move into to get back to the source. The source cells are assigned to 0 since they have reached the goal (the source). If a cell on the back-link is assigned 1, the cell immediately to its right will lead to the source with the least costly path. The lower right diagonal cell will be assigned 2, directly south would be value 3 and so forth.

1	0	0	5	4	5
7	1	0	5	5	6
3	8	7	6	6	3
3	5	7		3	4
3	4	4		4	5
0	5	5	5	5	5

6	7	8
5	0	1
4	3	2

Fig 8: Back-link positions in the back-link grid

Fig 7: The algorithm for calculating the cost back-link grid

3. The **cost-allocation grid** identifies which cells will be allocated to which source, on the basis of the lowest accumulative cost to reach the source.

1	1	1	1	1	1
1	1	1	1	1	1
2	1	1	1	1	2
2	2	1		2	2
2	2	2		2	2
2	2	2	2	2	2

Fig 9: The algorithm for calculating the cost allocation grid

In simple terms, the cost-distance grid represents how much it would cost each cell to reach a cell location from a source via the least-cost path. The cost-allocation grid defines, for each cell, the least costly source and the cost back-link identifies how to retrace the path to the source from the destination location.

5.5. Model calibration and testing

The value of a prediction with a model is only as good as the ability of the model to be effectively validated. The validation involves testing the model's predictive capabilities with information other than that used in running the model. In other words, the road modeled has to be compared with the same existing road from the field. Although the existing road might not be perfect from an engineering point of view, it is the best knowledge representation we have with respect to that road. Therefore, the measure of success for a modeled road is represented by the similarity of that road with the existing road from the field.

The basic validation approach was to replicate an existing road from the study area and then to compare the model's output to the existing road with respect to the length and spatial autocorrelation coefficient.

The process has two phases:

- the visual phase in which the newly modeled road is assessed visually and in which a meaningful range of values for the PATHDISTANCE function parameters is established
- the analytical phase in which the length and the spatial autocorrelation coefficient were calculated and compared for different runs and the best parameters were chosen to predict a new road in the study area

Spatial autocorrelation is a measure of the similarity of objects within an area. (ArcGIS help file). A spatial object can have at least two types of attributes:

1. aspatial attributes, such as: elevation, slope, aspect etc)
2. spatial attributes represented by the location of objects in a specified coordinate system. From an object based view of spatial data, spatial autocorrelation measures the relationship between the difference of the aspatial attributes of objects with the distance between the objects.

In our case the a-spatial attributes are represented by a Boolean value (0 for cells with no road and 1 for cells with roads) and spatial attributes are represented by x, y coordinates for a specific object.

Figure 10 presents two spatial objects (in this case two roads) which are close together, have similar a-spatial descriptors and therefore are highly spatially correlated. Figure 11 presents an example of low spatial autocorrelation where objects with similar a-spatial descriptors are more far apart in space.

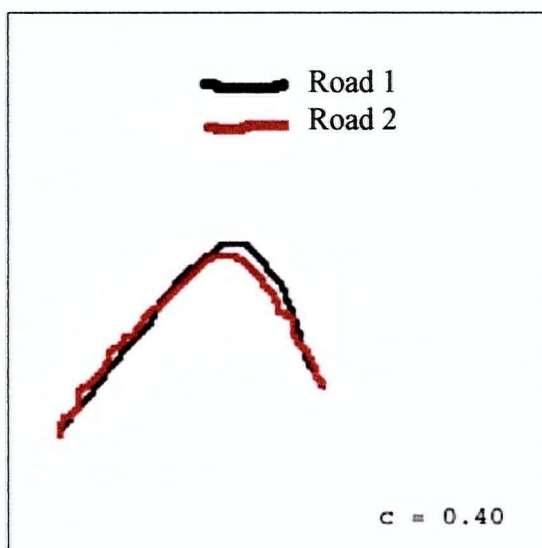


Fig. 10: An example of high spatial autocorrelation

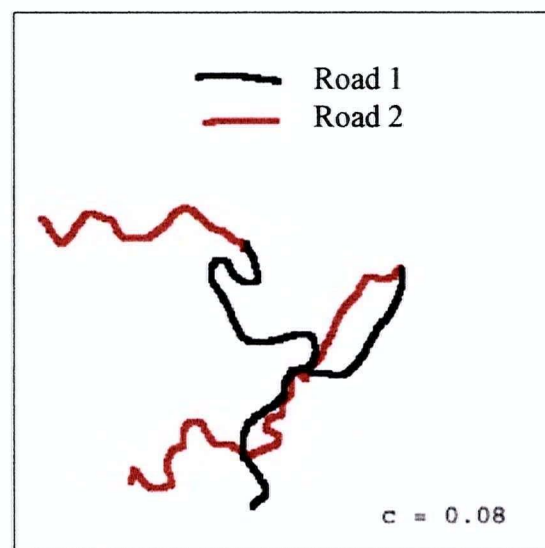


Fig. 11: An example of low spatial autocorrelation

Goodchild (1996) identifies two objects that are very close together and have very different a-spatial attributes as having a negative spatial autocorrelation. In the context of ArcINFO GRID module, the objects correspond to cells and the a-spatial descriptors or attributes correspond to cell values.

The formula for calculating the CORRELATION index is:

$$c = \sum_k^n c_{ij} / (sqrt(\sum_k^n (z_i - \bar{z}_i)^2 * sqrt(\sum_k^n (z_j - \bar{z}_j)^2)) \quad [4]$$

The general notation used in correlation formulas and their GRID interpretation are the following:

n - the total number of cells in a grid

i - any cell on the first input grid

j - any cell on the second input grid

z_i – the value of the attribute of cell i

z_j – the value of the attribute of cell j

\bar{z}_i – the mean value of the attribute of the first grid

\bar{z}_j – the mean value of the attribute of the second grid

c_{ij} – the similarity of i's and j's attributes : $(z_i - \bar{z}_i) * (z_j - \bar{z}_j)$

In the terms of the above notation, spatial autocorrelation is simply a measure of the attribute similarities in the set of c_{ij} with the locational similarities, and then summing the results into a single coefficient (Goodchild, 1986).

Once the model is successfully calibrated, it can be used to project future roads in the study area. Because these predictions cannot be verified at the time, they must be understood in terms of probabilities. The predictions are thought to be a good starting point for the specialist in developing a new forest road. Even if the route selected is not perfect from an engineering point of view, it still can provide good information for the forest road-planning engineer. For preliminary planning the important thing is the relative usefulness of road routes, not the exact road layout. The route selected by the computer should be modified based on the knowledge and experience of the specialist as well as the exact terrain conditions surveyed in the field.

6. Results and discussions

This chapter presents the results for the 5–case study analyzed (see Table 2)

Each study addresses a certain combination of factors that influence road design and construction, including: terrain conditions, stream crossings, road complexity and configuration, etc. To help evaluate the performance of the model, for each road studied, the model was run on two cost surfaces: one generated by the neural network, which accounted for all 12 parameters (see Table 3) and the other by a more traditional approach (Sessions, 1993), which typically accounts for only two parameters, slope and curvature.

The evaluation criteria used for the proposed road was the spatial autocorrelation coefficient calculated for the following combinations:

1. neural net road – existing road
2. slope-curvature road – existing road

H50 was the first road modeled. It has a relatively simple layout but was constructed on very steep terrain (Fig 12), making it suitable to test the model's performance in high slope areas. The second road modeled was **H90**, with a more complex design, on very steep slope and with three switchbacks (Fig13). The road **H110** is the third road modeled, in an area with many streams and lakes with a moderate to steep slope. It is composed of two segments that join together in a “Y” shape, each segment linking a landing to the main road (Fig 14). The forth road modeled is **A42**, which is a very simple segment of road, on an area that is not steep but is crossed by numerous streams (Fig. 15). The last

road modeled was L00, on the North side of Loon Lake, and it is composed of four segments of road linked together. This experiment was conducted to test the applicability of the model in areas with no roads and therefore no possibility to train the Neural Network. The training dataset utilized comes from H90 – H110 area (Fig 16), that has similar characteristics with the area North of Loon Lake where the model was applied.

The Vertical Factor

The PATHDISTANCE function utilized in the model was run using different values for the vertical factor (VF) parameter. This parameter takes into account the cost necessary to overcome the slope between two cells. To calculate the VF, the vertical angle or slope is calculated between cells from the z values assigned to each location and then the slope is correlated to a vertical factor on a graph. There may be a threshold angle such that if the slope exceeds this angle, then the cost is so great that it becomes a barrier to travel. This threshold is referred to as the *cut-angle*. The VF is assigned to infinity when the slope exceeds the cut-angle. Examples of the cut-angle values are shown in the table below:

Table 4 Cut-angle values for road H50

Runs	Cut-angle (percents)	Cut-angle (degrees)
1	100	45
2	30	16.7
3	10	5.7

For H50, the model was run using a range of values for the cut-angle, from 1% to 100%. A visual inspection of the map (Fig 12) shows that the smaller the cut-angle value is, the closer the modeled road is to the original. The best fit is achieved at a cut-angle value of 10% , which is suitable for forest road design. Once the appropriate parameters were established, the model was run and the road length and spatial autocorrelation coefficient were calculated.

For the other roads, the PATHDISTANCE function was run without using the cut-angle parameter and therefore without limiting the maximum slope for the road. By removing the cut-angle from the equation, other parameters than slope (e.g. stream crossing, lakes, networking, etc) became more important in choosing the optimal path for this road. However, this led to creating a proposed road with slopes greater than the maximum slope admissible for a forest road. By introducing the cut-angle parameter in the PATHDISTANCE function, there is a risk that the algorithm stops when no viable alternatives (cells with slope smaller than the cut-angle parameter value) can be found. In real life, when situations like this occur, the planner can prescribe blasting and/or cut and fill operations to overcome the obstacle.

The road length

Table 5 shows that the lengths of the neural network model roads are closer to their corresponding existing road than the length of the slope-curvature model roads. This is important because the cost of building and maintaining a road is directly related to the length of that road.

Table5: Road lengths table for all roads modeled

Road Name	Length (m)		
	Original	Neural Network	Slope – Curvature
H50	1065	1059	1383
H90	1554	1652	2609
H110	1151	1004	1459
A42	714	856	976
L00	3406	3401	8401

The neural network road length varies from 87% of the original road in case of H110 to 120% of the original in case of road A42. The roads predicted with the slope-curvature model vary in length from 127% for H90 to 247% for L00.

Stream crossings

By visually inspecting all the maps, it can be seen that in all cases the roads modeled with the neural network have fewer stream crossings than the roads modeled using slope-curvature (also see Table 6). This indicates that the neural network has learned from the training dataset that stream crossings are “expensive” and therefore assigned a high cost to them. Because of the high cost, the optimal path algorithm, which is trying to minimize the total cost of the road, avoided those areas.

Table 6 Stream crossings for all roads modeled

Road	Stream crossings		
	Original	Neural Network	Slope-Curvature
H50	0	0	0
H90	0	0	3
H110	0	0	6
A42	2	2	1
L00	5	2	-

The spatial autocorrelation coefficient

The spatial autocorrelation coefficient was calculated for all the modeled roads, to shows the degree of similarity between the original road, the neural network road and the slope-curvature road.

Table7:Spatial autocorrelation coefficient values for all roads modeled

Road modeled	Spatial autocorrelation coefficient	
	Original – Neural Network	Original – Slope-Curvature
H50	0.40	0.12
H90	0.41	0.10
H110	0.30	0.08
A42	0.29	0.08
L00	0.28	0.04

Table 7 shows that in all cases the spatial autocorrelation coefficient between the original road and the neural network model is much higher than the spatial autocorrelation coefficient calculated for the slope curvature model. Table 8 in Appendix 2 shows a graphical representation of the spatial autocorrelation differences between the neural network model and the slope curvature model.

7. Conclusions and Recommendations

The research conducted for this thesis attempted to determine whether artificial neural networks combined with GIS techniques might be useful in preliminary design of roads in forestry. First, a cost surface was derived using a neural network algorithm (back-propagation) and then an optimal path algorithm (PATHDISTANCE) was run on the generated cost surface. Finally, the resulting roads were compared to the same roads generated by a slope-curvature model in terms of length, stream crossings and spatial autocorrelation

The results indicate that forest roads can be predicted with a certain degree of accuracy, although the accuracy of the prediction may not be suitable for some applications. In all cases, the neural network model performed better than the slope-curvature model. This technique may be useful for practical forestry in situations when very accurate GIS data is available and the cost of preliminary field work is prohibitively expensive. The utility of this method will increase once high accuracy data collection technologies became cost effective and provide very accurate terrain data for GIS modeling. (e.g. LIDAR – Light Detection And Ranging – with approximately 1 meter horizontal accuracy and 0.3 meters vertical accuracy).

This project is just a starting point for the computer modeling of forest road planning using GIS. There is much room for improving and exploring. From an operational standpoint it would be useful to redo the analysis using TRIM (Terrain Resource Information Management) data, provided by the Ministry of Environment, for an area larger than the Malcolm Knapp Research Forest.

One of the disadvantages of this model is that the predicted roads might have a very short turning radius, because the algorithm is based on a cell-by-cell selection and therefore further studies are needed in order to accommodate logging vehicle requirements.

Another shortcoming of the model is that it doesn't take into account earthwork calculations around horizontal curves and through vertical curves. This, combined with a PATHDISTANCE algorithm that doesn't use CUTANGLE values can lead to the creation of steep road segments that are not feasible from an engineering point of view.

Additionally, experiments could be performed to study the model's behavior in areas with no road network, or where the road network is poorly developed. This will allow the integration of this model with other forest management models that are usually applied to large forested areas over multiple harvest periods (e.g. Forest Planning Studio, CARPP, etc).

One limitation of the approach taken by this research is that in the process of training the artificial neural network utilized only one algorithm (back-propagation). Future work might attempt to investigate this issue, by testing different algorithms with different training parameters.

One of the most beneficial improvements would be to develop a user-interactive and integrated system, where the road locating procedure can be used for drawing the road network and where the users would be allowed to make changes to the network thus designed.

8. List of references:

- Ahamed, K.M., S.E. Reutebuch, T.A. Curtis. 2000. Accuracy of high-resolution airborne laser data with varying forest vegetation cover. *2nd International Conference on Earth Observation and Environmental Information, 11-14 November 2000, Cairo, Egypt.*
- Anderson, A. and Nelson J. 2003 – Procedures for projecting and evaluating forest road networks in strategic plans, *Master thesis, University of British Columbia, Faculty of Forestry*
- Charniak, E. and Mcdermott, D. 1985 – Introduction to Artificial Intelligence. *Addison-Wesley publishing Company, Massachusetts*
- Coulter, E.D., Chung W., Akay. A, Sessions, J. 2001 – Forest road layout using a high resolution digital terrain model generated from LIDAR data. *In preparation for the First International Precision Forestry Symposium, University of Washington, Seattle*
- Craig, R.F. 1992 – Soil Mechanics. *Chapman and Hall, London, 1994*
- Deadman, P. Brown, R.D. Gimblett, H.R. 1993 – Modeling rural residential settlement patterns with cellular automata. *Journal of Environmental Management, Vol.37, pp: 147-160*
- Douglas R.A. and Henderson, B.S. 1987. Computer assisted forest road route location. *Proc of the Council on Forest Engineering 10th annual meeting . High technology in forest engineering, 1987. Syracuse, New York. Pp: 201-214*

- Dykstra, D.P. 1984. Mathematical programming for natural resource management. *McGraw-Hill Book Co., New York. 318p*
- Easson, G.L. ,Barr D.J. - Integration of GIS and Artificial Neural Networks for Natural Resource Applications. *Proceedings of the ESRI user conference Palm Springs, California, May 20-24 1996*
- ESRI, 1992 – Cell based modeling with GRID. *ArcINFO User' s Guide. Environmental Systems Research Institute*
- Ford, L. R., Fulkerson D. R. 1956 Maximal Flow through a Network. *Canadian Journal of Mathematics, vol. 8, pp. 578-580*
- Gimblett, H.R. 1989 – Dynamic modeling in GIS – a cellular automaton approach to modeling the growth behavior of urban and rural development. *GIS National Conference 1989. Ottawa, Canada*
- Goldberg D. 1991 – Genetic algorithms in search, optimization and machine learning. *Addison-Wesley publishing Company, Massachusetts*
- Goodchild, M.F. 1986 – Spatial Autocorrelation. *University of Western Ontario, London, Ontario, Canada*
- Hammond, C., Hall, D., Miller, S. and Swetik, P. 1992 – Level I stability analysis (LISA) Documentation. *US Department of Agriculture, Forest Service, Table 5.5*
- Hogan, J.D. 1973. Using computers in forest road location and design. *Canadian Forest Industries 1993(7) pp. 34-37*
- Klinka, K 1975 Ecosystem maps for University of British Columbia Research Forest, Maple Ridge, BC, Canada

Kobayashi, H. 1984 Planning system for road-route locations in mountainous forests. *Journal of the Japanese Forest Society* 66(8) pp: 313-319

Kouichi Ichihara, Tosimi Tanaka, Isao Sawaguchi, Shuji Umeda and Katsumi Toyokawa 1996 – The method for designing the profile of forest roads supported by genetic algorithm. *Can. J. For. Res. Vol.1*, pp: 45-49

Murray, A.T. 1998 – Route planning for harvest site access. *Can. J. For.Res. Vol.28*, pp. 1084-1087

Minamikata, Y. 1984 Effective forest road planning for forest operations and the environment. *Proc. COFE/IUFRO Conference, Aug 11-18 1984 at University of Maine, Orono*, pp:219-224

Newmhan, R.M. 1995 – ROADPLAN, a tool for designing forest road networks. *Journal of Forest Engineering Vol.6, No.2*, pp: 17-26

Nieuwenhius, M.A. 1986. Development of a forest road location procedure as an integral part of a map based information system. *PhD Thesis, University of Maine, Ann Arbor, Michigan 193p.*

Rogers, L., Schiess, P. 2001 – PEGGER and ROADVIEW – A new GIS tool to assist engineers in operations planning. *The international Mountain Logging and 11th Pacific Northwest Skyline Symposium 2001*, pp. 177-182

Sessions, J. Liu, K. 1993 – Preliminary planning of road systems using Digital Terrain Models. *Journal of Forest Engineering , Vol.9, No.2*

Sessions, J. 1987. A Heuristic Algorithm for the Solution of the Variable and Fixed Cost Transportation Problem. *In Proc: The 1985 Symposium on System Analysis in Forest Resources. Univ. of Georgia, Athens. pp. 324-336.*

Sessions, J. and J.B. Sessions. 1988. A scheduling and network analysis program for tactical harvest planning. In: *Proceedings of the International Mountain Logging and Pacific Northwest Skyline Symposium, Portland, Oregon. December 12-16, 1988. p. 71- 75.*

Smith, David K. 1982 – Network Optimization Practice: a computational guide. *Ellis Horwood Limited, pp.55-92*

Sui, D.Z. 1992 – An initial investigation of integrating Neural Networks with GIS for spatial decision making. *Proceedings of GIS/LIS 1992, San Jose California*

Tan. J, 1992 – Planning a forest road network by a spatial data handling-network routing system. *Acta Forestalia Fennica 227, 1992*

Walker, H.D. and Logheed, W.H. 1989 Road network design in wood supply analysis. *The Forestry Chronicle, December 1989, pp:431-440*

Wang, F. 1992 – Incorporating a Neural Network into GIS for agricultural land suitability analysis. *Proceedings of GIS/LIS 1992, San Jose California*

Wijngaard, P.J.M. and Reiders, M.P. 1985 Optimisation of a forest road network. *Netherlands Journal of Agriculture Science 33 (1985) pp: 175-179*

Appendix 1 – Maps

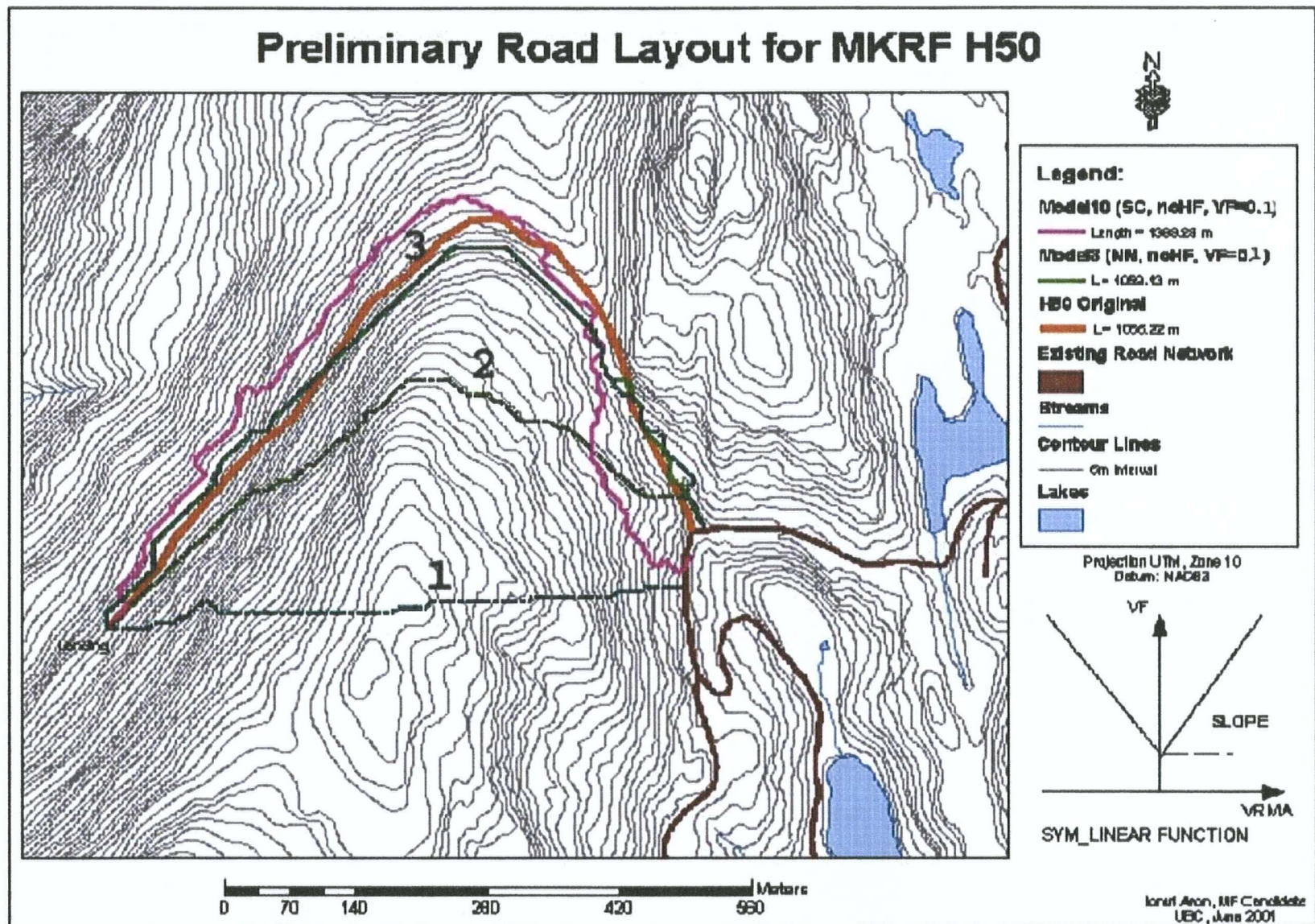
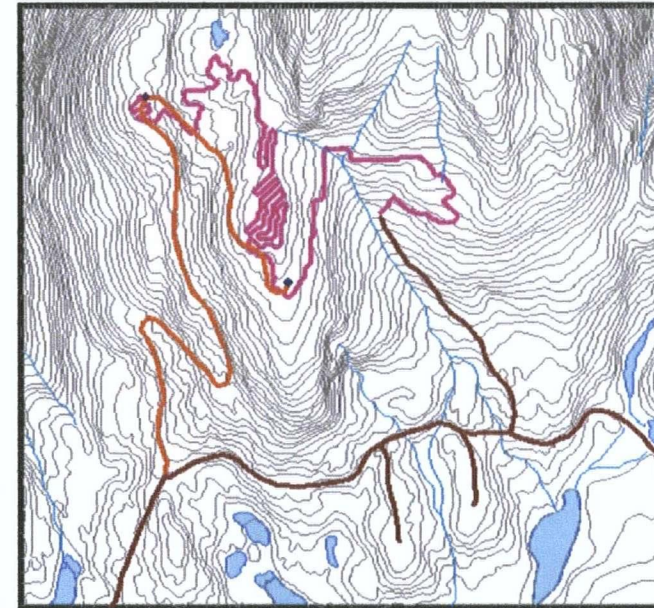
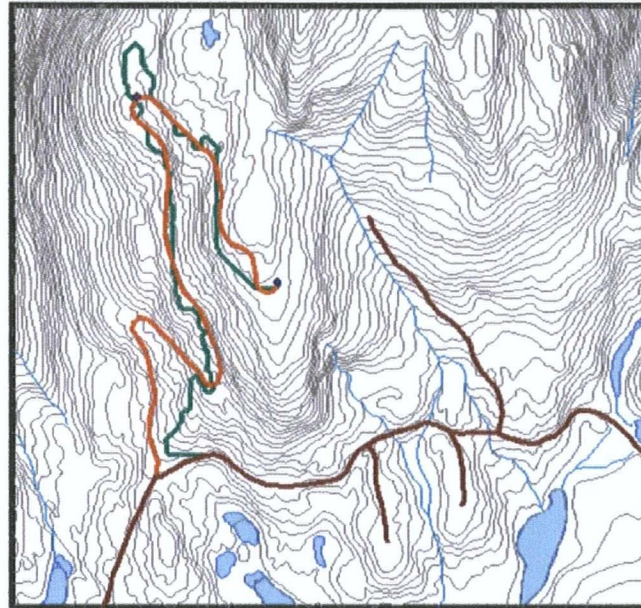


Fig. 12 Map of road H50

Preliminary Road Layout for MKRF H90

Model 1

Model 2



Legend

Contour Lines
— 5m interval

Streams
—

Lakes
■

Cutblocks
♦

Roads
—

H90 Original
—

Length = 1553.92 m

Model 1(NN, noHF, Table)
—

Length = 1652.37m

Model 2(SC, noHF, Table)
—

Length = 2609.09 m



Projection UTM, Zone 10
Datum: NAD 83 Canada

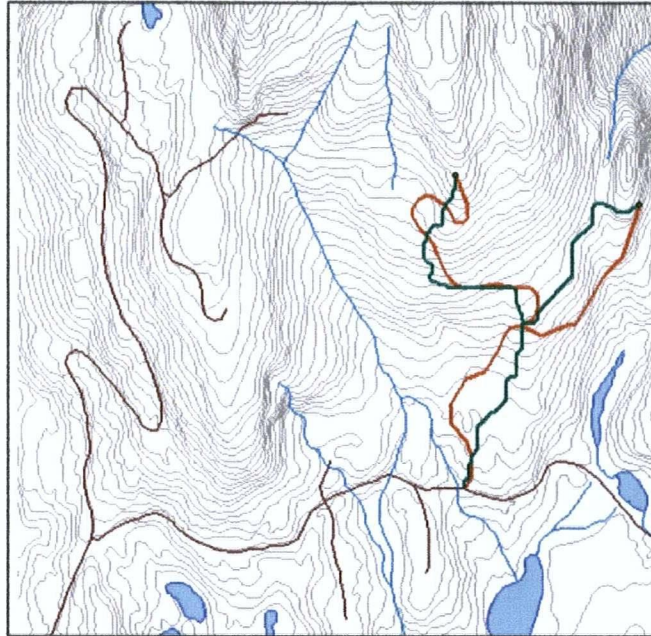
0 87.5 175 350 525 700 Meters

Ionut Aron, MF Candidate
UBC, July 2001

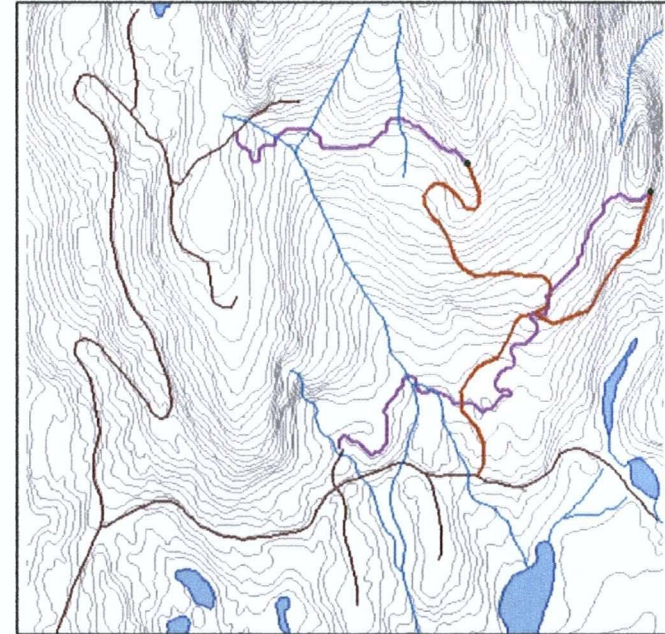
Fig.13 Map of road H90

Preliminary Road Layout for MKRF H110 - Study 10

Neural Network Model



Slope - Curvature Model



Legend:

Roads

Existing Road Network

Contour Lines

5m interval

Streams

Lakes



H110 Original

Length = 1150.70 m

Model4 (NN, noHF, VF=1.0)

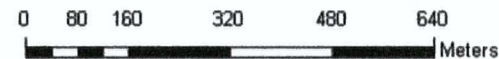
Length = 1004.27 m

Model9 (SC, noHF, VF=1.0)

Length = 1459.39 m



Projection UTM, Zone 10
Datum: NAD83 Canada



Ionut Aron, MF Candidate
UBC, June 2001

Fig. 14 Map of road H110

Preliminary Road Layout for MKRF A42

Study 1

Study 2

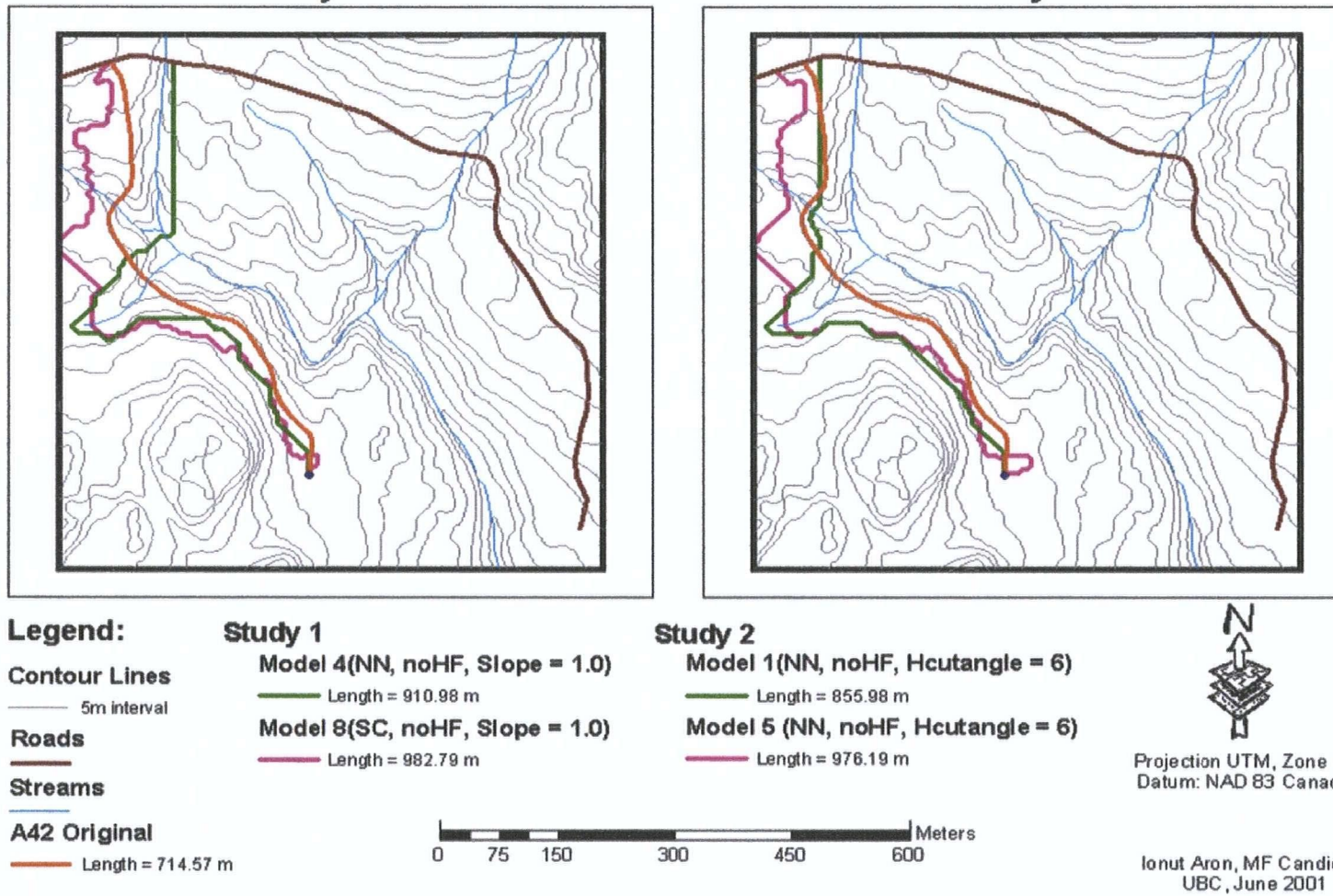


Fig15 Map of road A42

Preliminary Road Layout for MKRF L00

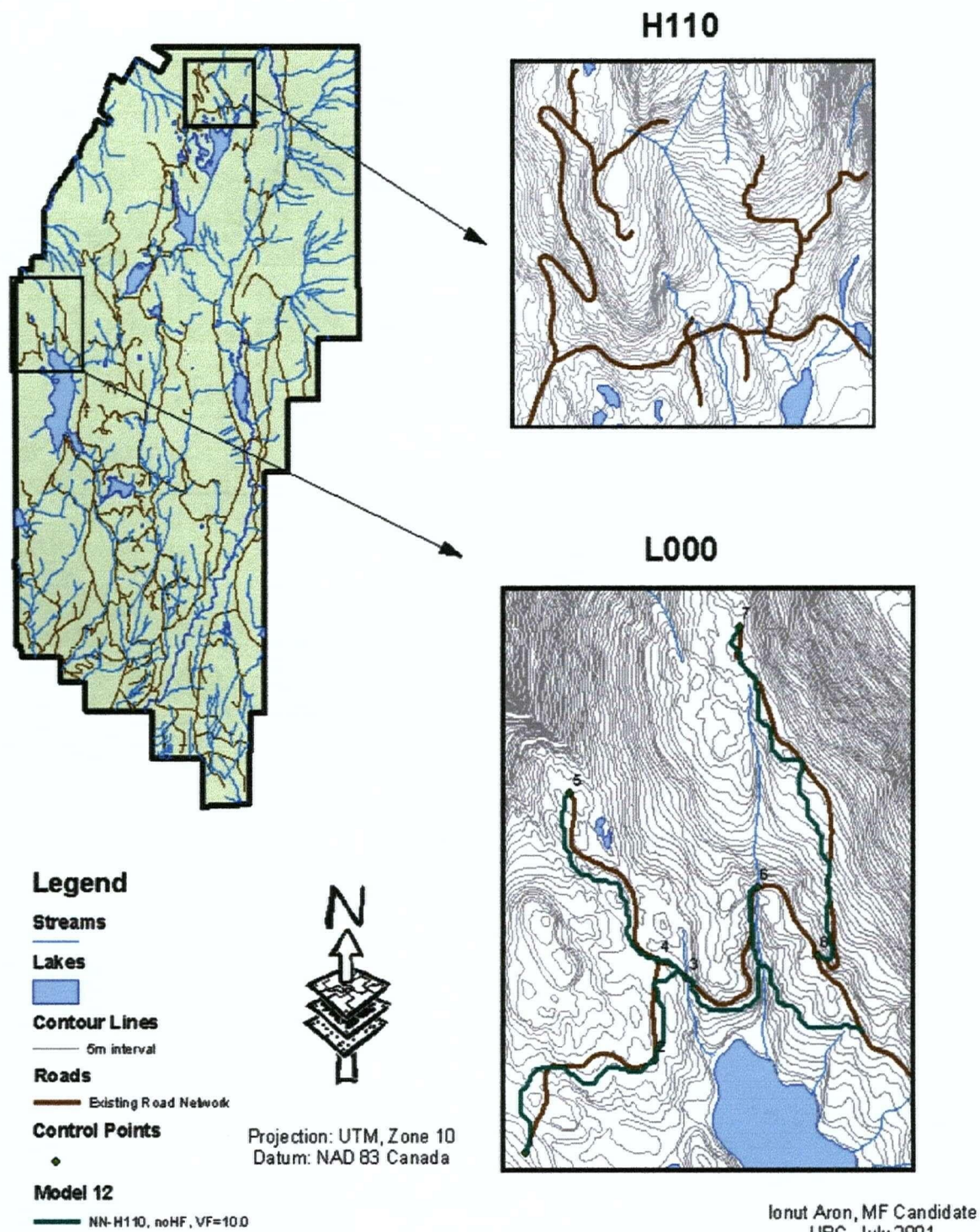

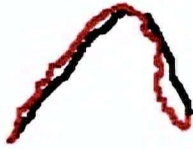




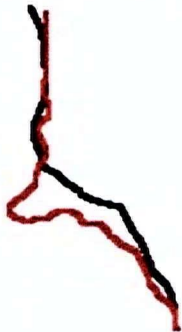
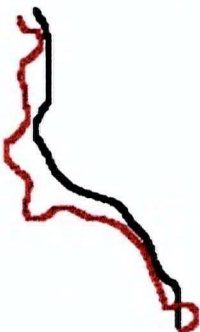
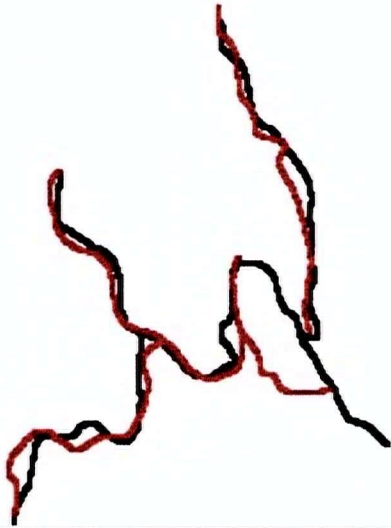



Fig16 Map of road L00

Appendix 2 – Spatial autocorrelation

Table 8 Spatial autocorrelation comparison table for the 5 roads modeled

Road	Original – Neural Network	Original – Slope Curvature
H50		
	C = 0.40	C = 0.12
H90		
	C = 0.41	C = 0.10
H110		
	C = 0.30	C = 0.08

Road	Original – Neural Network	Original – Slope Curvature
A42		
	$C = 0.29$	$C = 0.08$
L00		
	$C = 0.28$	$C = 0.04$

Appendix 3: Soil bearing capacity

The ultimate soil bearing capacity (q_f) is defined as the least pressure which would cause shear failure of the supporting soil immediately below and adjacent to a foundation (R.F. Craig, 1992).

The ultimate soil bearing capacity can be express by Terzaghi' s formula:

$$q_f = \frac{1}{2} \gamma B N_\gamma + c N_c + D N_q \quad [5]$$

Where:

γ - unit weight of a soil and represents the ratio between total weight and total volume

B – width of footing (B = 0.80m)

c – sheer strength parameter

D – depth of footing (D = 0.10m)

N_γ , N_c , N_q - bearing capacity factors

The values for γ and c have been obtained from Level I Stability Analysis Documentation by Hammond et al (1992) and the values for N_γ , N_c , N_q have been extrapolated from a graph from Craig (1992), pp 305.

Appendix 4: ArcINFO functions and procedures

1. SLOPE - identifies the rate of maximum change in z value from each cell.

SLOPE(<grid>, {DEGREE | PERCENTRISE})

SLOPE(<grid>, <z_factor>, {DEGREE | PERCENTRISE})

Arguments

<grid> - any valid combination of grids, scalars, numbers, operators and functions that produces an output grid.

{DEGREE | PERCENTRISE} - keywords specifying the units in which the value of slope will be expressed.

DEGREE - with this option the inclination of slope will be calculated in degrees.

PERCENTRISE - keyword to output the percent rise, also referred to as the percent slope.

<z_factor>- the number of ground x, y UNITS in one surface ZUNIT. The <grid> ZUNITS are multiplied by the specified <z_factor> to adjust the output grid ZUNITS to another unit of measure. The default value of the <z_factor> is 1. Higher z values will result in a more exaggerated relief (surface) and thus in a more extreme shading.

2. BUFFER - creates buffer polygons around specified input coverage features.

BUFFER <in_cover> <out_cover> {buffer_item} {buffer_table} {buffer_distance} {fuzzy_tolerance} {LINE | POLY | POINT | NODE} {ROUND | FLAT} {FULL | LEFT | RIGHT}

Arguments

<in_cover> - the coverage containing features to be buffered.

<out_cover> - the coverage to be created.

{buffer_item} - an item in the feature attribute table of <in_cover> whose value is used as the feature's buffer distance. If a buffer table is used, the {buffer_item} functions as a lookup item in {buffer_table}.

{buffer_table} - an INFO lookup table which lists a buffer distance for each {buffer_item}. A {buffer_table} can be specified only if {buffer_item} is specified. The {buffer_table} contains at least two items.

{buffer_item} - defined the same as {buffer_item} in the <in_cover> feature attribute table. The {buffer_table} must be sorted on this item in ascending order. DIST - the buffer distance for each {buffer_item} value. DIST must be defined as a numeric item (i.e., N, I, F, or B). A lookup table will categorize item values. The use of lookup tables is described in Managing Tabular Data.

{buffer_distance} - the distance used to create buffer zones around <in_cover> features when {buffer_item} or {buffer_item} and {buffer_table} are not specified. The default distance is 0.125 coverage units. This default buffer distance will be applied whenever this argument is skipped with a '#', omitted, or a distance of 0 is specified. The smallest buffer distance that can be computed is 0.00000005 coverage units. Specifying a buffer distance below this threshold will result in an empty output cover. For polygon features, if a negative buffer distance is used, buffers will be generated on the insides of polygons.

{fuzzy_tolerance} - the minimum distance between coordinates in the output coverage. A value of 0 will not be accepted. The default fuzzy tolerance is determined as follows:

- 1) Tolerance values are read from the coverage TOL file if it exists.
- 2) If no TOL file exists and the width of the BND is between 1 and 100, the tolerance is 0.002.
- 3) Otherwise, the tolerance is 1/10,000 of the width or height of the BND, whichever is greater.

{LINE | POLY | POINT | NODE} - the feature class to be buffered:

LINE - arcs will be buffered. This is the default option.

POLY - polygons will be buffered.

POINT - points will be buffered.

NODE - nodes will be buffered.

{ROUND | FLAT} - for lines, the shape of the buffer at the line end points. ROUND will make an end in the shape of a half-circle. FLAT will construct rectangular line endings with the middle of the short side of the rectangle coincident with the end point of the line.

{FULL | LEFT | RIGHT} - for lines, the buffer may be generated on either side (FULL), or a "half buffer" may be generated to the topological left (LEFT) or right side of a line (RIGHT). See notes for a further explanation of the concept of a half buffer.

3. INTERSECT - computes the geometric intersection of two coverages. Only those features in the area common to both coverages will be preserved in the output coverage.

INTERSECT <in_cover> <intersect_cover> <out_cover> {POLY | LINE | POINT}
{fuzzy_tolerance} {JOIN | NOJOIN}

Arguments

<in_cover> - the coverage whose polygon, line or point features will be intersected with the <intersect_cover>.

<intersect_cover> - the overlay coverage containing polygon features.

<out_cover> - the coverage to be created.

{POLY | LINE | POINT} - the <in_cover> feature class to be overlaid and preserved in the <out_cover>.

POLY - polygon-on-polygon overlay. This is the default option.

LINE - line-in-polygon overlay.

POINT - point-in-polygon overlay.

{fuzzy_tolerance} - the minimum distance between coordinates in the output coverage. By default, the minimum fuzzy tolerance value from input and intersect coverages is used. It is determined as follows:

- 1) Tolerance values are read from the existing coverage TOL file.
- 2) If no TOL file exists and the width of the BND is between 1 and 100, the tolerance is 0.002.
- 3) Otherwise, the tolerance is 1/10,000 of the width or height of the BND, whichever is greater.

A fuzzy tolerance value of 0 will not be accepted by INTERSECT.

{JOIN | NOJOIN} - specifies whether all items in both the <in_cover> feature attribute and <intersect_cover> PAT will be joined into the output coverage feature attribute table.

JOIN - all feature attribute items from both coverages will appear in the output coverage feature attribute table. If a duplicate item is encountered, the item in the input coverage will be maintained and the one in the join file will be dropped. This is the default option and is used unless NOJOIN is specified.

NOJOIN - only the feature's internal number (cover#) from the input coverage and the intersect coverage are joined in the output coverage feature attribute table. This option is useful in reducing the size of the output coverage feature attribute table. The relate environment is then used to relate to other tables containing the attributes of output coverage features.

4. PATHDISTANCE - calculates, for each cell, the least-accumulative-cost distance over a cost surface from a source cell or a set of source cells while accounting for surface distance and horizontal and vertical cost factors.

PATHDISTANCE(<source_grid>, {cost_grid}, {surface_grid}, {horiz_factor_grid}, {horiz_factor_parm}, {vert_factor_grid}, {vert_factor_parm}, {o_backlink_grid}, {o_allocate_grid}, {max_distance}, {value_grid})

Arguments

<source_grid> - a grid that identifies those cells from which a least-accumulative-cost distance is calculated to each cell. The input value types can be either integer or floating point.

{cost_grid} - a grid defining the impedance or cost to move planimetrically through each cell. The value at each cell location represents the cost per unit of surface distance for moving through the cell. Each cell location value is multiplied by the cell resolution (while also compensating for diagonal movement) to obtain the total cost of passing through the cell. The values on the {cost_grid} can be integer or floating point, but they cannot be negative (you cannot have a negative cost).

{surface_grid} - a grid identifying the z-values at each cell location. The values are used to calculate the actual surface distance that will be covered when passing between cells.

{horiz_factor_grid} - a grid defining the horizontal direction at each cell. The values on the grid must be integers ranging from 0 to 360 with 0 degrees being north, or towards the top of the screen, and increasing clockwise. Flat areas should be given the value of -1. The values at each location will be used in conjunction with the {horiz_factor_parm} in order to determine the horizontal cost incurred when moving from a cell to its neighbours.

{horiz_factor_parm} - defines the relationship between the horizontal cost factor and the horizontal relative moving angle (HRMA). The input parameter may be one of several keywords (and modifiers) identifying a defined horizontal factor

graph or an ASCII file that creates a custom graph. The graphs are used to identify the horizontal factor that will be used in calculating the total cost for moving into a neighbouring cell. If the parameter is used with any modifier, the string must be double quoted.

The format for the horizontal parameter is:

"keyword {modifier_1 ... modifier_n}"

If no keyword is specified, the default horizontal parameter is BINARY.

In the explanations of the horizontal factor keywords and modifiers, two acronyms are used:

- HF stands for the horizontal factor defining the horizontal difficulty that is encountered in moving from one cell to the next.
- HRMA stands for the horizontal relative moving angle, which identifies the angle between the horizontal direction from a cell and the moving direction.

Horizontal-factor keywords:

BINARY - indicates that if the HRMA is less than the cut angle, the HF is set to the value associated with the zerofactor; otherwise it is infinity.

FORWARD - establishes that only forward movement is allowed. The HRMA must be greater or equal to 0 and less than 90 ($0 \leq \text{HRMA} < 90$). If the HRMA is greater than 0 and less than 45 degrees, the HF for the cell is set to the value associated with the zerofactor. If the HRMA is greater than or equal to 45 degrees, then the side value modifier value is used. The HF for any HRMA equal to or greater than 90 degrees is set to infinity.

LINEAR - defines that the HF is a linear function of the HRMA.

INVERSE_LINEAR - specifies that the HF is an inverse linear function of the HRMA.

TABLE - identifies that an ASCII file will be used to define the horizontal-factor graph used to determine the HFs. The name of the table is entered as a modifier after a blank space following the keyword.

Modifiers to the horizontal keywords:

ZEROFACTOR - establishes the horizontal factor to be used when the HRMA is 0. This factor positions the y-intercept for any of the horizontal-factor functions.

CUTANGLE - defines the HRMA angle beyond which the HF will be set to infinity.

SLOPE - establishes the slope of the straight line used with the LINEAR and INVERSE_LINEAR horizontal-factor keywords. The slope is specified as a fraction of rise over run (i.e., 45 percent slope is 1/45, which is input as 0.02222).

SIDEVALUE - establishes the HF when the HRMA is greater than or equal to 45 degrees and less than 90 degrees for the FORWARD horizontal-factor keyword is specified.

table_name - identifies the name of the ASCII table defining the HF. It is used in conjunction with the TABLE horizontal-factor keyword.

{vert_factor_grid} - a grid defining the z-values for each cell location. The values are used for calculating the slope that is used to identify the vertical factor incurred when moving from one cell to another.

{vert_factor_parm} - defines the relationship between the vertical cost factor and the vertical relative moving angle (VRMA). The input parameter may be one of several keywords (and modifiers) identifying a defined vertical factor graph or an ASCII file creating a custom graph. The graphs are used to identify the vertical factor that will be used in calculating the total cost for moving into a neighbouring cell. If the parameter is used with any modifier, the string must be double quoted.

The format for the vertical parameter is:

"keyword {modifier_1 ... modifier_n}"

If no keyword is specified, the default vertical parameter is SEC.

In the explanations of the vertical-factor keywords and modifiers, two acronyms are used:

- VF stands for the vertical factor defining the vertical difficulty that is encountered moving from one cell to the next.
- VRMA stands for the vertical relative moving angle, which identifies the slope angle between the FROM or processing cell and the TO cell.

Vertical-factor keywords:

BINARY - specifies that if the VRMA is greater than the low-cut angle and less than the high-cut angle then the VF is set to the value associated with the zerofactor; otherwise it is infinity.

LINEAR - indicates that the VF is a linear function of the VRMA.

SYM_LINEAR - defines that the VF is a linear function of the VRMA in either the negative or positive side of the VRMA, respectively, and the two linear functions are symmetrical with respect to the VF (y) axis.

INVERSE_LINEAR - indicates that the VF is an inverse linear function of the VRMA.

SYM_INVERSE_LINEAR - identifies that the VF is an inverse linear function of the VRMA in either the negative or positive side of the VRMA, respectively, and the two linear functions are symmetrical with respect to the VF (y) axis.

COS - defines the VF as the cosine-based function of the VRMA.

SEC - identifies the VF as the secant-based function of the VRMA.

COS-SEC - indicates that the VF is the cosine-based function of the VRMA when the VRMA is negative and the secant-based function of the VRMA when the VRMA is nonnegative.

SEC-COS - specifies that the VF is the secant-based function of the VRMA when the VRMA is negative and the cosine-based function of the VRMA when the VRMA is nonnegative.

TABLE - identifies that an ASCII file will be used to define the VFs. The name of the table is entered as a modifier after a blank space following the keyword.

Modifiers to the vertical-factor keywords:

ZEROFACTOR - establishes the vertical factor to be used when the VRMA is 0.

This factor positions the y-intercept of the specified function. By definition, the zerofactor is not applicable to any of the trigonometric vertical functions (COS, SEC, COS_SEC, or SEC_COS). The y-intercept is defined by these functions.

LCUTANGLE - defines the VRMA angle below which the VF will be set to infinity.

HCUTANGLE - defines the VRMA angle above which the VF will be set to infinity.

SLOPE - establishes the slope of the straight line used with the LINEAR and INVERSE_LINEAR vertical-factor keyword. The slope is specified as a fraction of rise over run (e.g., 45 percent slope is 1/45, which is input as 0.02222).

table_name - identifies the name of the ASCII file defining the VF. It is used in conjunction with the TABLE vertical-factor keyword.

{o_backlink_grid} - the name of the output back-link grid. The back-link grid contains values from 0 through 8, which is a code identifying the direction of the next neighbouring cell (the succeeding cell) when retracing (from the destination to the source) the least-accumulative-cost path from a source cell to a destination.

If the path is to pass into the right neighbour, the cell will be assigned the value '1', '2' for the lower-right diagonal cell and continuing clockwise. The value '0' is reserved for source cells.

{o_allocate_grid} - the name of the output cost-allocation grid. The output grid defines for each cell, the zone on the <source_grid> that achieves the minimum-cost distance or the least-accumulative cost in order to reach the cell.

{max_distance} - defines the threshold that the accumulative-cost-distance values cannot exceed. If an accumulative-cost-distance value exceeds the

{max_distance}, the output value for the cell will be NODATA. The {max_distance} defines the extent to which the accumulative-cost distances are computed.

{value_grid} - an optional input grid that identifies the zone value that should be used for each cell on the <source_grid>. The value defined by the {value_grid} for each source cell will be assigned to all cells that are allocated to the source cell location in terms of the minimum-cost distance. The default zone value for a source cell is the value on <source_grid>.

This grid is particularly important if the source grid was created by TEST or a Boolean operator that will only output 1s, 0s and NODATA, or if alternative values or zones are to be used instead of the existing ones on the source input

Default Values for Horizontal Factor Modifiers

Keywords	Zerofactor	Cutangle	Slope	Sidevalue
Binary	1.0	45	-	-
Forward	0.5	45 (fixed)	-	1.0
Linear	0.5	181	1/90	-
Inverse_linear	2.0	180	-1/90	-

Default Values for Vertical Factor Modifiers

Keyword	Zerofactor	Lcutangle	Hcutangle	Slope	Cospower
Binary	1.0	-30	30	-	-
Linear 1.0	-90	90	1/90	-	-
Sym_linear	1.0	-90	90	1/90	-
Inv_linear	1.0	-45	45	-1/45	-
Sym_inverse_linear	1.0	-45	45	-1/45	-

Cos	-	-90	90	-	-	1.0
Sec	-	-90	90	-	-	1.0
Cos_sec	-	-90	90	-	1.0	1.0
Sec_cos	-	-90	90	-	1.0	1.0

5. COSTPATH - produces an output grid that records the least-cost path(s) from selected cell(s) in the input <fromcell_grid>, or from interactive selection on the display, to the closest source cell defined within the <accumcost_grid> in terms of cost distance.

COSTPATH(<fromcell_grid>, <accumcost_grid>, <backlink_grid>, {BYCELL | BYZONE | BYLAYER})

COSTPATH(<*>, <accumcost_grid>, <backlink_grid>, {BYCELL | BYLAYER}, {verify_grid})

Arguments:

<fromcell_grid> - a grid that identifies those cells from which the least cost path is determined to the least costly source. The grid consists of cells which are to be considered in the COSTPATH calculations having valid values ('0' is a valid value), and the remaining cells must be assigned to NODATA.

<accumcost_grid> - the name of a cost distance grid to be used to determine the least cost path from the <fromcell_grid> cell locations to a source. The <accumcost_grid> was created with the COSTDISTANCE (or by the COSTALLOCATION or COSTBACKLINK functions) or PATHDISTANCE functions. The <accumcost_grid> stores, for each cell, the minimum accumulative cost distance over a cost surface from each cell to a set of source cells.

<backlink_grid> - the name of a cost back link grid used to determine the path to return to a source via the least-cost path. For each cell in the back link grid, a value identifies the neighbour that is the next cell on the least accumulative cost path from the cell to a single or set of source cells.

{BYCELL | BYZONE | BYLAYER} - a keyword defining the manner in which the values and zones on the <fromcell_grid> or from the interactive input will be interpreted in the cost path calculations.

BYCELL - for each cell with valid values on the <fromcell_grid> or from the interactive input, a least-cost path is determined and saved on the output grid of the COSTPATH function. With the BYCELL keyword, each cell of the <fromcell_grid> input is treated separately, and a least-cost path is determined for each 'from' cell.

BYZONE - for each zone on the <fromcell_grid> or from interactive input, a single least-cost path is determined and saved on the output grid of the COSTPATH function. With the BYZONE keyword, for each zone, the least-cost path begins from the cell with the lowest cost distance weighting in the zone.

BYLAYER - for all cells on the <fromcell_grid> input, the least cost path is derived from the cell with the minimum of the least cost paths to source cells.

<*> - allows for the interactive (graphical) entrance of the from cells. A dialogue is entered and the user is prompted for the points from which to calculate the least cost path. To enter a point, the cursor is positioned over the desired location, and the appropriate button on the mouse is clicked (the appropriate button is specified in the dialogue). If the point is not the one desired, a second button is depressed. To stop entering points, the appropriate button on the mouse specified by the dialogue is depressed. Only the locations of single cells can be chosen with the <*> option when identifying the from cells in which cost

paths are to be calculated. There is a maximum of 100 cells or picks with the cursor.

{verify_grid} - defines the grid whose cell values will be printed for verification when interactively entering the input locations with the <*> argument. Once a cell is chosen (by depressing the appropriate button on the mouse), the value for the cell location on the {value_grid} will be displayed on the screen. If the cell is the one that is desired, another cell can be chosen or the session can be ended. If the value that is printed to the screen is not the cell that is desired, the cell input can be canceled by depressing the appropriate button on the mouse. The value that is printed to the screen is not used in the processing of the COSTPATH function.

AML procedure for side slope calculation

```
/* Calculates the side slope for a cell
/*
/*
/*      !----!----!----!
/*      ! ul ! up ! ur !
/*      !----!----!----!
/*      ! le !cell! ri !
/*      !----!----!----!
/*      ! ll ! lo ! lr !
/*      !----!----!----!
/*
/* Upper left
&if [exists slope_ul -grid] &then
    kill slope_ul all
/* Upper
&if [exists slope_up -grid] &then
    kill slope_up all
/* Upper right
&if [exists slope_ur -grid] &then
    kill slope_ur all
/* Right
&if [exists slope_ri -grid] &then
    kill slope_ri all
/*Lower right
&if [exists slope_lr -grid] &then
    kill slope_lr all
/*Lower
&if [exists slope_lo -grid] &then
    kill slope_lo all
```

```

/* Lower left
&if [exists slope_ll -grid] &then
    kill slope_ll all
/* Left
&if [exists slope_le -grid] &then
    kill slope_le all

DOCELL
slope_ul = (abs(h50dem_int(-1, -1) - h50dem_int(0, 0))) / 1.41 / 5 * 100
slope_up = (abs(h50dem_int(0, -1) - h50dem_int(0, 0))) / 5 * 100
slope_ur = (abs(h50dem_int(1, -1) - h50dem_int(0, 0))) / 1.41 / 5 * 100
slope_ri = (abs(h50dem_int(1, 0) - h50dem_int(0, 0))) / 5 * 100
slope_lr = (abs(h50dem_int(1, 1) - h50dem_int(0, 0))) / 1.41 / 5 * 100
slope_do = (abs(h50dem_int(0, 1) - h50dem_int(0, 0))) / 5 * 100
slope_ll = (abs(h50dem_int(-1, 1) - h50dem_int(0, 0))) / 1.41 / 5 * 100
slope_le = (abs(h50dem_int(-1, 0) - h50dem_int(0, 0))) / 5 * 100
END

```

AML procedure for data collection

```
/*A-----AUTHOR-----
/*
/*Original Coding:   ESRI
/*
/*N-----NAME-----
/*
/*GRIDSPOT70.AML
/*Copyright 1995, Environmental Systems Research Institute, Inc.
/*
/*P-----PURPOSE-----
/*
/* This AML sets an item in a point coverage PAT equal
/* to the value of a grid cell at the corresponding location. It is intended
/* to be similar to the LATTICESPOT command except that it does not
/* perform interpolation. This AML requires ARC/INFO 7.0 or higher.
/*
/*U-----USAGE-----
/*
/* GRIDSPOT <grid_name> <point_cover> {spot_item}
/*
/*V-----VARIABLES-----
/*
/*           Local variables:
/*
/* grid_name Name of the grid to take cell values from
/* ptcov      Name of point coverage to which cell values are added
/* spot_item  Name of PAT item to which cell values are added. If
/*            omitted, the default item name is SPOT.
/* old$messages Save setting of &messages
```

```

/* old$displaySave setting of DISPLAY
/* old$echo  Save setting of &echo
/*
/*          Global variables: NONE
/*
/*C-----CALLS-----
/*
/* None.
/*
/*=====DISCLAIMER=====
/*You may use, copy, modify, merge, distribute, alter, reproduce and/or
/*create derivative works of this AML for your own internal use. All
/*rights not specifically granted herein are reserved to ESRI.
/*
/*THIS AML IS PROVIDED "AS-IS" WITHOUT WARRANTY OF ANY KIND,
EITHER
/*EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
/*WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE,
/*WITH RESPECT TO THE AML.
/*
/*ESRI shall not be liable for any damages under any theory of law
/*related to your use of this AML, even if ESRI is advised of the
/*possibilities of such damage. This AML is not supported by ESRI.
/*=====

&args grid_name ptcov spot_item
&severity &error &routine bailout

/* -----Argument Checking-----

```

```

&if [show program] ne 'ARC' &then
    &return This aml must be run from ARC

&if [null %grid_name%] &then
    &return Usage: GRIDSPOT <grid> <point_coverage> {spot_item}

&if ^ [exists %grid_name% -grid] &then
    &return %grid_name% is not a grid.

&if [null %ptcov%] &then
    &return Usage: GRIDSPOT <grid> <point_coverage> {spot_item}

&if ^ [exists %ptcov% -point] &then
    &return %ptcov% is not a point coverage.

&if [null %spot_item%] &then
    &sv spot_item SPOT

&s old$messages [show &messages]
&messages &off

/* Check whether spot_item exists in the point coverage PAT

&if [iteminfo %ptcov%.pat -info %spot_item% -exists] &then
    &type WARNING: Existing item %spot_item% in %ptcov%.PAT will be
    recalculated.
&else
    additem %ptcov%.pat %ptcov%.pat %spot_item% 4 12 f 3

/* Now go into Arcplot and get the values

```

```

&sv old$display [show display]
display 0
ap
/* Declare and open a cursor to read and write to the PAT

cursor ptcursor declare %ptcov% points rw
cursor ptcursor open

&s old$echo [show &echo]
&echo &off

/* Start a loop to go through the PAT, find the cell value at each
/* point location, and write it to the SPOT item in the PAT.

&do &while %:ptcur.aml$next%
  &s temp [show cellvalue %grid_name% [show select %ptcov% point 1 xy]]
  &if [type %temp%] = 1 &then
    &s :ptcur.%spot_item% -9999
  &else
    &s :ptcur.%spot_item% %temp%
  cursor ptcursor next
&end

q /* quit from arcplot
&call exit
&return /* To calling aml
/*
/* ----- Routine Exit -----
/*
&routine exit
&if [variable old$display] &then

```



```
display %old$display%
&if [variable old$messages] &then
  &messages %old$messages%
&if [variable old$echo] &then
  &echo %old$echo%
&return
/*
/* ----- Routine Bailout -----
/*
&routine bailout
&severity &error &ignore
&call exit
&return; &return &error Bailing out of gridspot.aml
```