

Edith Cowan University
Research Online

Theses : Honours

Theses

1995

Implementation and evaluation of enhanced areal interpolation using MapInfo and MapBasic

Gordon Wragg
Edith Cowan University

Follow this and additional works at: https://ro.ecu.edu.au/theses_hons

 Part of the [Programming Languages and Compilers Commons](#)

Recommended Citation

Wragg, G. (1995). *Implementation and evaluation of enhanced areal interpolation using MapInfo and MapBasic*. https://ro.ecu.edu.au/theses_hons/634

This Thesis is posted at Research Online.
https://ro.ecu.edu.au/theses_hons/634

Edith Cowan University

Copyright Warning

You may print or download ONE copy of this document for the purpose of your own research or study.

The University does not authorize you to copy, communicate or otherwise make available electronically to any other person any copyright material contained on this site.

You are reminded of the following:

- Copyright owners are entitled to take legal action against persons who infringe their copyright.
- A reproduction of material that is protected by copyright may be a copyright infringement. Where the reproduction of such material is done without attribution of authorship, with false attribution of authorship or the authorship is treated in a derogatory manner, this may be a breach of the author's moral rights contained in Part IX of the Copyright Act 1968 (Cth).
- Courts have the power to impose a wide range of civil and criminal sanctions for infringement of copyright, infringement of moral rights and other offences under the Copyright Act 1968 (Cth). Higher penalties may apply, and higher damages may be awarded, for offences and infringements involving the conversion of material into digital or electronic form.

**Implementation and Evaluation of
Enhanced Areal Interpolation using
MapInfo and MapBasic**

**Bachelor of Science (Mathematics) Honours
Thesis**

Gordon Wragg

Student No. 5800652

Edith Cowan University

Faculty of Science, Technology & Engineering

School of Mathematics, Information Technology & Engineering

Department of Mathematics

Supervisors: Dr L Bloom

Dr P Pedler

February 1995

USE OF THESIS

The Use of Thesis statement is not included in this version of the thesis.

Table of Contents

Chapter	Page
Abstract	iv
Declaration	v
1. Introduction	1
1.1. Background and Significance	1
1.2. Notation and Preliminaries	2
1.3. Purpose of Project	4
2. Mathematical Framework	5
2.1. Univariate Concepts	5
2.2. Multivariate Distributions	6
2.3. Some Useful Distributions	8
2.4. Exponential Families of Distributions	11
2.5. Model Fit	15
2.6. EM Algorithm	16
3. Enhanced Areal Interpolation	18
3.1. Areal Weighting Interpolation	18
3.2. Enhanced Areal Interpolation	20
3.3. Model Fit	26
4. Implementation	28
4.1. Initial Development	28
4.2. Implementation	35
4.3. Software and Hardware	36
4.4. Design	37
4.5. Execution	38
5. Evaluation	46
5.1. Example 1	46
5.2. Example 2	47
5.3. Example 3	48
5.4. Comments	49

Chapter	Page
6. Brief Summary	51
6.1 Implementation	51
6.2 Usefulness	51
6.3 Future Directions	52
7. References	53
Appendices	55
Appendix 1: Spreadsheet Model	55
Appendix 2: MapBasic Application Design	58

Abstract

Many researchers today have a need to analyse data in a spatial context. An inherent problem is the mismatch of boundaries between the geographic regions for which data is collected and those regions for which the data is required.

Often the solution is to interpolate data from one set of regions to another.


This project examines and implements a method of areal interpolation that enables the user to use extra information in areal interpolation to increase the "intelligence" of the process.

This method of *Enhanced Areal Interpolation* uses a conditional Poisson distribution and the EM algorithm to provide estimated values of a variable. *Enhanced Areal Interpolation* assumes that data is available for a set of *source* regions, and is required for a set of *target* regions. Extra information available about the target regions provides an improved fit of the estimates compared to *Areal Weighting Interpolation* which uses area proportionality to distribute the data.

The theory and concepts are illustrated with an example and implemented using the software packages *MapInfo version 3 for Windows* and *MapBasic version 3 for Windows*.

Declaration

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

Signature 

Date 2/2/95

1. Introduction

1.1 Background and Significance

Data interpolation is a problem that is inherent in any field of research involving spatial data analysis. In such analysis a researcher often encounters difficulties in using data available for regions that are not those regions that the researcher wishes to use. In Australia today there are large amounts of data available to the researcher on a geographical basis. Unfortunately much of this has been collected for relatively large areas, or by organisations for specific internal or reporting purposes. One example is the data collected by the Department of Social Security (DSS) on numbers of persons claiming particular benefits. The data in this case is only available by DSS office zones or postcode. For the researcher trying to identify small regions of disadvantaged persons this data is not appropriate. This is a problem of small area synthesis.

Many researchers collect or purchase data on a geographical basis and the geographical areas used will have a great influence on the outcome of the research. For example, it may be possible to obtain data on single parent families, but only for Local Government Areas (LGAs). The user who is interested in the distribution of these families, but for smaller areas has few options. One option is to conduct a survey, which will provide the information required, but at high cost, and is time consuming. There are also other factors apart from cost and time. Some of these are:

- comparability of the data to other collections;
- possible need for a survey to be ongoing for a time series analysis;
- need to take into account bias and errors;
- obtaining a large enough sample.

Many researchers will try to find other options. One option is to use point data, and aggregate those points that fall within the target region. This option does not take into

account the areal nature of the data. Another option is to allocate data from a source region to a target region on a closest fit basis. This method relies heavily on the knowledge and experience of the analyst. A variation of this method is used by CDATE 91 whereby a region is selected as being within a boundary if the centroid of the region falls within the boundary. Lastly, there is the option of allocating data from source regions to target regions based on the proportion of area overlap. This method assumes that the data is uniformly distributed. The last method is very likely to be the method found in a Geographic Information System (GIS) package, and is the method used by MapInfo version 3.0. It was discussed by Markoff and Shapiro (1973), and Goodchild and Lam (1980) described the process as *Areal Interpolation*. We shall refer to it here as *Areal Weighting Interpolation*.

Flowerdew et al (1991) describe a statistical method of areal interpolation that uses ancillary data. This method relies upon the ancillary data being available to the analyst to provide additional information in the interpolation process, and so enhance the precision of such interpolations. We shall refer to this method as *Enhanced Areal Interpolation*. In their paper, they outline the use of an iterative algorithm known as the EM algorithm. This is a statistical technique to fit missing data and is presented in general form by Dempster, Laird and Rubin (1977). Dempster, Laird and Rubin (1977) also outline a general process using the software packages GLIM and ArcInfo to implement their procedure. Their implementation of the Enhanced Areal Interpolation method involved solving problems of interfacing these quite different software packages. The difficulty was only solved by being able to obtain access to the object code of the software (Kehris, 1989), which makes it difficult for the average user to make use of this method.

1.2 Notation and Preliminaries

The terms and notation used here are similar to those used by Flowerdew et al (1991), which is the main reference for the project. The process is to interpolate data for a set

of regions called the target regions, from the data associated with another set of regions called the source regions.

Let the variable of interest be Y. The data on this variable is available for a set of source regions S, but is needed for a set of target regions T.

The source regions and target regions are separate partitions of the same space, and can be represented by the column vectors

$$S = [s_1, s_2, \dots, s_m]'$$

$$T = [t_1, t_2, \dots, t_n]'$$

where there are m source regions and n target regions.

These two sets of regions will form regions of intersection. Let the areas of the intersection regions be represented by the matrix

$$A_{ij} = [a_{ij}] \quad i = 1, 2, \dots, m; \quad j = 1, 2, \dots, n,$$

where a_{ij} is the area of the intersection region $s_i t_j$; $i = 1, 2, \dots, m$; $j = 1, 2, \dots, n$. The areas of each source region and each target region can be found from this matrix using row and column sums. The area of the i th source region s_i is then

$$a_{i.} = \sum_{j=1}^n a_{ij},$$

and that of the j th target region t_j is then

$$a_{.j} = \sum_{i=1}^m a_{ij},$$

where the dot (.) indicates summation.

These areas can be represented as column vectors by

$$A_s = [a_{1.}, a_{2.}, \dots, a_{m.}] \quad \text{for source regions, and}$$

$$A_t = [a_{.1}, a_{.2}, \dots, a_{.n}] \quad \text{for target regions.}$$

The value of the variable Y for a source region s_i is $y_{i\bullet}$ (known) and for a target region t_j is $y_{\bullet j}$ (unknown). These can be expressed as vectors

$$y_s = [y_{1\bullet}, y_{2\bullet}, \dots, y_{m\bullet}]' \quad \text{and} \quad y_t = [y_{\bullet 1}, y_{\bullet 2}, \dots, y_{\bullet n}]'$$

Where a source region s_i and target region t_j intersect, the value of Y for the intersection is denoted by y_{ij} . In this project only extensive variables as described by Flowerdew et al (1991) will be considered. A variable is extensive when its values can be aggregated by summation. Some examples are number of people, total expenditure, and number of electors. Additional information about the target regions will be used in the interpolation process. This will be specified by the value of at least one ancillary variable X_t on the target regions which can be represented as the vector

$$x_t = [x_{t1}, x_{t2}, \dots, x_{tn}]'$$

1.3 Purpose of Project

The first major objective of this project is to examine the statistical theory that provides the basis for *Enhanced Areal Interpolation*. The relevant theory includes the general concepts and background theory of conditional distributions, exponential families and maximum likelihood. The theory covers particularly the application of these concepts using the Poisson distribution.

Flowerdew et al (1991) noted that they implemented the enhanced areal interpolation using two software packages. The GIS was ArcInfo, and the maximum likelihood estimates were calculated in the statistical package GLIM. They noted that the main difficulty was in the interface between the two, and it was managed only after modifying the code of the two packages. This option is beyond the abilities of the average user. The second major objective of this project is to implement the enhanced areal interpolation in a more robust manner, which means using software and computer hardware that is readily available for many users, and setting the implementation up in such a way that using it requires little additional knowledge on the part of the user.

2. Mathematical Framework

There are a number of statistical concepts and results needed for the algorithms used in this project and these are stated here. The concepts are for the most part taken from Freund (1992), with additional parts from McCullagh & Nelder (1988) and Johnson & Wichern (1992).

2.1 Univariate Concepts

2.1.1 Definition: If X is a discrete random variable, then $f(x) = P(X = x)$ is the *probability distribution* of X .

2.1.2 Definition: If X is a continuous random variable, $f(x)$ is the *probability density function* (pdf) of X , and satisfies

$$P(a \leq X \leq b) = \int_a^b f(x) dx$$

$a, b \in \mathfrak{R} ; a \leq b.$

2.1.3 Definition: The *expected value* or *mean* of a function $u(X)$ of a random variable X is defined as

$$E[u(X)] = \begin{cases} \sum u(x) \cdot f(x) & \text{if } X \text{ is discrete} \\ \int_{-\infty}^{\infty} u(x) \cdot f(x) dx & \text{if } X \text{ is continuous.} \end{cases}$$

When $u(X) = X$, the expected value is the *mean* or *expected value* of X and we write $\mu = E[X]$. The following concepts and results will use $u(X) = X$.

2.1.4 Definition: The *variance* of X is a measure of dispersion and is defined by

$$\sigma^2 = E[(X - \mu)^2]$$

which reduces to

$$\sigma^2 = E[X^2] - \mu^2.$$

2.2 Multivariate Distributions

2.2.1 Definition: The *joint probability distribution* of n discrete random variables X_1, X_2, \dots, X_n is given by

$$f(x_1, x_2, \dots, x_n) = P(X_1 = x_1, X_2 = x_2, \dots, X_n = x_n).$$

The *joint probability density function* of n continuous random variables is given by

$$P((X_1, X_2, \dots, X_n) \in A) = \int \int \dots \int_A f(x_1, x_2, \dots, x_n) dx_1 dx_2 \dots dx_n$$

where A is a region in n space.

2.2.2 Definition: For n discrete random variables X_1, X_2, \dots, X_n , with joint probability distribution $f(x_1, x_2, \dots, x_n)$ at (x_1, x_2, \dots, x_n) the *marginal distribution* of X_i at x_i is given by

$$g(x_i) = \sum_{x_1} \dots \sum_{x_{i-1}} \sum_{x_{i+1}} \dots \sum_{x_n} f(x_1, x_2, \dots, x_n).$$

For n continuous random variables, the *marginal density function* of X_i is given by

$$g(x_i) = \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} f(x_1, x_2, \dots, x_n) dx_1 \dots dx_{i-1} dx_{i+1} \dots dx_n.$$

2.2.3 Definition: For the n random variables X_1, X_2, \dots, X_n at x_1, x_2, \dots, x_n , the *conditional density* of $X_1, X_2, \dots, X_{i-1}, X_{i+1}, \dots, X_n$ given $X_i = x_i$, is given by

$$f(x_1, x_2, \dots, x_n | x_i) = \frac{f(x_1, x_2, \dots, x_n)}{g(x_i)}, \quad g(x_i) \neq 0.$$

where $g(x_i)$ is the marginal pdf of X_i .

2.2.4 Definition: The n random variables X_1, X_2, \dots, X_n , with joint probability distribution(density) function $f(x_1, x_2, \dots, x_n)$ at x_1, x_2, \dots, x_n , are *statistically independent* if and only if

$$f(x_1, x_2, \dots, x_n) = f_1(x_1) f_2(x_2) \dots f_n(x_n),$$

where $f_i(x_i)$ is the marginal distribution(density) of X_i at x_i for $i = 1, 2, \dots, n$.

If the random variables are identically distributed then

$$f_i(x_i) = f(x_i) \quad \text{for } i = 1, 2, \dots, n.$$

2.2.5 Definition: If X and Y are random variables, and $f(x|y)$ is the value of the conditional probability distribution(density) of X given $Y = y$, the *conditional expectation* of a function $u(X)$ of X , given $Y = y$ is

$$E[u(X)|y] = \begin{cases} \sum u(x) f(x|y) & \text{if } X \text{ is discrete, and} \\ \int_{-\infty}^{\infty} u(x) f(x|y) dx & \text{if } X \text{ is continuous} \end{cases}$$

2.2.6 Maximum Likelihood

The method of maximum likelihood is a general estimation method. The essential feature of the maximum likelihood method is that an estimate of an unknown population parameter is chosen using the sample values in such a way as to maximise the probability of obtaining the observed sample.

If x_1, x_2, \dots, x_n are values of a random sample from a population with the parameter θ , then the likelihood function of the sample is denoted by $L(\theta)$ and is given by

$$L(\theta) = f(x_1, x_2, \dots, x_n; \theta).$$

The function f is the joint probability distribution or density of the random variables X_1, X_2, \dots, X_n at the sample point. If the n random variables are independent this reduces to

$$L(\theta) = f_1(x_1; \theta) f_2(x_2; \theta) \dots f_n(x_n; \theta) = \prod_i f_i(x_i; \theta) \quad (2.2.6)$$

where $f_i(x_i; \theta)$ is the marginal distribution (density) of X_i at x_i for $i = 1, 2, \dots, n$.

The value $\hat{\theta}$ of the parameter θ which maximises this function at the sample point x_1, x_2, \dots, x_n , is the *maximum likelihood estimate* of θ . This value of θ will also maximise the log likelihood function, namely

$$\ell(\theta) = \ell_n[L(\theta)] = \sum_i \ell_n[f_i(x_i; \theta)].$$

As $\theta \in \mathbf{R}$, we can use differentiation to find the value of θ that maximises $\ell(\theta)$. Thus the value $\hat{\theta}$ is a solution to the equation

$$\frac{dL(\theta)}{d\theta} = 0 \Leftrightarrow \frac{d\ell(\theta)}{d\theta} = 0.$$

2.3 Some Useful Distributions

2.3.1 Uniform Distribution

A random variable X is said to have a discrete uniform distribution if and only if its probability distribution is

$$f(x) = \frac{1}{n} \quad x = x_1, x_2, \dots, x_n$$

where $x_i \neq x_j$ when $i \neq j$. The discrete uniform distribution has mean and variance

$$\mu = \sum_{i=1}^n x_i \cdot \frac{1}{n}, \quad \sigma^2 = \sum_{i=1}^n (x_i - \mu)^2 \cdot \frac{1}{n}.$$

A random variable X has continuous uniform density if and only if its pdf is given by

$$f(x) = \begin{cases} \frac{1}{b-a}, & a < x < b \\ 0, & \text{otherwise} \end{cases}$$

where a, b are real numbers. The continuous uniform density has mean and variance of

$$\mu = \frac{a+b}{2}, \quad \sigma^2 = \frac{(b-a)^2}{12}$$

2.3.2 Binomial Distribution

Consider a random experiment with two outcomes, 'success' and 'failure', and constant probability p of a success. Then the number of successes X in n independent repetitions of the experiment has a binomial distribution with parameters n , p , and is a binomial random variable, with probability distribution given by

$$b(x; n, p) = \binom{n}{x} p^x (1-p)^{n-x} \quad \text{for } x = 0, 1, \dots, n.$$

The mean and variance of the binomial distribution are

$$\mu = np, \quad \sigma^2 = np(1-p).$$

2.3.3 Poisson Distribution

The Poisson distribution is a limiting form of the binomial distribution as n tends to infinity and p tends to zero such that the binomial mean np is a constant λ .

A random variable X has a Poisson distribution with parameter λ if its probability distribution is given by

$$p(x; \lambda) = \frac{\lambda^x e^{-\lambda}}{x!}; \quad x = 0, 1, \dots; \quad \lambda > 0.$$

The Poisson distribution has mean and variance given by

$$\mu = \sigma^2 = \lambda.$$

The Poisson distribution is used for modelling random events in space or time.

Note that if the random variables X_1, X_2, \dots, X_n are independent Poisson variables with parameters $\lambda_1, \lambda_2, \dots, \lambda_n$ respectively, then

$$Y = \sum X_i$$

is also a Poisson random variable with parameter

$$\lambda = \sum \lambda_i.$$

2.3.4 Conditional Poisson Distribution

Suppose X_1, X_2, \dots, X_n are independent Poisson variables. Then the conditional distribution of X_i given $\sum X_i = Y = y$ is given by (2.2.3) as

$$f(x_i|y) = \frac{f(x_i, y)}{g(y)}$$

where

$$g(y) = \frac{e^{-\sum \lambda_i} (\sum \lambda_i)^y}{y!}.$$

From (2.2.1) the joint probability distribution is

$$\begin{aligned} f(x_i, y) &= P(X = x_i, Y = y) \\ &= P(X = x_i, \sum^* X_i = y - x_i) \\ &= \left[\frac{e^{-\lambda_i} (\lambda_i)^{x_i}}{x_i!} \right] \left[\frac{e^{-\sum^* \lambda_i} (\sum^* \lambda_i)^{y-x_i}}{(y-x_i)!} \right] \end{aligned}$$

where

$$\sum^* X_i = Y - X_i, \quad \sum^* \lambda_i = \sum \lambda_i - \lambda_i.$$

Therefore

$$\begin{aligned}
 f(x_i|y) &= \frac{\left[\frac{e^{-\lambda_i} (\lambda_i)^{x_i}}{x_i!} \right] \left[\frac{e^{-\sum \lambda_i} (\sum \lambda_i)^{y-x_i}}{(y-x_i)!} \right]}{\frac{e^{-\sum \lambda_i} (\sum \lambda_i)^y}{y!}} \\
 &= \binom{y}{x_i} \frac{(\lambda_i)^{x_i} (\sum \lambda_i)^{y-x_i}}{(\sum \lambda_i)^{x_i} (\sum \lambda_i)^{y-x_i}} \\
 &= \binom{y}{x_i} \left(\frac{\lambda_i}{\sum \lambda_i} \right)^{x_i} \left(1 - \frac{\lambda_i}{\sum \lambda_i} \right)^{y-x_i}
 \end{aligned}$$

It follows that the conditional distribution of X_i given $Y = y$ is binomial with parameters

$$n = y, \quad p = \frac{\lambda_i}{\sum \lambda_i} \quad (2.3.1)$$

and conditional mean

$$\mu = np = \frac{\lambda_i}{\sum \lambda_i} y.$$

This result will be used later in equation (3.2.4).

2.4 Exponential Families of Distributions

The binomial and Poisson distributions are both members of a group of distributions known as Exponential Families. The pdf of an exponential family member can be written in the form

$$f(x; \phi) = \frac{b(x)e^{\phi t(x)}}{a(\phi)}$$

where

$t(x)$ is a function t of the observation x ,

ϕ is the model parameter,

$b(x)$ is a function b of the observation,

$a(\phi)$ is a function of the parameter ϕ and is a normalising factor.

In this study we consider one parameter exponential families.

2.4.1 Poisson; mean = λ .

The Poisson distribution can be written as

$$f(x; \lambda) = \frac{\left(\frac{1}{x!}\right) e^{(\ln \lambda)x}}{e^\lambda}; \quad x = 0, 1, \dots; \lambda > 0,$$

which is an exponential family member with

$$b(x) = \frac{1}{x!}, \quad a(\phi) = e^\lambda, \quad \phi = \ln \lambda, \quad t(x) = x.$$

2.4.2 Normal; mean = μ , variance = λ .

The pdf of the normal distribution is given by

$$f(x; \mu, \sigma^2) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

which requires two parameters. By specifying the value of the variance $\sigma^2 = 1$ say, this distribution becomes a one parameter pdf where

$$\begin{aligned} f(x; \mu) &= \frac{1}{\sqrt{2\pi}} e^{-1/2(x-\mu)^2} \\ &= \frac{1}{\sqrt{2\pi}} e^{-1/2(x^2 - 2x\mu + \mu^2)} \\ &= \frac{1}{\sqrt{2\pi}} \cdot \frac{(e^{-1/2 x^2}) e^{\mu x}}{e^{1/2 \mu^2}} \end{aligned}$$

which is an exponential family member with

$$b(x) = \frac{e^{-x^2/2}}{\sqrt{2\pi}}, \quad a(\phi) = e^{\mu^2/2}, \quad \phi = \mu, \quad t(x) = x$$

2.4.3 Binomial; known n, unknown p.

The binomial distribution with known n and unknown p has one parameter namely p.

Its pdf is given by

$$\begin{aligned} f(x; p) &= \binom{n}{x} p^x (1-p)^{n-x} \\ &= \frac{\binom{n}{x} \left(\frac{p}{1-p}\right)^x}{(1-p)^{-n}} \\ &= \frac{\binom{n}{x} e^{x \ln\left(\frac{p}{1-p}\right)}}{(1-p)^{-n}} \end{aligned}$$

which is an exponential family member with

$$b(x) = \binom{n}{x}, \quad a(\phi) = (1-p)^{-n}, \quad \phi = \ln\left(\frac{p}{1-p}\right), \quad t(x) = x.$$

2.4.4 Properties of Exponential Families

Exponential families have a number of valuable properties (see McCullagh & Nelder 1988).

Suppose X_1, X_2, \dots, X_n are independent, identically distributed random variables whose distribution is a member of an exponential family. From 2.2.1 and 2.4.1 above, the joint probability density function is

$$\begin{aligned} f(x_1, x_2, \dots, x_n; \phi) &= \prod_i f(x_i; \phi) \\ &= \frac{e^{\phi \sum_i t(x_i)} \prod_i b(x_i)}{[a(\phi)]^n}. \end{aligned}$$

This is a member of the exponential family with

$$b(x) = \prod_i b(x_i), \quad a(\phi) = [a(\phi)]^n, \quad \phi = \phi, \quad t(x) = \sum_i t(x_i).$$

Furthermore, the sum $\sum_i t(x_i)$ is a sufficient statistic for estimating the parameter ϕ .

For example if the random variables were normally distributed as per 2.4.2 above then the sum $\sum x_i$ contains all the information in the sample x_1, x_2, \dots, x_n for finding the estimate $\hat{\mu}$ of the mean μ . If X_i is written as a column vector \bar{x} the joint pdf becomes

$$f(\bar{x}; \phi) = \frac{b(\bar{x})e^{\phi t(\bar{x})}}{a(\phi)}.$$

Hence the joint pdf of an exponential family is also an exponential family and the structure of an exponential family remains invariant under sampling. This result along with (2.2.6) gives the likelihood function

$$L(\phi; \mathbf{x}) = \frac{e^{\phi t(\mathbf{x})} b(\mathbf{x})}{a(\phi)}.$$

The log likelihood function is then given by

$$\ell(\phi; \mathbf{x}) = \phi t(\mathbf{x}) + \ln b(\mathbf{x}) - \ln a(\phi),$$

This can be maximised by differentiating with respect to ϕ and setting to zero which yields

$$\frac{\partial}{\partial \phi} \ell(\phi; \mathbf{x}) = t(\mathbf{x}) - \frac{a'(\phi)}{a(\phi)} = 0$$

and so is a maximum when

$$\frac{a'(\phi)}{a(\phi)} = t(\mathbf{x}).$$

This then allows the maximum likelihood estimator to be readily found when using an exponential family.

2.5 Model Fit

A measure of how well a member of an exponential family models a given set of data is provided by the deviance (McCullagh & Nelder 1988). This is a measure of discrepancy between the data values and the modelled values and is a function of the data values only. The value of the deviance allows the appropriateness of different methods of areal interpolation to be compared.

For a random variable X , the log likelihood as a function of the mean value parameter μ is

$$\ell(\mu; x) = \ln f(x; \mu)$$

The maximum value of the log likelihood function is then $\ell(\hat{\mu}; x)$, which depends only on the data and not on the parameters. Deviance is defined (McCullagh & Nelder 1985, p17) as

$$D(\hat{\mu}; x) = -2[\ell(\hat{\mu}; x) - \ell(x; x)]$$

For independent Poisson random variables X_1, X_2, \dots, X_n , with means $\mu_1, \mu_2, \dots, \mu_n$ respectively, the joint probability distribution of X_1, X_2, \dots, X_n is given by

$$f(x; \mu) = \prod_i f_i(x_i; \mu_i).$$

The log likelihood, as a function of the model parameter $\mu = (\mu_1, \mu_2, \dots, \mu_n)'$, is then

$$\begin{aligned} \ell(\mu; x) &= \ln f(x; \mu) \\ &= \ln \left[\prod_i \frac{e^{-\mu_i} \mu_i^{x_i}}{x_i!} \right] \\ &= \sum_i [-\mu_i + x_i \ln \mu_i - \ln(x_i!)] \end{aligned}$$

with maximum value when $\mu_i = \hat{\mu}_i$ of

$$\ell(\hat{\mu}; x) = \sum_i [-\hat{\mu}_i + x_i \ln \hat{\mu}_i - \ln(x_i!)]$$

Hence the deviance becomes

$$\begin{aligned}
 D(\hat{\mu}; x) &= -2[\ell(\hat{\mu}; x) - \ell(x; x)] \\
 &= -2 \sum_i [-\hat{\mu}_i + x_i \ln \hat{\mu}_i - \ln(x_i!) + x_i - x_i \ln x_i + \ln(x_i!)] \\
 &= 2 \sum_i [\hat{\mu}_i - x_i + x_i (\ln x_i - \ln \hat{\mu}_i)] \\
 &= 2 \sum_i \left[x_i \ln \left(\frac{x_i}{\hat{\mu}_i} \right) - (x_i - \hat{\mu}_i) \right]. \quad (2.5.1)
 \end{aligned}$$

For the Poisson distribution, the second term usually sums to zero (McCullagh & Nelder 1988).

2.6 EM Algorithm

The EM Algorithm is an iterative process comprising two steps, the expectation or E step and the maximisation or M step. This algorithm is given comprehensive treatment in Dempster, Laird and Rubin (1977), where it is presented in a general form. It is difficult to provide numerical steps for the algorithm (Dempster, Laird & Rubin 1977), as there are many and varied situations where it can be used, and the actual steps to be undertaken rely upon the specific situation.

The E step replaces all missing data values by their conditional expectations given the observed data and estimated values of all population parameters. The M step maximizes the log likelihood function using the now complete data to obtain new estimates for values of the population parameters. These new population parameter values are then used in the E step. Both steps are iterated, continuing until suitable convergence occurs..

E step

The E step is sometimes referred to as a prediction step (Johnson & Wichern, 1982), since a prediction or estimate is made for missing data, given some estimate of the unknown population parameter.

The E step uses estimates of the population parameters from the M step. For the first E step there has to be some starting value for these parameters. The estimation process will depend on the model being used.

For example in a one parameter normal model as in 2.4.2, an initial estimate for missing values could be the mean μ of the data available. The E step uses the M step estimate for further iterations.

For a single parameter exponential family this step will estimate $t(x)$ by

$$t^{(p)} = E[t(x)|x; \phi^{(p)}]$$

where $t^{(p)}$ is the p th iteration estimate for the data values.

M step

The maximum likelihood estimator ϕ is now found from the complete data set. The estimated values from the E step are treated as observed data, and so the value of the maximum likelihood estimator is estimated by

$$E[t(x)|\phi] = t^{(p)},$$

which will give the $(p+1)$ th estimate of ϕ . This simple form of the equation for estimating ϕ is made possible only because of the form of the exponential family and the availability of a complete data set.

3. Enhanced Areal Interpolation

3.1 Areal Weighting Interpolation

The areal weighting interpolation method introduced by Goodchild and Lam (1980) assumes that the variable of interest is uniformly distributed within each source region, and hence any sub-region will have a value proportional to the fraction of the area that is within the sub-region. For example, if a region with a population of n persons is divided into p divisions of equal area, the uniform distribution specifies that each sub-division will contain n/p persons.

This is a process of weighting the distribution using area. If the matrix $W = [w_{ij}]$, $i = 1, 2, \dots, m$; $j = 1, 2, \dots, n$ is a matrix of weights based on the proportion of intersection region area to source region area, then the elements of W can be found by

$$w_{ij} = \frac{\text{Intersection Area}}{\text{Source Area}} = \frac{a_{ij}}{\sum_j a_{ij}} \quad (3.1.1)$$

The data for the variable Y is known for the source regions. The problem is to interpolate values of Y for the target regions. These estimates are denoted by

$$y_t = [\hat{y}_1, \hat{y}_2, \dots, \hat{y}_n]'$$

We can then write

$$\begin{aligned} \hat{y}' &= y'_s W \\ &= [y_1, y_2, \dots, y_m] \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{1.} & a_{1.} & \dots & a_{1.} \\ a_{21} & a_{22} & \dots & a_{2n} \\ a_{2.} & a_{2.} & \dots & a_{2.} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m1} & \dots & a_{mn} \\ a_{m.} & a_{m.} & \dots & a_{m.} \end{bmatrix} \\ &= \left[\frac{y_1 a_{11}}{a_{1.}} + \frac{y_2 a_{21}}{a_{2.}} + \dots + \frac{y_m a_{m1}}{a_{m.}}, \dots, \frac{y_1 a_{1n}}{a_{1.}} + \frac{y_2 a_{2n}}{a_{2.}} + \dots + \frac{y_m a_{mn}}{a_{m.}} \right] \\ &= \left[\sum_{i=1}^m \frac{y_i a_{i1}}{a_{i.}}, \sum_{i=1}^m \frac{y_i a_{i2}}{a_{i.}}, \dots, \sum_{i=1}^m \frac{y_i a_{in}}{a_{i.}} \right]. \end{aligned}$$

Hence the estimate \hat{y}_j of the value of Y on the target region t_j can be written as

$$\hat{y}_j = \sum_{i=1}^m \frac{y_i a_{ij}}{a_{i.}} \quad (3.1.2)$$

We see that the source Y value is apportioned according to the ratio of intersection area to source region areas, and then \hat{y}_j is found by the appropriate column sum.

This method of interpolation is frequently used in practice. Although the main assumption of uniform distribution may be questioned, effects of non-uniformity may be reduced by ensuring that source regions are relatively small.

3.1.1 Example

Consider Figure 3.1 with two source and three target regions so that $m = 2$, $n = 3$, and

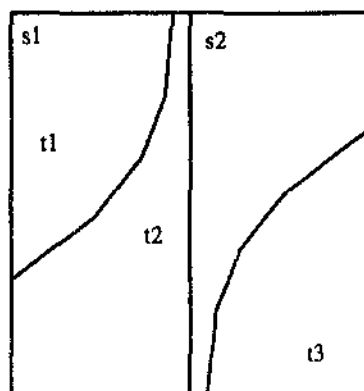


Fig 3.1

we also have

$$y_s = [4, 6]',$$
$$A_s = [2 \ 2]', \quad A_t = [1 \ 2 \ 1]'$$

and so

$$A_{st} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

Using (3.1.1) gives

$$W = \begin{bmatrix} 0.5 & 0.5 & 0 \\ 0 & 0.5 & 0.5 \end{bmatrix},$$

and the target value estimates \hat{y}_t can be found by

$$\begin{aligned} \hat{y}_t &= y_s' W \\ &= [4 \ 6] \begin{bmatrix} 0.5 & 0.5 & 0 \\ 0 & 0.5 & 0.5 \end{bmatrix} \\ &= [2 \ 5 \ 3]. \end{aligned}$$

3.2 Enhanced Areal Interpolation

The areal weighting interpolation method described and illustrated in section 3.1 is a reasonably simple and straight forward process that is relatively easy to implement in practice. The Enhanced Areal Interpolation method differs from this in the use of

- ancillary data on the target regions to enhance the precision of the interpolated values,
- a statistical distribution to model data on intersection regions, and
- the EM algorithm to implement the estimation procedure providing estimates for y values on the intersection regions and hence target regions.

3.2.1 The Ancillary Variable

The ancillary variable X has known values on the target regions and is used to indicate target regions with a similar distribution of the variable of interest. The use can be likened to a classification or category, where target regions with the same value of the ancillary variable will have a similar distribution. The model uses this extra information to distribute the variable of interest Y to the intersection regions accordingly. Essentially, X is used to 'flag' target regions of a similar type. One example (Flowerdew et al 1991) uses the soil types, clay and limestone, in the interpolation of population. In this case, assuming independent Poisson data the model becomes

$$\mu_{ij} = \begin{cases} \lambda_1 A_{ij} : x = 1 (\text{clay}) \\ \lambda_2 A_{ij} : x = 2 (\text{limestone}). \end{cases} \quad (3.2.1)$$

Thus the mean of the variable of interest Y on each intersection region is proportional to the area of the intersection with constant of proportionality λ_x depending on the soil type x of the region.

Without loss of generality, we assume that X takes the values $1, 2, \dots, k$, with $k \leq n$. The proportionality constant λ is set so that there is a distinct value λ_x for each distinct value x of X . It becomes an indicator for target regions that are presumed to have the same distribution of the Y variable. Hence

$$\lambda = [\lambda_x] ; x = 1, 2, \dots, k.$$

From (3.2.1) the model then becomes

$$\mu_{ij} = \begin{cases} \lambda_1 A_{ij} : x = 1 \\ \lambda_2 A_{ij} : x = 2 \\ \vdots \\ \lambda_k A_{ij} : x = k. \end{cases} \quad (3.2.2)$$

3.2.2 The Statistical Model

This application of the enhanced areal interpolation method assumes that Y is an extensive variable and that on the intersection regions the Y_{ij} have a Poisson distribution with a mean involving an unknown parameter β ; so that

$$Y_{ij} \sim \text{Poisson}(\mu_{ij}); \text{ mean } \mu_{ij} = \mu(\beta, x_j, A_{ij})$$

The implementation of the EM algorithm requires some initial estimate to prime it. This is provided by first applying Areal Weighting Interpolation to the data and using the result obtained as this initial estimate. This essentially duplicates the first iteration of the E step and so this application of the EM algorithm commences with the M step. The first estimates from Areal Weighting Interpolation are designated as $\hat{y}_{ij}^{(1)}$, the superscript (1) indicating that this is the first iteration of the EM algorithm.

Maximum likelihood estimates are used to maximise the joint density likelihood at the observed values. The likelihood function, in the special case of two distinct values of X , becomes

$$\begin{aligned} L(\lambda_1, \lambda_2; \hat{y}_{ij}^{(r)}) &= f(y_{ij}; \lambda_1) f(y_{ij}; \lambda_2) \\ &= \prod_{ij} \frac{e^{-\mu_{ij}} (\mu_{ij})^{y_{ij}}}{y_{ij}!} \\ &= \frac{e^{-\lambda_1 \sum_{(1)} A_{ij}} e^{-\lambda_2 \sum_{(2)} A_{ij}} \lambda_1^{\sum y_{ij}} \lambda_2^{\sum y_{ij}} \prod_{(1)} A_{ij}^{y_{ij}} \prod_{(2)} A_{ij}^{y_{ij}}}{y_{ij}!} \\ &= \left(e^{-\lambda_1 \sum_{(1)} A_{ij}} \lambda_1^{\sum y_{ij}} \right) \left(e^{-\lambda_2 \sum_{(2)} A_{ij}} \lambda_2^{\sum y_{ij}} \right) K \end{aligned}$$

where (x) ; $x = 1, 2$; indicates a sum (product) over all intersections i, j with the same value of the ancillary variable X , and where K is a constant function of λ_1, λ_2 .

To maximise the likelihood function, L is differentiated with respect to λ_x ; $x = 1, 2$; obtaining

$$\frac{\partial L}{\partial \lambda_x} = 0,$$

giving

$$-\lambda_i \sum_{(x)} A_{ij} + \sum_{(x)} y_{ij} = 0,$$

and hence

$$\hat{\lambda}_x = \frac{\sum_{(x)} y_{ij}}{\sum_{(x)} A_{ij}}. \quad (3.2.3)$$

The estimated parameter values $\hat{\mu}_{ij}$ can be calculated for each intersection region, using (3.2.1) as

$$\hat{\mu}_{ij} = \begin{cases} \hat{\lambda}_1 A_{ij} : x = 1; \\ \hat{\lambda}_2 A_{ij} : x = 2. \end{cases}$$

These estimates are then used for the next E step.

3.2.3 E step

The 'missing' data values are those values of Y on the regions of intersection of the source and target regions. The E step uses the values of Y on the source regions, together with estimates of parameter values, to provide a set of estimates \hat{y}_{ij} for the values y_{ij} on the intersection regions. Using the Poisson model described above, these estimates are calculated from the known values y_i on the source regions.

Assuming Y_{ij} are independent Poisson random variables with parameters μ_{ij} , we require the conditional distribution of Y_{ij} given y_i . Section (2.3.4) shows that the conditional Poisson distribution is binomially distributed and has parameters

$$n = y_i, \quad p = \frac{\mu_{ij}}{\sum_k \mu_{ik}}; \quad i = 1, 2, \dots, m; \quad j = 1, 2, \dots, n.$$

Hence the estimates are the conditional expectation of y_{ij} given y_i and μ_{ij} ,

$$\begin{aligned}\hat{y}_{ij} &= E[y_{ij}; \mu_{ij}, y_i.] \\ &= np \\ &= y_i \cdot \frac{\hat{\mu}_{ij}}{\sum_k \hat{\mu}_{ik}}.\end{aligned}\tag{3.2.4}$$

In practice the implementation of the E step is straightforward, (3.2.4) is used to find the next \hat{y}_{ij} value. It is convenient to consider these values in a matrix $\hat{U} = [\hat{\mu}_{ij}]$, and the matrix operation equivalent to (3.2.4) becomes

$$\hat{y}_{ij} = Y_s' \begin{bmatrix} \hat{\mu}_{ij} \\ \hat{\mu}_{i.} \end{bmatrix}; \quad i = 1, 2, \dots, m; \quad j = 1, 2, \dots, n\tag{3.2.5}$$

where the denominators are the row sums of \hat{U} , namely

$$\hat{\mu}_{i.} = \sum_j \hat{\mu}_{ij}$$

The matrix in (3.2.5) is analogous to the weighting matrix W in areal weighting interpolation.

The stopping rule has to be considered at this stage in the algorithm. This is discussed in section 3.2.6.

The estimates from the E step are then used in the M step.

3.2.4 M Step

Each time the E step is completed the maximum likelihood equations are used to fit the model to these estimated values and produce maximum likelihood estimates for the unknown population parameters.

The M step is composed of two distinct operations. The proportionality constant, lambda, is estimated first, followed by the estimation of the $\hat{\mu}_{ij}$ using

$$\hat{\mu}_{ij} = \begin{cases} \hat{\lambda}_1 A_{ij} : x = 1 \\ \hat{\lambda}_2 A_{ij} : x = 2 \\ \vdots \\ \hat{\lambda}_k A_{ij} : x = k. \end{cases}$$

The lambda estimates are calculated using (3.2.3)

$$\hat{\lambda}_k = \frac{\sum_{(k)} y_{ij}}{\sum_{(k)} A_{ij}},$$

where the sum $\sum_{(k)}$ is the sum of the values for which $x = k$.

3.2.5 The Estimation Procedure

The EM algorithm involves iteration of the E and M steps until convergence (see 3.2.6). After the r th iteration the estimates for Y , μ and λ on the intersection regions are defined by

$$\hat{y}_{ij}^{(r)}, \hat{\mu}_{ij}^{(r)} \text{ and } \lambda_x^{(r)} \text{ respectively.}$$

Once the estimates \hat{y}_{ij} have been calculated the estimates \hat{y}_j are calculated by

$$\hat{y}_j = \sum_i \hat{y}_{ij}. \quad (3.2.6)$$

3.2.6 Convergence

There is a need to decide when the estimate has converged with a reasonable level of accuracy and hence stop the iterative process. The method chosen for this study is to use a stopping rule, which allows for some flexibility, yet provides a reasonable level of accuracy to be reached.

For each intersection region, after the r th iteration of the EM algorithm, we define the difference between estimated values for each intersection region as

$$\Delta_{ij}^{(r)} = \hat{y}_{ij}^{(r)} - \hat{y}_{ij}^{(r-1)} \quad (3.2.7)$$

This difference is used to determine when a sufficient level of accuracy has been reached and the iterative process may cease. Suppose a sufficient level of accuracy is $\epsilon = 0.05$ say. Then when

$$\max\left(\left|\Delta_{ij}^{(r)}\right|\right) = \delta_{ij}^{(r)} < \epsilon, \quad \text{for } i = 1, 2, \dots, m; j = 1, 2, \dots, n \quad (3.2.8)$$

the process will stop.

An iteration counter for r is also needed to limit the number of iterations should convergence become slow.

3.3 Model Fit

The likelihood function for this model on the intersection regions is

$$L(\mathbf{y}; \boldsymbol{\mu}) = \prod_i \prod_j \frac{e^{-\hat{\mu}_{ij}} \hat{\mu}_{ij}^{\hat{y}_{ij}}}{\hat{y}_{ij}!}$$

with log likelihood

$$\ell(\hat{\boldsymbol{\mu}}; \hat{\mathbf{y}}) = \sum_i \sum_j \left[-\hat{\mu}_{ij} + \hat{y}_{ij} \ln \hat{\mu}_{ij} - \ln(\hat{y}_{ij}!) \right]$$

giving deviance (2.5.2) on the intersection regions as

$$\begin{aligned} D(\hat{\boldsymbol{\mu}}; \hat{\mathbf{y}}) &= -2 \left[\ell(\hat{\boldsymbol{\mu}}; \hat{\mathbf{y}}) - \ell(\hat{\mathbf{y}}; \hat{\mathbf{y}}) \right] \\ &= 2 \sum_i \sum_j \left[\hat{y}_{ij} \left(\ln \frac{\hat{y}_{ij}}{\hat{\mu}_{ij}} \right) - (\hat{y}_{ij} - \hat{\mu}_{ij}) \right]. \end{aligned} \quad (3.3.1)$$

Deviance can also be calculated from the source region y values and is denoted as D_s .

The estimated y values for the target regions can be used to find a 'target deviance' D_t .

The calculations can be made as follows. From the E step we have

$$\hat{y}_{ij} = \frac{y_i \hat{\mu}_{ij}}{\sum_k \hat{\mu}_{ik}}$$

and hence

$$\frac{\hat{y}_{ij}}{\hat{\mu}_{ij}} = \frac{y_{i.}}{\sum_k \hat{\mu}_{ik}} = \frac{y_{i.}}{\hat{\mu}_{i.}}$$

Using this result and summing (3.3.1) over j, gives

$$D_s = 2 \sum_i \left[y_{i.} \ln \left(\frac{y_{i.}}{\hat{\mu}_{i.}} \right) - (y_{i.} - \hat{\mu}_{i.}) \right] \\ \left(\sum_j \hat{y}_{ij} = y_{i.}, \sum_j \hat{\mu}_{ij} = \hat{\mu}_{i.} \right). \quad (3.3.2)$$

By summing (3.3.1) over i gives

$$D_t = 2 \sum_j \left[\hat{y}_{.j} \ln \left(\frac{\hat{y}_{.j}}{\hat{\mu}_{.j}} \right) - (\hat{y}_{.j} - \hat{\mu}_{.j}) \right] \\ \left(\sum_i \hat{y}_{ij} = \hat{y}_{.j}, \sum_i \hat{\mu}_{ij} = \hat{\mu}_{.j} \right). \quad (3.3.3)$$

For the matrix of estimated Y values

$$\hat{y}_{ij} = [\hat{y}_{ij}]$$

where

$$\hat{y}_{i.} = \sum_k \hat{y}_{ik}, \quad \hat{y}_{.j} = \sum_\ell \hat{y}_{\ell j},$$

and for the matrix of estimated μ_{ij} values

$$\hat{\mu}_{ij} = [\hat{\mu}_{ij}]$$

where

$$\hat{\mu}_{i.} = \sum_k \hat{\mu}_{ik}, \quad \hat{\mu}_{.j} = \sum_\ell \hat{\mu}_{\ell j}.$$

4. Implementation

4.1 Initial Development

The application of the enhanced areal interpolation method is best illustrated with a simple example. This example was incorporated into a spreadsheet model. A copy of this model is in Appendix A.

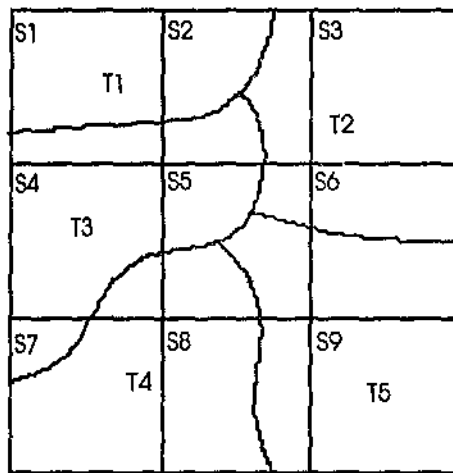


Fig 4.1 - Source regions and target regions.

Consider nine square source regions each having unit area arranged into a regular 3x3 grid forming a square with a total area of nine. This 3x3 grid is completely divided into five target regions, as shown in Figure 4.1.

The data available for the source regions is given Table 4.1

Region	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	Total
i	1	2	3	4	5	6	7	8	9	-
Ai.	1	1	1	1	1	1	1	1	1	9
Yi.	2	3	5	2	3	4	2	3	6	30

Table 4.1 - Source region data.

The five target regions have an ancillary variable X_i with the values as shown in Table 4.2.

Region	t1	t2	t3	t4	t5
j	1	2	3	4	5
X_j	1	2	3	3	2

Table 4.2 - Ancillary values for target regions.

The nine source regions and five target regions form intersection regions. The areas of these intersection regions are given in Table 4.3.

Area A_{ij}	j					Σ
	1	2	3	4	5	
1	0.7696	0.0000	0.2304	0.0000	0.0000	1.000
2	0.4440	0.3254	0.2306	0.0000	0.0000	1.000
3	0.0000	1.0000	0.0000	0.0000	0.0000	1.000
4	0.0000	0.0000	0.8766	0.1234	0.0000	1.000
5	0.0000	0.1240	0.3235	0.2785	0.2740	1.000
6	0.0000	0.4700	0.0000	0.0000	0.5300	1.000
7	0.0000	0.0000	0.1490	0.8510	0.0000	1.000
8	0.0000	0.0000	0.0000	0.6785	0.3215	1.000
9	0.0000	0.0000	0.0000	0.0000	1.0000	1.000
Σ	1.2135	1.9195	1.8101	1.9314	2.1255	9.000

Table 4.3- Intersection region areas.

Table 4.3 shows the areas of the intersection regions, with source region number as the row heading and target region number as column heading. The row totals give the area of each source region, and the column totals give the area of each target region. Note that the table is sparse, as each target region intersects only a few source regions.

Initial estimates for the EM algorithm are obtained from the areal weighting interpolation estimates. Table 4.3 is the equivalent of the area matrix $A = [a_{ij}]$, the intersection region areas. Equation (3.1.2) is used to find the areal weighting estimate of the y values.

For example

$$\begin{aligned}\hat{y}_{11} &= \frac{a_{11}}{a_1} y_1 \\ &= \frac{0.7696}{1} (2) \\ &= 1.5392.\end{aligned}$$

The result of the calculations for all intersection regions is given in Table 4.4.

est y _i	j					Σ
	1	2	3	4	5	
1	1.539	0.000	0.461	0.000	0.000	2.00
2	1.332	0.976	0.692	0.000	0.000	3.00
3	0.000	5.000	0.000	0.000	0.000	5.00
4	0.000	0.000	1.753	0.247	0.000	2.00
5	0.000	0.372	0.970	0.836	0.822	3.00
6	0.000	1.880	0.000	0.000	2.120	4.00
7	0.000	0.000	0.298	1.702	0.000	2.00
8	0.000	0.000	0.000	2.035	0.965	3.00
9	0.000	0.000	0.000	0.000	6.000	6.00
Σ	2.871	8.228	4.174	4.820	9.906	30.00

Table 4.4 - Areal weighting estimates.

Note that the column totals give the areal weighting interpolation estimate for the target regions. This can be compared with the enhanced areal interpolation results.

E step (1)

The first estimates are calculated using (3.2.4) which are the areal weighting interpolation estimates.

M step (1)

The number of lambda values needs to be determined before commencing with the M step of the EM algorithm. In practice the ancillary values x_i , $i = 1, 2, \dots, n$ will not necessarily take the values $1, 2, \dots, k$, $k \leq n$, and will need recoding. Suppose this recoded value, called the lambda flag, is z_j ; $j = 1, 2, \dots, n$.

In our example 'scanning' through the target regions from $j = 1$ to 5 , finds the first ancillary value $x_1 = 1$, and so put $z_1 = 1$. The next ancillary value is $x_2 = 2$ which is distinct from x_1 so put $z_2 = 2$. The next ancillary variable $x_3 = 3$, and so put $z_3 = 3$. The next ancillary variable $x_4 = x_3 = 3$, and so $z_4 = z_3 = 3$. Lastly, $x_5 = x_2 = 2$, and so put $z_5 = z_2 = 2$. The recoded variable Z takes the values $1, 2, 3$, and $k = 3$.

Once the lambda flags are found then (3.2.1) is used to estimate μ_{ij} for each intersection region.

From (3.2.2) the model is

$$\mu_{ij} = \lambda_z A_{ij}, \quad z = 1, 2, 3.$$

The first step is to find the lambda estimates using (3.2.3). For example

$$\begin{aligned} \hat{\lambda}_2^{(1)} &= \frac{\sum_j y_j^{(1)}}{\sum_j a_j} \quad : z = 2; j = 2, 5 \\ &= \frac{y_2^{(1)} + y_5^{(1)}}{a_2 + a_5} \\ &= \frac{8.228 + 9.906}{1.919 + 2.126} \\ &= 4.483. \end{aligned}$$

Similarly the other values are calculated. Table 4.6 shows the new lambda estimates for all target regions as calculated. Note that the lambda values for target regions $j = 1, 3, 4$ are equal.

Lambda (1)						
λ	j					Σ
	1	2	3	4	5	
X_j	1	2	3	3	2	11
ΣY_j	2.871	8.228	4.174	4.820	9.906	30.00
ΣA_j	1.214	1.919	1.810	1.931	2.125	9.00
λ	2.366	4.483	2.404	2.404	4.483	16.14

Table 4.6 - Lambda estimates.

The estimates for μ_{ij} are then calculated using these lambda values in (3.2.2). As an example $z_1 = 1$ and so the calculation of the estimate for μ_{11} is given by

$$\begin{aligned}\hat{\mu}_{11}^{(1)} &= \hat{\lambda}_1 a_{11} \\ &= 2.366(0.7696) \\ &= 1.8207.\end{aligned}$$

Table 4.7 shows the remainder of the μ_{ij} estimates to 1 decimal place. Note that

Table 4.7 is similar to matrix U, but as a table includes the row and column totals. This is useful in a spreadsheet as deviance can be calculated at any point.

M step (1)		j					
est mu	1	2	3	4	5	Σ	
1	1.8207	0.0000	0.5539	0.0000	0.0000	2.375	
2	1.0503	1.4591	0.5543	0.0000	0.0000	3.064	
3	0.0000	4.4833	0.0000	0.0000	0.0000	4.483	
4	0.0000	0.0000	2.1072	0.2967	0.0000	2.404	
5	0.0000	0.5560	0.7776	0.6695	1.2283	3.231	
6	0.0000	2.1072	0.0000	0.0000	2.3762	4.483	
7	0.0000	0.0000	0.3582	2.0457	0.0000	2.404	
8	0.0000	0.0000	0.0000	1.6310	1.4415	3.073	
9	0.0000	0.0000	0.0000	0.0000	4.4833	4.483	
Σ	2.8710	8.6056	4.3512	4.6429	9.5293	30.000	

Table 4.7 - New estimates of μ_{ij} .

E Step (2)

With $\hat{\mu}_{ij}$ calculated the M step is completed, and the estimated y values are calculated for the next iteration of the algorithm. Table 4.7 is very useful since the estimated y value for each intersection region is found using (3.2.4). There are three inputs to this calculation, the y_i , $\hat{\mu}_{ij}^{(1)}$ which is the cells of Table 4.7 and $\mu_{i.}^{(1)}$ which is the row total from Table 4.7.

An example calculation using (3.2.4) is shown for the intersection region of s_2 and t_1 .

$$\hat{\mu}_{21}^{(1)} = 1.0503$$

$$\begin{aligned} \hat{y}_{21}^{(2)} &= \frac{1.0503}{1.0503 + 1.4591 + 0.5543} \cdot 3 \\ &= \frac{1.0503}{3.064} \cdot 3 \\ &= 1.0285. \end{aligned}$$

In a similar manner all the values are found. Table 4.8 shows the remainder of the estimates of the intersection region Y values after the second E step. The process then continues.

Convergence

E step (2)						
est y _{ij}	j					Σ
	1	2	3	4	5	
1	1.5335	0.0000	0.4665	0.0000	0.0000	2.000
2	1.0285	1.4287	0.5428	0.0000	0.0000	3.000
3	0.0000	5.0000	0.0000	0.0000	0.0000	5.000
4	0.0000	0.0000	1.7531	0.2469	0.0000	2.000
5	0.0000	0.5162	0.7219	0.6216	1.1403	3.000
6	0.0000	1.8800	0.0000	0.0000	2.1200	4.000
7	0.0000	0.0000	0.2980	1.7020	0.0000	2.000
8	0.0000	0.0000	0.0000	1.5925	1.4075	3.000
9	0.0000	0.0000	0.0000	0.0000	6.0000	6.000
Σ	2.5620	8.8249	3.7824	4.1629	10.6678	30.000

Table 4.8 - Estimated Y values after second E step.

For this example the accuracy level chosen was $\epsilon = 0.001$. Table 4.9 shows the results of the delta calculations after the second E step. These are values of $D_{ij}^{(2)}$.

D	J					Σ
	1	2	3	4	5	
1	0.000	0.000	0.000	0.000	0.000	0.000
2	-0.090	0.149	-0.060	0.000	0.000	0.000
3	0.000	0.000	0.000	0.000	0.000	0.000
4	0.000	0.000	0.000	0.000	0.000	0.000
5	0.000	0.229	0.045	0.030	-0.303	0.000
6	0.000	0.000	0.000	0.000	0.000	0.000
7	0.000	0.000	0.000	0.000	0.000	0.000
8	0.000	0.000	0.000	0.168	-0.168	0.000
9	0.000	0.000	0.000	0.000	0.000	0.000
Σ	-0.090	0.378	-0.015	0.198	-0.471	0.000

Table 4.9 - Delta values after E step 2.

From Table 4.9 it is clear that the maximum absolute value for delta is 0.303 and so the process should continue. The spreadsheet was designed to be iterative, and once the iteration process was started it continued until the specified level of accuracy was reached. This occurred after seven iterations and the results are given in Table 4.10.

Est Yst	J					Σ
	1	2	3	4	5	
1	1.5458	0.0000	0.4542	0.0000	0.0000	2.000
2	0.9043	1.6347	0.4610	0.0000	0.0000	3.000
3	0.0000	5.0000	0.0000	0.0000	0.0000	5.000
4	0.0000	0.0000	1.7531	0.2469	0.0000	2.000
5	0.0000	0.5835	0.6058	0.5216	1.2891	3.000
6	0.0000	1.8800	0.0000	0.0000	2.0000	4.000
7	0.0000	0.0000	0.2980	1.7020	0.0000	2.000
8	0.0000	0.0000	0.0000	1.3694	1.6306	3.000
9	0.0000	0.0000	0.0000	0.0000	6.0000	6.000
Σ	2.4500	9.0983	3.5721	3.8398	11.0398	30.000

Table 4.10 - Spreadsheet estimates

The source deviance was 0.4144 for the enhanced areal interpolation estimates.

This exercise provided valuable insight to implementation issues and also provided results that could be used to check the output from the implementation.

4.2 Implementation

The implementation was made in three broad stages using the source and target regions as described in section 4.1.

(i) Stage 1 involved the integration of the application with MapInfo and the construction of a user interface.

(ii) Stage 2 used the inbuilt MapInfo operations for creation of intersection regions. MapInfo version 3 includes a function that can use the shape of a polygon to divide up another polygon. The terminology used by the software manuals is that of a 'cookie cutter'. This split function allows the data attributes associated with the polygon being split to be allocated by three methods

1. No data is allocated, the data attribute is left blank or zero,
2. Data is duplicated, as in a region name;
3. Numeric data only can be allocated using area proportioning.

The third option is areal weighting interpolation. Each attribute of a data item can have an option individually specified.

(iii) Stage 3 was the implementation of the EM algorithm and the production of estimates on the target regions.

4.3 Software and Hardware

Both the software and hardware for the implementation had to be of a kind readily available to a large number of users. This was particularly important in the choice of hardware, as being that which is likely to be found in a wide variety of uses.

4.3.1 Software

The calculation aspect of the enhanced areal interpolation can be implemented in many packages, and as shown in the previous chapter, can be designed into a spreadsheet package. The crucial aspect of the software is the GIS package which is used to calculate the areas of source and intersection regions. It needed to be

- readily available,
- relatively inexpensive,
- easy to learn and use
- easily able to interface with the calculation software, and
- able to operate on a wide variety of computers.

Based on these criteria the software package chosen was *MapInfo version 3.0 for Windows*. This software package is termed a desktop mapping application rather than a GIS. For our purposes it had all the necessary features. An added benefit was the associated programming language *MapBasic version 3.0 for Windows*, which provided the ability to integrate the whole application.

4.3.2 Hardware

The software packages chosen are available for a wide variety of computers including Unix, Macintosh and IBM compatible Personal Computers using Microsoft Windows software. The hardware used was an IBM PC compatible computer with an Intel 486 Central Processing Unit (CPU), and 4 Mb of random access memory using Microsoft Windows version 3.1. This choice of hardware would ensure that any application package produced would be able to operate on hardware that is readily available in the wider community.

4.4 Design

An appropriate representation of the implementation is to consider the source and target regions as in a table as shown in Table 4.1. This was the method used for the example in Chapter 4 and enables matrix operations to be used which results in great simplicity of implementation. MapInfo allows the use of only one dimensional arrays (in effect vectors). This restriction greatly influenced the design of the application.

MapInfo is essentially a database management system (DBMS) that stores the data in tables. Each row in a table is a data item and the columns are the attributes of the data. MapInfo has a special geographic attribute that can be given to the data in a table that allows the data items to have an associated geographic image, be it a point, line or polygon, that is geocoded in any of a number of co-ordinate systems. Hence the main difficulty of the implementation was that table manipulation was required via relational database operations. A row in a table can also be referred to as a data item or record, while a column can also be referred to as a data attribute or a data field.

The design of the implementation was strongly influenced by this aspect of MapInfo, and so involved a combination of array type operations, such as loops, with relational database operations, such as row or record selection. MapBasic is designed for easy creation of modular code, and the application is strongly modular in design, making it relatively straightforward to modify any aspect of operation, or make enhancements either as modifications or additions.

4.5 Execution

4.5.1 Integration and User Interface

The application has been set up to operate as a sub menu item in MapInfo. It has four sub menus of its own. These sub menus are

- commence the application,
- exit the application and remove the menu option,
- give a brief description of the application,
- provide help information.

Figure 4.2 shows the MapInfo menus and the application sub menus.

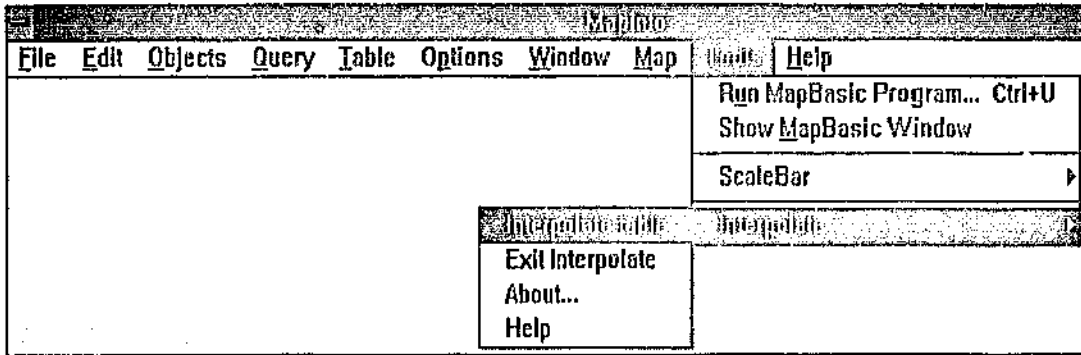


Figure 4.2 - MapInfo menus & sub menus.

The user may set up the application to install itself whenever MapInfo starts up.

When the application is run from within MapInfo, it only installs itself by adding an interpolation menu item. The enhanced interpolation commences when the third sub menu is chosen. The user is given a dialog box on the screen as shown in Figure 4.3. This dialog box requests the name of the table that contains the source regions. This table is referred to as the source table. Once this table has been selected a pop up menu of attributes becomes available from which to chose the column that contains the Y values. Another input area allows the user to select the target table, and the corresponding column in that table that is to be used as the ancillary variable. The user is not able to chose the same table for both source and target regions.

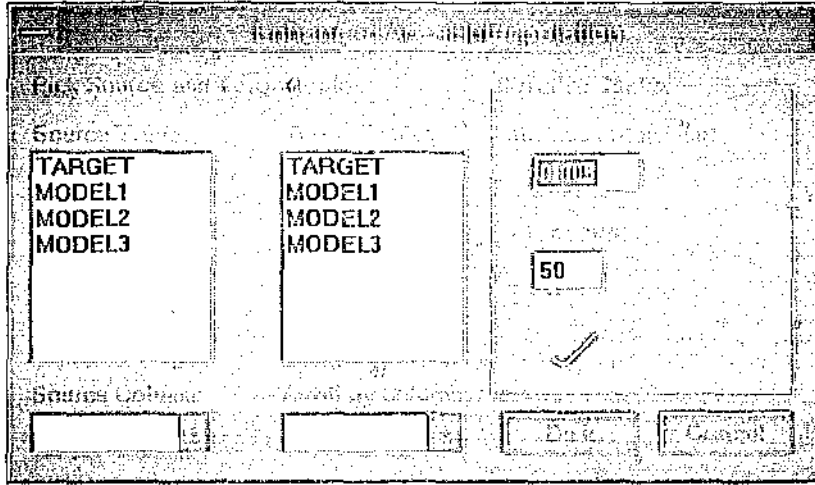


Fig 4.3 - User interface dialog box.

The dialog box also contains a section for iteration control. This allows the user to specify the level of accuracy ϵ and the maximum number of iterations. These values are initially set to 0.005 and 50 respectively.

4.5.2 Table Manipulation

A major part of the implementation effort was in having to manipulate MapInfo tables, especially since every operation requires some rows of a table to be selected, and the selected rows create another table called a selection table. To give some idea of this, consider the example described in section 4.1. This has nine source regions, five target regions and 15 intersection regions and one iteration through the EM algorithm results in the creation of over 80 tables.

On the positive side, alterations to selected rows are carried through to the original table. This can be very useful for altering the fields of specific records in a table. For example consider a table containing a column of postcodes and a column of locality names together with other columns. A selection could be made on a particular locality name, say Perth City. The selection can be made to contain only the postcode and

locality name columns. The locality name column in this selection table would contain only the locality selected (Perth City), and the postcode column would contain only the postcodes associated with this locality. These postcodes could be altered to some other value. This alteration would then be carried through to the original table from which the selection was made, only altering those rows that were selected; that is only the postcode for those data records with locality name of Perth City would be changed. This is a very powerful feature. This particular ability was especially useful in calculating estimates for only selected intersection regions in the M step.

One other aspect of the table structure that had a major influence on the application design was the way the split function operated. As mentioned above, this function operates like a 'cookie cutter'. A table is specified to the function as the one to be 'cut', a second table is specified as the 'cookie cutter' and divides up the first table. In many MapInfo operations the result produced has no effect on the original arguments to the operation. However the split function differs in this respect. The net result is that after the split function is used the source table has been 'cut up' into intersection regions and no longer exists in its original form, and it is very difficult to put back together again.

This effect of the split function could cause unwanted alteration of a data table. To ensure that this could not happen the implementation was designed to operate on copies of tables. This had two effects on the implementation design. Firstly the source and target tables are copied to temporary tables and all operations take place on these temporary tables, thus ensuring that the users' data is not affected until all calculations are complete. Secondly, the split source table becomes the main table on which all the application operations are carried out. This then becomes an *intersection table*.

4.5.3 EM Algorithm

The EM algorithm is the major feature of the implementation and accounts for over a third of the application code. Most of the rest is accounted for by the user interface. Implementation follows closely the spreadsheet design discussed in Chapter 4.

To account for the first iteration of the E step being replaced by the areal weighting estimates, the EM algorithm is implemented as an ME algorithm with the M step first. This suits the convergence limit, as this is calculated after the E step and so is more suited to this reversal.

As noted in Chapter 3, the lambda flags need to be set up first. However, the explanation in Chapter 3, while easy to follow, does not constitute an efficient algorithm as each target region is examined and then compared to some list of previous ancillary values to determine the z value to be allocated. This is CPU time consuming and memory using. A computationally more efficient algorithm which should reduce the number of comparisons made is described below.

Before the split operation the target table is indexed $j = 1, 2, \dots, n$ in the order in which the data records occur, and the source table is indexed in a similar manner. This ensures that there is a source table and target table index in each intersection region. In the split operation any data attribute not specified will default to either a blank or zero entry. This is a simple way of ensuring that all lambda flag fields are initially zero. The allocation of data flags proceeds as below.

1. Lambda flag column is denoted as lf , and the ancillary variable column as $xvalue$. A copy of each lambda value is kept in a one dimensional array denoted as $lamf(z)$, where the maximum number of z values is set to the number of intersection regions.
2. Set a counter variable 'index = 1', number of flags counter 'count = 1'.
3. From the intersection table get record number = index, set lambda flag $lf = 1$ and $lamf(1)$ to the lambda value .
4. Now select all records from the intersection table that have the same lambda value as $lamf(1)$. For these selected records set $lf = 1$.
5. While $index \leq$ number of intersection table rows do the following
 - 5.1. Increment index by 1.
 - 5.2. From intersection table get record with record number = index
 - 5.3. If lf for this record is zero then (needs a lambda flag),
 - increment count by 1,
 - set $lamf(count)$ to the lambda value for this record,
 - select all records from the intersection table that have the same lambda value as $lamf(count)$, and for these records set $lf = count$.
6. The final value of count gives the number of lambda flags set which is variable k .

This then gives k as the maximum value of z . It is worthwhile to note that although each record is examined, if the flag has been set, then it is skipped over, thus speeding up the execution of this stage of the implementation.

The M step then operates in a loop that is executed once for each lambda flag. MapInfo provides a very useful summation operation that simplifies the calculations. Say a data table is used for the finances of shop locations and has a column that contains lease payments. It may be a useful financial analysis to obtain total lease payments by distribution region. Rows can be selected from the table using distribution region as the selection criterion, and a column can be included that provides the sum of the lease values. This sum can be assigned to a variable and used in other calculations. This means that an entire column of data could be arithmetically operated on by this assigned value. For each lambda flag, the calculation to find μ_{ij} is done once and the updates all selected records. This summation operation makes the E step very simple, and this took only ten lines of code to implement.

4.5.4 Producing Results

A simple test determines if the value of ϵ has been reached or if the maximum number of iterations has been reached. If either case is true the iterations stop, both values are recorded and reported to the user as they may be useful for further analysis.

Both source deviance D_s and target deviance D_t are calculated. The former uses the source index column and the latter uses the target index column that set up in the intersection table. They can both be reported to the user. An interesting situation arises in the calculation of source deviance. It is possible for the Y variable column in the

source table to contain zero for one or more source regions. These regions contribute zero to the estimates, but the deviance calculation must take into account the possibility of a zero value. Where the source region has a y value of zero the deviance calculation for that region uses only the value μ_i .

The target region index is used to sum the estimates in a process we refer to as assembling. At this point the table specified by the user as the target table is modified by the addition of a column with the same column title as the Y variable column in the source table. It is into this column that the estimates are summed. The temporary tables are not deleted as they may be of interest to the user. The temporary tables are deleted when either another iteration is commenced, or when the application is closed.

5. Evaluation

In this chapter we include the results obtained to date and compare the areal weighting interpolation and enhanced areal interpolation methods. The first example used is that described in Chapter 4, and some of the results may be indicative of the 'artificial' nature of this example. Rather than move directly to 'real' data, it was thought more prudent to study variations on the example from Chapter 4. The original example is referred to as Example 1 and the variations are Example 2 and Example 3. The target regions remain the same for all examples, only the source regions are modified by grouping and merging those from Example 1. A description of each example is given and this is followed by a comparative summary.

5.1 Example 1

Example 1 is shown in Figure 4.1. The results from this example are given in Table 4.10 in Section 4.1. The main feature of this example is having congruent source regions. The main concentration of the values of the Y variable is in the top row of source regions accounting for 33% of the total, and the right column which accounted for 50% of the total. The distribution of the values in this right column of regions can have a major effect on the final estimates. The overall areal density of Y is 3.333. Using areal density allows source regions with high variability to be identified easily.

5.2 Example 2

Example 2 is shown in Figure 5.1. In this example the original source regions s_1 and s_2

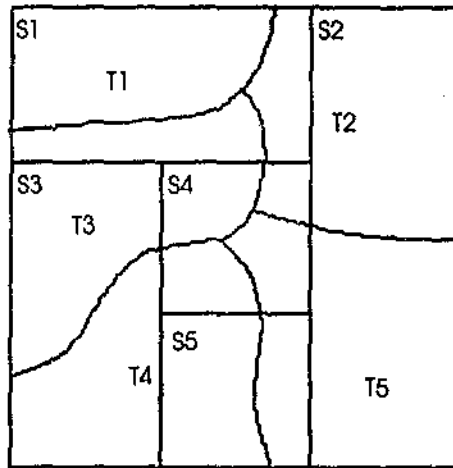


Figure 5.1

have been merged into one region to form s_1 . Regions s_3 , s_6 , and s_9 have been merged to form new source region s_2 . Regions s_4 and s_7 have been merged to form s_3 . This has the effect of 'smoothing out the variations in Y '. This smoothing has greatest effect on the right column, which should have greatest effect on target regions t_2 and t_5 . Table 5.1 gives the data for the source regions.

Region	s_1	s_2	s_3	s_4	s_5	Total
i	1	2	3	4	5	-
A_i	2	3	2	1	1	9
Y_i	5	15	4	3	3	30
Density	2.50	5.00	2.00	3.00	3.00	3.33

Table 5.1 - Source regions for Example 2

The smoothing effect can be seen in that the greatest areal density is down from 6 to 5, while there has been no change in the minimum density.

5.3 Example 3

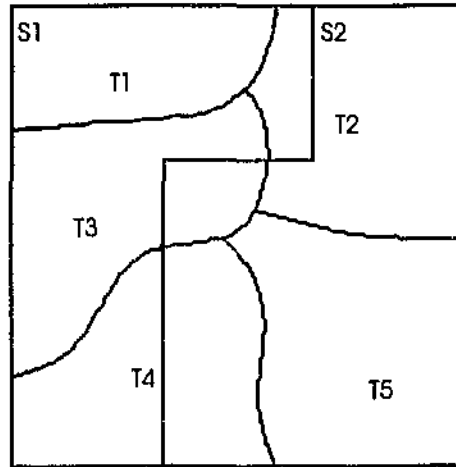


Figure 5.2 - Example 3 regions.

Example 3 is shown in Figure 5.2. This example continues the merging process, and has only two source regions. Example 3 has the greatest smoothing and very little of the detail that was present in Example 1. Region s_2 has a y value of 21, more than 50% of the total, with 56% of the total area. This source region now wholly encloses t_5 and some effect should be seen here. The original s_3 and s_9 are merged and the effects of these original two source regions should be less than in Example 1.

Table 5.2 gives the data for the source regions.

Region	s_1	s_2	Total
i	1	2	-
A_i	4	5	9
Y_i	9	21	30
Density	2.25	4.20	3.33

Table 5.2 - Source regions for example 3.

5.4 Comments

The results for all three examples are given in Table 5.4. For each example the target region estimates are given. For the enhanced areal interpolation the source deviance and number of iterations is also given. For each example the squared differences are given to compare estimates.

Example Results								
Example	Method	Target Regions					Source Deviance	Iterations
		1	2	3	4	5		
1	AWI	2.871	8.227	4.172	4.819	9.896	0.8014	6
	EAI	2.449	9.096	3.571	3.840	11.028	0.4139	
	Sq diff	0.178	0.756	0.360	0.958	1.281		
2	AWI	3.034	8.530	4.172	4.819	9.426	0.4252	7
	EAI	2.473	9.543	3.566	3.840	10.558	0.0114	
	Sq diff	0.315	1.027	0.366	0.959	1.282		
3	AWI	2.731	7.422	4.701	6.211	8.918	0.7284	9
	EAI	1.681	9.282	4.227	4.515	10.278	0.0000	
	Sq diff	1.103	3.458	0.225	2.874	1.849		

AWI: Areal Weighting Interpolation
EAI: Enhanced Areal Interpolation

Table 5.3 - Summary of results.

In each example the EAI estimates have lower deviance than the AWI estimates, indicating better fit. The AWI performed best in Example 2 with its lowest deviance. In Example 3, EAI produces a much better fit than AWI. This is due to the extra information that the ancillary variable provides to EAI. This can be seen best in target region t_2 , where the AWI estimate has decreased greatly from that obtained in Examples 1 and 2, whereas the EAI estimate is close to the values in the first two examples. AWI in using the uniform distribution is affected by the smoothing of the data to a much greater extent than EAI.

Where ancillary data is available for target regions that are generally smaller than source regions the EAI provides a more precise estimate than AWI. Table 5.4 gives the results of timings made for each example

Example	Iterations	Time(sec)	sec per iteration
1	6	104	17.33
2	7	87	12.43
3	9	72	8.00

Table 5.4 - Iteration timings.

The length of time for each iteration is clearly influenced by the number of source regions. The number of source regions for each example are in the ratio 9:5:2, while the times for one iteration are not in this ratio, there is some correspondence. The timings indicate that for larger numbers of source and target regions, the processing time will increase. For this reason the hardware used for this paper should be regarded as the recommended minimum.

6. Brief Summary

6.1 Implementation

The implementation can easily be operated by a user with little technical knowledge of the EM algorithm or Exponential Families of distributions. The implementation has been tested to handle a wide range of inputs and to detect and advise the user of invalid inputs. The implementation uses commercially available software and hardware that is widely distributed in the community. In meeting these criteria, it can be said to be robust.

The enhanced areal interpolation is an intensive processing task and while able to operate on older and slower hardware than that used in this paper, our recommendation is that the hardware used for this implementation is the minimum that should be used.

6.2 Usefulness

As noted in Chapter 4, enhanced areal interpolation improves on the precision of areal weighting interpolation when the source data is smoothed due to source regions encompassing a number of smaller target regions. The extra information provided by the ancillary variable is very useful in such situations. Enhanced areal interpolation can only be used where such an ancillary variable is available. For example, in estimating population numbers, the ancillary variable can be given values according to the main land use of the target region, or the target region may be simply classified as urban/non-urban.

The measure of deviance is worthwhile using for areal weighting interpolation as well as for enhanced areal interpolation. It provides a relatively simple method of measuring the precision of estimates.

The deviance value can also be used to measure the performance of different ancillary variable for the same set of data and regions. This allows the user to identify the ancillary variable that will provide the more precise estimate.

6.3 Future Directions

The timings in Chapter 5 indicate that the implementation may have room for speed improvement so that larger numbers of regions can be used without a speed penalty. The implementation uses up a great deal of memory and there may be techniques that can reduce this usage.

The next step is to test the implementation on real data and regions. Such an investigation could look at various aspects such as the effect of the number of ancillary variable values. Testing the implementation on real regions would provide iteration timings that may help improve the speed of the implementation.

In this paper a Poisson distribution is assumed, but as pointed out by Green (1989) the binomial distribution may have a use in enhanced areal interpolation and other distributions may have a use.

Moxey and Allanson (1994) have produced a comparison of techniques based on *Enhanced Areal Interpolation* considering categorical variables. An investigation of their paper may indicate that a more generalised implementation is possible.

7. References

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society B* 39. 1-38.

Flowerdew, R., Green, M., and Kehris, E. (1991). Using Areal Interpolation Methods in Geographic Information Systems. *Papers in Regional Science: The Journal of the RSAI*, 70,3, 303-15.

Freund, J. E. (1992). *Mathematical Statistics*. Fifth Edition. Englewood Cliffs, New Jersey: Prentice-Hall.

Goodchild, M., and Gopal, S. (Eds). (1989) *The Accuracy of Spatial Databases*. London: Taylor & Francis.

Goodchild, M. F., and Lam, N. S.-N. (1980). Areal Interpolation: A Variant of the Traditional Spatial Problem. *Geo-Processing 1*, 297-312.

Green, M. (1989). *Statistical Methods for Areal Interpolation: The EM Algorithm for Count Data*. Lancaster, England: North West Regional Research Laboratory, Research Report No 3.

Johnson, R. A. and Wichern, D. W. (1992). *Applied Multivariate Statistical Analysis*. Third Edition. Englewood Cliffs, New Jersey: Prentice-Hall

Kehris, E. (1989). *Interfacing ARC/INFO with GLIM*. Lancaster, England: North West Regional Research Laboratory, Research Report No 5.

McCullagh, P., and Nelder, J. A. (1988). *Generalized Linear Models*. Second Edition. London: Chapman & Hall.

Markoff, J., and Shapiro, G. (1973). The linkage of data describing overlapping geographical units. *Historical Methods Newsletter* 7, 34-46.

Moxey, A., and Allanson, P. (1994). Areal Interpolation of Spatially Extensive Variables: A Comparison of Alternative Techniques. *International Journal of Geographical Information Systems*. Sep 1994, 8, 3, 479-487.

Appendices

Appendix 1: Spreadsheet Model

This appendix shows the spreadsheet models, and the formulas used for calculation.

The first set of tables provides the source region inputs of area and Y values.

Project Example

Source Regions

i	As	Ys
1		
2		
3		
4		
5	As	Ys
6		
7		
8		
9		
Σ	9	30

Intersection Regions

Area	Ast	j					Σ
		1	2	3	4	5	
i	1	0.760	0.000	0.231	0.000	0.000	1.000
	2	0.				0.00	=Row sum
	3	0.				0.00	
	4	0.				0.00	
	5	0.	Ast			0.73	
	6	0.				0.30	
	7	0.				0.00	
	8	0.				0.22	
	9	0.000	0.000	0.000	0.000	1.000	
Σ		At = col sum					

Areal Weighting estimate

Yst	t					Σ	
	1	2	3	4	5		
i	1	1.54	0.00	0.46	0.00	0.00	=Ys
	2					0.00	
	3					0.00	
	4					0.00	
	5		Yst=Ast*Ys/As			0.82	
	6					0.12	
	7					0.00	
	8					0.97	
	9	0.00	0.00	0.00	0.00	0.00	
Σ		Yt = col sum					30

The table labelled "Intersection Regions" contains the areas of the intersection regions. Each table shown either contains a name, formula or both. Formulas are preceded by an equals sign. Hence the column containing "= row sum" in the Intersection Regions table contains a formula that sums the row values, but the column does not have a name.

The bottom row of this table contains "At = col sum". The equals sign and the formula to the right of indicates that these values are found from summing each column. "At" before the equals sign is the name given to the result of this calculation. This name may be used in other formulae.

The last table shown in this set calculates the Areal Weighting Interpolation estimates (Yst) for each intersection region. The column totals provide estimates for the target region Y values.

The second set of tables show the calculations used in the EM algorithm and to calculate deviance.

EM Algorithm using X1

Step:

Lambda (1)

λ	t					Σ
Xt			Xt			
ΣYt	=IF(Step=1,Yt-1,Yta)					=
ΣAt	=At					row sum
newla	newla = (sum Yt)/(sum At)					

H step (1)

μ	t					Σ
1						
2						
3						
4	mua = newla * Ast					musa =
5						row sum
6						
7						
8						
9						
Σ	muta = col sum					30.000

Source Deviance

	Y_{sb}	Dst	Dst'
1		=IF(estYsb	=IF(est
2		b=0,musa	Ysb=0,
3	estYsb	estYsb,es	musa-
4	=Ysa	tYsb*LN	estYsb,c
5		(estYsb/	stYsb*L
6		musa)-	N(estYs
7		(estYsb-	b/musa))
8		musa))	
9			
Ds		=2*col sum	

E step (1)

est Yst	t					Σ
1						
2						
3						
4	Ysta = (mua/musa) * Ys					Ysa = row
5						sum
6						
7						
8						
9						
Σ	Yta = col sum					30

Target Deviance (est Yt)

	Y_{tb}	Dst	Dst'
1		=(Yta*L	=
2		N(Yta/m	Yta*LN(
3	=Yt	uta)-(Yta-	Yta/muta
4		muta)))
5			
Dt		=2*col sum	

The cell named "Step" controls the iterative process. If the value of this cell is set to one, then iteration does not occur, if the value is set to any value other than one, then iteration occurs. The key to this is the table named "Lambda".

Lambda

The first row contains the ancillary values. The second row contains the target region Y value estimates. If Step = 1 then this row contains the Areal Weighting Interpolation target region estimates (Y_{t-1}), otherwise the values are taken from table "E step" (Y_{ta}). This table is calculated using the second row of Lambda and hence forms the iterative loop.

The third row is a copy of the target region areas. The fourth row calculates the lambda values (see 3.2.3). This calculation has to be set up specifically for the regions being used.

M step and E step

Table "M step" produced Table 4.7. The cells of this table are calculated using (3.2.2). Table "E step" produced Table 4.8 and implements equation 3.2.4.

Deviance Tables

The deviance tables use (3.3.2) and (3.3.3) to calculate source and target deviance respectively. McCullagh and Nelder (1988) show that for the Poisson distribution the second term usually sums to zero (section 2.5). The deviance tables were set up so that the second term was included in the first deviance calculation and excluded in the second deviance calculation. This arrangement provided a measure accuracy of the spreadsheet design.

Appendix 2: MapBasic Application Design

Listed below is the main application used to implement *Enhanced Areal Interpolation*. This listing is partly written in MapBasic. Most of the listing is a brief description of the various steps.

```
*****
' Program : GWINT.MB                      Author: Gordon Wragg    1994  12/12
'
' This program designed to implement an enhanced areal interpolation process
' based on the Poisson distribution and the EM algorithm.
'
' The user specifies two open tables. The first, called the source table
' contains a column of data that the user would want interpolated to the
' second table which is the target table. The target table must have a
' column of auxiliary data that is distributed similarly.
' The program will use this information to do the interpolation.
'
*****

' a file of standard definitions is included
Include "mapbasic.def"
Include "menu.def"

' also include file to use "auto-load" library.
Include "auto_lib.def"

' Include functions
Include "col_type.mb"
Include "colmax.mb"
Include "gwhlp.mb"

' Define some constants
These define constants used in dialog boxes.

Define temporary table names.

This section declares subroutines and functions.

' define global variables
```

```
*****
' Program starts with 'Main', and execution begins here. Put up a menu, then
' the program will "go to sleep" and wait for the user to select a menu item.
' When selected, the appropriate subroutine is called.
*****
```

Sub Main

```
' Need to Assign global variables for auto load library
gsAppFilename = "gwint.mbx"
gsAppDescription = "gwint"
```

Put up installation message.

Create menus, the subroutines are activated when a menu option is chosen.

End Sub

```
*****
' SubRoutines
*****
```

Sub About

```
Dialog Title "About Interpolate"
```

This is a dialog box that provides basic information about the program.

End Sub

```
*****
' Interpolate
*****
```

Sub Interpolate

```
'This procedure displays a dialog, for the user to identify
'the source table and column to interpolate,
'and the target table containing the auxiliary variable column.
```

Set up Error checking.

Cleanup any temporary tables left over from previous runs.

'initialise variables

'check that at least 2 tables are open

'check that mapper is front window

Get area units being used.

Now display main dialog box.

Handler routines are set up to handle choices.

Table Handler looks after table and variable options.

If the user changes stop limit or number of iterations then the new values are recorded.

If the cancel button is used then exit.

Check that number of iterations > 0 and that stop limit > 0.00005.
Copy source and target tables and save with temporary names.

'now alter temporary source table for extra columns'

Call Split routine to split the source regions using the target regions.

Call EM algorithm routine.

Call Deviance routine.

Call Assemble routine to sum target region estimates and put into target table.

All done.

End Sub

' Table handler

Sub Tabhdler

This routine handles the choices for source and target tables, and variable columns.

If table1 index not zero then source table chosen so get name.

 If column1 index not zero then Y variable column chosen.

Modify dialog box to show choices.

If table2 index not zero then target table chosen so get name.

 If column2 index not zero then X variable column chosen.

Modify dialog box to show choices.

Check that source and target tables are not the same.

End Sub

' AllOkay

Sub AllOkay

This routine is called after the dialog box is closed and checks that columns are the correct data type to be used and that the source and target tables are different.

If an error is found a dialog box is displayed to advise the user and the main dialog box is redisplayed.

End Sub

' SplitUp

Sub SplitUp

This is the split routine.

The source regions are split with the Y values interpolated using areal weighting.

The area is calculated and added to the source table for each intersection region.

Add index to source table to point to associated target region.

'now make sure that source areas are all within target area

End Sub

```

*****
' Sub procedure: EM
*****

Sub EM

This is the EM algorithm routine.

Get number of intersection regions.

'Now set up the lambda flags
For each intersection region
    if the lambda flag is zero then
        set it up.

'now ready to commence EM algorithm
Do Until done
    record previous estimate

    'start M step
    For i = 1 to number of lambda flags
        Sum y estimate, areas and mu from source table where lambda flag
        equal to i into sumy, sumarea and musum
        record values
        calculate lambda = sumy/sumarea
        update source table with mu = lambda * starea.

    'M step done, now do E step
    For i = 1 to number of source table rows
        Sum mu, y values from source table where source index = i
        into musum and ysva.
        calculate y estimate= mu * ysva/ musum
        update values in source table.

    'check stop limit and number of iterations
    Calculate delta and check with stop limit.
    If maxdelta < deltamax OR count >= itmax then
        done = TRUE
    End If
Loop.

End Sub

```

```
*****  
' Deviance  
*****
```

Sub Deviance

```
' first calculate source deviance sdev  
For i = 1 to number of source regions  
    sum mu and y estimates where source region index = i into musum and  
    ysum.  
Put values into an array musum(i) and ys(i).  
If ysum = 0 then dev(i)=(musum - ysum)  
    sdev = sdev + (musum - ysum)  
Else dev(i)=(ysum * Log(ysum/musum) - (ysum - musum))  
    sdev = sdev + (ysum * Log(ysum/musum) - (ysum - musum))  
End If.  
Accumulate deviance.  
Next  
Calculate source deviance = 2 * deviance total.  
  
' now calculate target deviance tdev  
Calculate similar to source deviance.
```

Report deviance values.

End Sub

```
*****  
' Assemble  
*****
```

Sub Assemble

```
Add Y value column to target table and index target regions  
  
For each target region match index with that in source table  
    sum y estimates and update target table.  
  
Save target table.
```

End Sub