# MONTE CARLO SIMULATIONS FOR ESTIMATING UNCERTAINTY IN GIS-DERIVED BASE FLOOD ELEVATIONS ARISING FROM POSITIONAL ERRORS IN VECTOR DATA

By

SAMUEL HANCHETT HENDERSON

A THESIS PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE

UNIVERSITY OF FLORIDA

2000

# ACKNOWLEDGMENTS

I would like to express my sincere appreciation for all those who have helped me through this master's thesis. Particular thanks go to my advisor, Dr. Bon Dewitt, for selflessly providing an open door and constructive criticism. His observations contributed a great deal to this research. I also wish to thank my other committee members, Dr. Scot Smith and Dr. Timothy Fik, for their useful comments and suggestions.

Thanks go to my fellow graduate students who provided support and comic relief, and to 3001, Inc. for providing the data and means to carry out this study. I am indebted to Sandra Russ and Joan Jones for their warm humor and for keeping me on track and in line. Finally, I would like to thank my parents, William and Ann Henderson, for their love, support, and encouragement.

TABLE OF CONTENTS

Abstract of Thesis Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Master of Science

MONTE CARLO SIMULATIONS FOR ESTIMATING
UNCERTAINTY IN GIS-DERIVED BASE FLOOD ELEVATIONS
ARISING FROM POSITIONAL ERRORS IN VECTOR DATA

By

Samuel Hanchett Henderson

December 2000

Chairman: Bon A. Dewitt, Ph. D.
Major Department: Civil and Coastal Engineering

Geographic Information Systems (GIS) are increasingly being used to aid in spatially referenced decision making. GIS-derived data are frequently presented without measures of reliability, leading to false certainty in the quality of data. This thesis proposes software implementing Monte Carlo simulations for estimating uncertainty arising from positional error propagation in vector-based GIS. A generic application, the *Positional Error Simulator (PES)*, was developed in ARC/INFO to simulate error in source data sets, carry out spatial analysis, and summarize uncertainty in derived values.

This error model was applied to a predictive erosion hazards study conducted for the Federal Emergency Management Agency. This study used GIS to produce estimates of Base Flood Elevation (BFE) for coastal structures at 10-year intervals, 60 years into the future. The analysis is carried out on three data layers: coastal structures, flood zones derived from Flood Insurance Rate Maps, and the erosion hazard area consisting of the

current and 60-year projected erosion reference feature. Two representations of BFE are investigated: a zonal-based *nominal* elevation, and a smoothed or *interpolated* representation.

Traditional Monte Carlo simulations were applied to the Erosion Hazards Study in order to investigate the nature of error propagation in complex GIS analysis. The PES application was used to simulate error in source data sets and summarize variability in reported BFE attributes. Resulting errors are presented at the study-wide and high uncertainty levels, and discussed in reference to error propagation through data layer interactions and spatial functions.

It was shown that 90% of nominal BFEs returned errors of 1ft or less and 95% of interpolated BFEs had less than 0.5ft of error. Nominal and interpolated BFE errors were as high as 2.5 and 1.25ft respectively. Additional simulations were performed to consider error in individual source coverages, while holding others error-free. These tests allowed the identification of source layer contributions to output uncertainty levels. It was shown that the structure layer contributed the most to output uncertainty. Basic metrics were introduced to compare observed levels of error and expected levels based on individual layer contributions. This data showed that most errors accumulated as expected.

Simulation results can be used to express data quality in metadata statements, to appropriately represent spatial data, and to validate data collection and processing methods through a suitability framework. A major consideration is the time and computer storage requirements for implementing simulations. Care must be taken in the interpretation of output uncertainty estimates as the value of summary statistics are largely based on unpredictable realized error distributions.

# CHAPTER 1
# INTRODUCTION

Geographic Information Systems are increasingly being used to aid in spatially referenced decision making. GIS encompasses users from federal governments to environmental researchers, and the results of GIS analyses are used to support decision making in fields from emergency vehicle dispatch to environmental modeling. GIS provides efficient and largely automatic modeling of geographic phenomenon by allowing numerous datasets to be assembled, stored, combined and queried.

By generically handling spatial data, supporting an array of data formats, and not accounting for basic accuracy information, GIS is a hotbed of error propagation. Furthermore, the digital mapping environment has led to false certainty in data quality (Thapa and Bossler, 1992). Because basic data quality statements such as scale and resolution are neither retained nor used as limiting factors to the types of operations that may be carried out, GIS products may be applied to uses for which they were never intended (Goodchild, 1993). Lanter and Veregin (1992) have clearly summarized the problem.

> A GIS provides a means of deriving new information without simultaneously providing a mechanism for establishing its reliability. In such applications input data quality is often not ascertained, functions are applied to these data without regard for the accuracy of derived products, and these products are presented without an associated estimate of their reliability or an indication of the types of error they may contain. (Lanter and Veregin, 1992, p. 825)

The uncertainty problem is a major topic in the research field. Since the computerization of geographic analysis, researchers have been concerned with the presence, quantification, and visualization of errors in spatial databases. In the last decade, the National Center for Geographic Information and Analysis (NCGIA) and the University Consortium for Geographic Information Science (UCGIS) have highlighted spatial data quality as a key research initiative. The Federal Geographic Data Committee (FGDC) has required data quality information to accompany all spatial data in accordance with the Spatial Data Transfer Standard and the Content Standard for Digital Geospatial Metadata (FGDC, 1998a).

There are few practical tools for dealing with uncertainty in GIS (Veregin, 1989a; Canters, 1997; Hunter and Goodchild, 1999a). In search of automated solutions, many researchers have concluded that traditional mathematical error propagation techniques cannot be applied to complex GIS analysis and have focused on computer simulation methods as a practical alternative. Despite significant advances, comprehensive tools have not been implemented in popular GIS software.

This thesis provides a method for estimating uncertainty in GIS-derived values arising from positional errors in vector data. Similar error analyses have been conducted on simple GIS operations such as the determination of area and length, point-in-polygon overlay, and buffer operations. There has been no application of a vector-based simulation model to a substantial GIS analysis consisting of chains of multiple functions and data layers.

This thesis investigates attributes derived in an Erosion Hazards Study conducted by the Federal Emergency Management Agency. This study (discussed in depth in Chapter 3) derived two measures for the Base Flood Elevation of coastal structures: a zonal-based *nominal* elevation and a smoothed or *interpolated* representation. In addition, a model was implemented to estimate these variables at 10-year intervals over a 60-year period. This study is a complex GIS analysis in which error propagation is expected to play a major role on the quality of derived values.

The goal of this thesis is to apply traditional Monte Carlo simulations to the Erosion Hazards Study in order to investigate the nature of error propagation in GIS. The underlying hope of this research is that the algorithms introduced and problems revealed will lead to new directions for future research. To address these goals, the following questions are investigated.

1. What levels of uncertainty exist in derived Base Flood Elevations?
2. What are the potential uses and limitations of the proposed error model and the resulting uncertainty data?
3. Which data layer contributes the most to output uncertainty?
4. How do errors from individual inputs accumulate during spatial analysis?

The proposed error model is formalized into a generic application, the *Positional Error Simulator (PES)*, for the simulation of vector data and estimation of uncertainty in GIS derived information. This proposed software is a substantial improvement on a similar model developed by Hunter and Goodchild in 1996. Although the theories and algorithms presented in this thesis can be applied in any GIS, they were specifically implemented in ARC/INFO. This choice is supported by the popularity of the software and the availability of the Arc Macro Language (AML). ARC/INFO terminology will be used throughout this document.

This thesis is divided into six chapters and one appendix. Chapter 2 provides background on spatial data quality and highlights the importance of positional data quality. Mathematical and simulation-based methods for error propagation are introduced. A brief review of related research is presented, concluding with a description of Hunter and Goodchild's uncertainty model. Chapter 3 discusses the algorithms used to derive base flood elevations for the Erosion Hazards Study. Chapter 4 presents the proposed error model, the practical software written for ARC/INFO, and describes how the model was applied to evaluate error propagation in the Erosion Hazards Study. Chapter 5 reports the results of the case study error analysis and goes over practical uses for the uncertainty information. A discussion of the results, and overall evaluation of the model is also provided. Chapter 6 summarizes work done and conclusions made. The appendix contains AML and C code written for this study.

# CHAPTER 2
# BACKGROUND

Before the introduction of a positional error model written for this study, some background information is presented. A general discussion of spatial data quality is offered, highlighting the importance of positional information and clarifying terms associated with error-description. Next, the concept of error propagation is introduced, drawing distinctions between traditional variance propagation and simulation methods. Lastly, a review of related research is presented, including a description of a similar vector based error model developed by Hunter and Goodchild.

## Spatial Data Quality

The term *error* is used loosely in GIS literature. From discussions of image classification accuracy to topological problems such as overshoots, dangles, and slivers, *errors* may exist in virtually every aspect of a spatial database. Fundamentally, GIS data are made up of position and attribute information. The positional or spatial component defines *where* on the earth's surface features exist while the attribute or thematic component describes *what* the objects represent. Data quality refers to how well position and attribute information portray real-world entities. The differences between the physical truth and the digital representation, whether they are positional or attribute in nature, are generically termed *errors*.

A great deal of research has focused on describing, quantifying, and representing these differences, resulting in an abundance of error descriptors. Error, reliability, uncertainty, accuracy, precision, resolution, and scale are often used interchangeably and sometimes incorrectly in reference to spatial data quality (Goodchild, 1993).

All spatially referenced data are subject to errors introduced during the data collection process. The magnitude of error is largely dependent on the data collection method, and to a lesser degree on the refinement of the processing methods (Goodchild, 1993). Thapa and Burtch (1991) recognizes primary and secondary methods of data collection in GIS. Primary methods include those that directly measure the position of features, including surveying, GPS, photogrammetry, and remote sensing. Secondary methods involve obtaining geospatial data from existing documents through digitizing or scanning. GIS databases are often composed of data from both primary and secondary sources, and therefore are subject to varying types and degrees of error.

Data quality has formally been described by various government organizations including the Federal Geographic Data Committee. The Content Standards for Digital Geospatial Metadata (FGDC, 1998a) and the Spatial Data Transfer Standard (USGS, 1997) refer to five components of spatial data quality: *lineage*, *positional accuracy*, *attribute accuracy*, *logical consistency*, and *completeness*. Lineage refers to the documentation of data collection and conversion methods leading to a data set's final digital representation. Positional accuracy describes the spatial extent of uncertainty in a digital data set and may be divided into horizontal and vertical components. Attribute accuracy consists of numerical estimates of variability in object properties. Logical consistency refers to the topological correctness of the dataset. Completeness describes rules used in the creation of the data set such as sampling methods, omissions, and definitions. A final spatial data quality component can be identified as temporal accuracy. Although it is not listed as an independent component, the standards recognize its importance and call for its inclusion in each of the other quality statements.

The FGDC requires all government funded spatial data products to be accompanied by adequate metadata in accordance with the CSDGM. In addition, data users in the private sector are increasingly requiring metadata documents alongside GIS products. To produce these documents, users must provide estimates for the quality of GIS layers and derived products.

Although position and attribute components are often handled independently, both must correctly reflect the geographic truth in order to faithfully represent spatial data. The full functionality of GIS is only realized through the generic handling of multiple thematic layers over a common spatial extent. This requires all data to be referenced to a common coordinate system or positional measurement scale. Positional data quality is therefore a fundamental aspect of data quality.

## Positional Data Quality

Positional Error is the difference between a measured value and the true value. Because no measurements are exact, the true value is only a theoretical quantity, and similarly, all representations of error are estimates. The mean of a group of repeated observations is often used as an estimate of the true value.

### Error Types

Spatial data are subject to three types of errors: random, systematic, and gross blunders. Random errors are the perturbations in data primarily due to the inexactness of the measurement process, arising through variability in human observations and equipment. The variation in a human's vision while reading an instrument may cause fluctuations in the data around the hypothetically true value. Random errors follow the laws of probability; they are more likely to be small than large, and just as likely to be positive or negative (Wolf and Ghilani, 1997). Systematic errors are usually the result of instrument miscalibrations causing all values to be shifted to one side, creating an imbalance in the data. Blunders are simply mistakes in the observation or recording of a measurement.

Although all data entering a GIS are subject to all three types of errors this research assumes that the data have been adequately processed to identify and remove systematic errors and blunders. Data derived by secondary methods (i.e. scanning, digitizing) further complicate the detection and removal of non-random errors. Assuming only the presence of random errors in derived data is fundamentally incorrect but practically unavoidable without advanced models for error propagation in data conversion techniques. Stanislawski, Dewitt, and Shrestha (1996) did propose such a model for evaluating error propagation through digitizing trials and subsequent transformations using least squares. They were able to quantify absolute and relative accuracies and effectively model systematic errors. Generally, however, the lineage information needed to track systematic errors through various conversions is not available, nor computationally feasible. Random errors cannot be removed from a database. Further discussion of error in this thesis will refer only to random errors.

Accuracy and Precision

The concept of accuracy is closely related to quality although it takes on a formal definition in regards to positional quality. Accuracy is the nearness of a measured quantity to its true value (Wolf and Ghilani, 1997). Positional accuracy specifically refers to the nearness of observed values to the feature's true position in the same coordinate system (Drummond, 1995). Positional accuracy is often quantified by root mean square error or simply RMSE (Anderson and Mikhail, 1998; Drummond, 1995). RMSE values are calculated for each point as follows.

$$RMSE_X = \sqrt{\frac{\sum_{i=1}^{n}\left(X_{meas,i} - X_{true,i}\right)^2}{n}} \qquad RMSE_Y = \sqrt{\frac{\sum_{i=1}^{n}\left(Y_{meas,i} - Y_{true,i}\right)^2}{n}}$$

True coordinates are simply values attained through a higher accuracy method. Combining the X and Y components into a single error descriptor gives the following equation known as *circular*, *horizontal*, or *radial* RMSE (FGDC, 1998c).

$$RMSE_{CIRCULAR} = \sqrt{\left(RMSE_X{}^2 + RMSE_Y{}^2\right)}$$

The term *precision* has two different although related usages in the spatial data field. Most data quality research refers to precision as the number of significant figures in a measurement (Goodchild, 1993). For example the number 154.234 is more precise than the number 154.23. This is often referred to as *arithmetic* or *reporting precision*.

In terms of measurement theory, *precision* is the degree of refinement or consistency in a measurement process (Wolf and Ghilani, 1997). It describes the closeness of repeated observations to one another, regardless of their relation to the true value. Although a group of observations may be very precise, systematic effects may have caused all measurements to be of poor accuracy (Wolf and Ghilani, 1997). Standard deviation is typically used to measure precision (Drummond, 1995) and is calculated by

$$\sigma_x = \pm\sqrt{\frac{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}{n-1}}$$

where $\bar{x}$ is the most probable or mean value, $x_i$ is the $i^{th}$ measurement, and $n$ is the number of observations.

Accuracy Assessment

The Spatial Data Transfer Standard (SDTS) identifies four methods for assessing positional accuracy of a digital dataset (USGS, 1997). The first, by *deductive estimate*, calls for practical estimates of errors in source data and assumptions made about error propagation. Secondly, *internal evidence* such as redundancy checks or adjustment statistics may be used. Third, *comparison to source*, calls for the visual comparison of derived data from source data if such a situation is possible. The fourth and preferred test of positional accuracy is by comparison to an *independent source of higher accuracy*.

Traditional accuracy standards such as the National Map Accuracy Standard (NMAS), the American Society for Photogrammetry and Remote Sensing (ASPRS) Accuracy Standards for Large-Scale Maps, and the FGDC's Geospatial Positioning Accuracy Standards, are formal methods for implementing the higher accuracy tests. These standards call for independent treatment of horizontal and vertical accuracy assessment. Only the horizontal component will be discussed here.

Although the current version of the National Map Accuracy Standards (NMAS) was developed in 1947, it is still widely used. NMAS calls for the testing of well-defined points by a source of higher accuracy. In order to meet the specification, 90% of test points must be within a certain distance, determined by map scale, of the higher accuracy location. For map scales larger than 1:20,000, the allowable error tolerance is 1/30 of an inch at map scale. For scales smaller than 1:20,000, error must be within 1/50 of an inch (USGS, 1999)

The ASPRS accuracy standards also link accuracy to map scale. Unlike the NMAS, the ASPRS standards provide numerous class levels and acceptable error tolerances for map scales larger than 1:20,000 (ASPRS, 1990). In addition the standards require a minimum of 20 checkpoints.

Part 3 of the Geospatial Positioning Accuracy Standards is called the National Standard for Spatial Data Accuracy (NSSDA) (FGDC, 1998c). The NSSDA is the current standard for digital data. The standards recognizes that in a digital environment, map scales can easily be modified and quickly loose their original meaning. To describe accuracy of digital data, the NSSDA requires the reporting of RMSE at the 95% confidence level. The standards also require at least 20 well-defined checkpoints.

Although positional accuracy standards can be used as a basis for data quality statements, data producers usually lack the budget and time required to collect a sufficient number of check points. Furthermore, the abstract nature of many data sets precludes the identification of well-defined points. Rather, data quality statements are usually based on deductive estimates. The user is therefore required to make assumptions about source errors and the accumulation of the errors during data conversion and processing.

## Error Propagation

Error propagation is the technique used to evaluate errors in computed quantities based on an understanding of how these errors accumulate through computational functions (Anderson and Mikhail, 1998). The distribution of original errors through a computational process will result in an augmentation of error in the result (Wolf and Ghilani, 1997). Standard deviation is traditionally used to describe error in observed quantities and output values derived through functions of those quantities (Anderson and Mikhail, 1998).

Formulas can be developed for the mathematical evaluation of error propagation, formally known as *variance propagation*. The *special law of propagation of variances* can be used to evaluate error transmitted through a single function Z. With *a,b,c…n* representing be observed variables with known random and independent errors $\sigma_a$, $\sigma_b$, $\sigma_c$, $\sigma_n$, the error in Z can be derived by partial derivates (Anderson and Mikhail, 1998).

$$\sigma_z = \pm\sqrt{\left(\frac{\partial f}{\partial a}\right)^2 (\sigma_a)^2 + \left(\frac{\partial f}{\partial b}\right)^2 (\sigma_b)^2 + \left(\frac{\partial f}{\partial c}\right)^2 (\sigma_c)^2 + \cdots + \left(\frac{\partial f}{\partial n}\right)^2 (\sigma_n)^2}$$

To propagate error through multiple functions and to handle correlated variables, the *general law of propagation of variances* must be used. (Wolf and Ghilani, 1997; Anderson and Mikhail, 1998). This is accomplished by computing a variance-covariance matrix,

$$\Sigma_{zz} = A\Sigma A^{T}$$

where $\Sigma_{zz}$ is the covariance matrix for the function, $A$ is the coefficient matrix of partial derivates known as the Jacobian matrix, and $\Sigma$ is the covariance matrix for the measurements. For non-linear functions, the partial derivatives in the Jacobian matrix can be estimated by a first-order Taylor series expansion (Wolf and Ghilani, 1997).

Openshaw (1989) recognizes that, in theory, existing mathematical error propagation techniques can be used to evaluate error in GIS derived products. However, in practice, he notes that the complexity of spatial operations and the combination of these functions leave formulas very difficult if not impossible to develop. For complex spatial analysis, Openshaw (1989), Goodchild (1995), Drummond (1995), and Kiiveri, (1997) recognize that the propagation of error can only be practically modeled through Monte Carlo simulations.

## Monte Carlo Simulation

Monte Carlo methods are used to simulate real-world situations using random-number generators. Chou (1969) identifies two requirements for applying the method, a model of reality and a mechanism to simulate the model. Applied to evaluating error propagation, the model is the expected probability distribution of error, and the simulation mechanism is a random number generator. Goodchild (1995) states Monte Carlo methods can be used to estimate uncertainty in GIS outputs by simulating inputs with distorted datasets or *realizations* based on their estimated uncertainties. Although these realizations are fictitious, they are derived from the expected probability distribution and therefore are equally possible representations. By replacing observed data with appropriately distributed random data, and allowing each simulation to progress through an identical series of functions, the variability in results can be summarized to assess the reliability of GIS output.

Openshaw (1989) has outlined a generic Monte Carlo approach for evaluating uncertainty in GIS – derived products.

1. Decide what levels and types of error exist in source data.
2. Replace the observed data (position and/or attribute) by a set of random variables drawn from appropriate probability distributions.
3. Apply a sequence of GIS operations to the Step 2 data.
4. Save the results.
5. Repeat Steps 2 to Step 4, N times.
6. Compute summary statistics.

Openshaw realizes that, in many cases, the magnitude and/or distribution of errors in source data may not be known. However, reasonable assumptions about input errors will still indicate plausible boundaries for output data quality. He also states that a key problem is determining the appropriate value of *N*. If too few iterations are used, the validity of summary statistics may be in question. On the other hand, increasing *N* may result in additional computational time without adding statistical significance. Openshaw recommends a minimum of 20 to 30 iterations to validate statistical summaries.

The simulation method can be universally applied to virtually any data type and anlaysis, despite its level of complexity (Canters, 1997; Openshaw, 1989; Goodchild, 1995). In addition the concept is logical and easily grasped by novice users. Furthermore, the method does not require the use of complex mathematics.

The major cost to the simulation method is the increase in required computer processing time. As most GIS production schedules are strained to carry out an analysis *one* time, repeating the analysis *N* times is costly and in many cases not feasible. The primary advantage of mathematical models over simulations is that they can be applied on-the-fly and can be applied to new situations of the same function (Huevelink, Burrough, and Stein, 1989)

There are numerous applications for the simulation-derived data. The results may accompany each database at the record layer and used as estimates in other analyses. The results can be summarized for the inclusion in metadata documents as data quality statements. The results can be used to correctly represent the quality of GIS-derived data by limiting the precision of reported attributes. Also, Openshaw (1989) recognizes the contribution of output errors from different inputs may be isolated by holding some inputs as error free. This information would prove invaluable to project planners and will be discussed in detail in Chapter 5.

## Related Studies

A common goal among researchers in the data quality field is to establish models that describe types and magnitudes of errors in source data, track errors through spatial functions, and report uncertainty information alongside GIS outputs (Goodchild, 1989, 1993; Guptill, 1989). This goal remains an ideal, as few methods exist for the automatic handling of uncertainty information. When organizing the large body of literature dealing with data quality and GIS, broad distinctions can be made based on data type (raster or vector), component type (positional or attribute) and model type (simulation or variance propagation). Goodchild (1995) and Lanter and Veregin (1992) have stated that examining spatial and thematic accuracy independently is not desirable. However, due to the complexities of the actual operations in use, the derivation of all-inclusive errors models is largely implausible. Although this research is only concerned with positional error propagated through spatial functions in a vector environment, many attribute and grid-based studies have contributed to the derivation of vector-based models. In order to place this research among others, a brief review of related work follows.

### Raster-Based Studies

The majority of research has concentrated on the propagation of errors in the raster environment. It is generally recognized that error propagation in raster or field data is somewhat easier than in vector or object based data (Goodchild, 1989). In many of these studies, the complexities of error propagation are reduced by holding grid cell positions fixed and only considering errors in the values or attributes of grid cells, allowing the accumulation of errors during raster overlay to be assessed by traditional mathematical error propagation on a cell-by-cell basis.

Research dealing with error propagation in the raster environment falls into two fundamental categories: thematic and elevation studies. In thematic related work, Heuvelink, Burrough, and Stein (1989) applied standard mathematical error propagation techniques to evaluate simple arithmetical operations in a raster environment. By only considering basic operations and applying them on a cell-by-cell basis, output errors were modeled with Taylor series expansions and traditional variance propagation. Fisher (1991a) uses simulations to find errors in monetary costs associated with soil map-unit inclusions. Lanter and Veregin (1992) propagate errors through raster overlay using variations of the traditional PCC (points correctly classified) index. Goodchild, Guoqing, and Shiren (1992) proposed an error model for thematic data based on stochastic simulations and used it to evaluate uncertainty in area and overlay operations on land-cover maps. Veregin conducted several studies on error modeling in a raster environment, concentrating on overlay (1989a and 1995) and buffer operations (1994,1996). In 1994 he used Monte Carlo simulation as a basis for creating formal mathematical models for error propagation. He argues that simulation modeling is too computationally intensive to be practically applied in GIS and that simulation studies have only led to anecdotal observations. Based on simulation results, he defines models that propagate error through the buffer operation. He argues that these models can then be applied to other buffer operations to achieve uncertainty information in real-time, without performing time-consuming simulations. Both Canters (1997) and Yuan (1997) use Monte Carlo simulations to evaluate uncertainty in area estimation based on errors in image classification. Arbia, Griffith, and Haining (1998) also use simulations to model error in basic raster overlay functions.

Much of the work in raster uncertainty propagation has focused on errors arising in data derived from digital elevation models (DEMs). Hunter and Goodchild have produced a considerable amount of research in this area. In 1994, they used simulations to evaluate uncertainty in areas burnt by forest fires in Portugal. These areas were primarily identified by a slope and aspect analysis of the DEM. In 1995b, Hunter and Goodchild reviewed various methods for evaluating the uncertainty of an elevation value interpolated from a DEM. In 1997, Hunter and Goodchild apply their model to slope and aspect measures. Ehlschlaeger, working with Goodchild (1994) also described uncertainty in elevations derived from DEM interpolation. Fisher (1991b) uses Monte Carlo simulations to evaluate uncertainty in viewshed areas. Lee, Snyder, and Fisher (1992) used simulations to model data errors in floodplain feature extraction. Wechsler (1999) built a practical set of tools in the ArcView GIS environment to evaluate basic DEM uncertainties including slope and aspect calculations through Monte Carlo simulations. Davis and Keller (1997) combined simulations and fuzzy models to evaluate slope stability.

Vector-Based Studies

Of particular relevance to this research are the few models proposed to deal with uncertainty in vector data. The *epsilon band* model is a general approach to describe uncertainty around lines. Although some differences in definition exist, the model fundamentally states that a line's digital representation will fall within some distance epsilon from the true line, the likelihood of which is described by a probability density function (pdf). Dunn, Harrison, and White (1990) identify three common pdfs in use with the epsilon model: rectangular, bell-shaped, and bimodal. Blakemore (1984) used the epsilon approach to assess point-in-polygon overlay errors, by dividing the polygons into zones described as *definitely in*, *possibly in*, *possibly out*, and *definitely out*. Dunn, Harrison, and White applied the model to digitizing trials in order to assess uncertainty in the areas of polygons. Goodchild (1995) criticized the epsilon band as not meeting the requirements of a formal error model by providing no means for generating distorted versions of the data.

Realizing that the majority of digital vector data has entered the GIS through a digitization process, it is logical to assume that the positional uncertainty present in source data will follow magnitudes and distributions revealed by repeating digitizing trials. Studies of digitizing error are therefore important to describe the positional uncertainty present in source data, leading to more plausible models of error propagation.

Bolstad, Gessler, and Lillesand (1990a and 1990b) used linear models to estimate errors in the digitization process due to map media, point type, and registration effects. In their study, four operators repeatedly digitized the same map. Differences between the digitized points and means, revealed a normal bell shaped distribution. Maffini, Arno, and Bitterlich (1989) investigate error through a series of digitizing trials. They used the results of these tests to make general statements about the nature of error, similar to the epsilon-model. Ehlers and Shi (1996) use traditional variance propagation models to describe positional uncertainty in points collected through digitizing. Their vector model of digitizing error is based on the assumption of normally distributed independent errors in the coordinates of points. Shi (1998) further developed his statistical model to incorporate uncertainty from one- to N-dimensional space, allowing for the inclusion of spatio-temporal relationships. Leung and Yan (1998) provide a similar set of point-based models for describing positional uncertainty in vector databases. Like Shi, they also assume a circular normal distribution around points and line vertices to describe error in points, lines, and polygons. They apply their model to evaluate uncertainty arising in basic spatial functions such as length and area.

A few authors disagree that such a point-based error model can be applied to lines and polygons. Goodchild (1993) argues that in regards to digitizing errors, positional uncertainty in a line cannot be adequately represented by independent errors in line vertices. Rather, he argues, the digitizing operation produces vertices whose error is highly correlated. Also, misregistration errors will tend to produce uniform shifts that can only be described by assuming some degree of correlation in the digitized line. Openshaw (1989) also argues that lines are subject to correlated errors.

Emmi and Horton (1995) used Monte Carlo simulations to evaluated error propagation in an assessment of seismic risk. They considered errors in the position of ground shaking intensity boundaries. However, in order create the random perturbations of vector data they did not draw on rigorous statistical theory. Instead they adopted a software-friendly solution of densifying arcs, applying a weed tolerance to shift arcs, and generalizing arcs. This in effect randomized the position of arcs and points but does so in a manner that cannot guarantee normally distributed data.

Hunter and Goodchild's Vector Uncertainty Model

The practical research of vector-based error propagation utilizing simulation techniques has come from Hunter and Goodchild. In 1996 Hunter and Goodchild proposed a model that creates distorted vector datasets based on estimated positional errors. The algorithms in this work are largely based on their previously developed model for raster data. These random datasets or *realizations* can then be used in a simulation model, which in turn can assess error propagation through spatial analyses and provide an estimate for uncertainty in end products. The model makes two assumptions: 1) that the error at line vertices behaves in a circular normal distribution, and 2) that the error's x and y components are independent. The model produces realized vector files through the use of two normally distributed error grids: one for each x and y components. By overlaying the error grids with the original vector data, a distorted but equally likely version is created.

Hunter and Goodchild apply the model to assess uncertainty in polygon areas, point-in-polygon overlay, and vector to raster conversion (1999a). After estimating positional errors present in the polygon vertices based on source map scale and digitization trials, 20 realizations of the polygon coverage were created. To represent uncertainty in reported areas, they calculated the mean and standard deviation for each polygon.

The vector uncertainty model of Hunter and Goodchild is available as AML and C code on the author's website. The software produces data realizations from ARC/INFO coverages based on estimated input errors. Because the positional errors are generated and stored in the ARC/INFO GRID format, the algorithm has several limitations. The user is required to choose a grid-cell spacing as input to the model. Any points, lines, and nodes falling within a single grid cell will have identical error distributions applied. If the grid spacing is too large, the realizations of certain features will be completely autocorrelated. If the spacing is too small, the algorithm slows considerably as x,y coordinate shifts are unnecessarily calculated for cells having no corresponding vector data. Furthermore, the application does not allow for any spatial analysis to be carried out, nor does it automate the calculation of uncertainty in derived attributes.

This thesis proposes a similar but improved vector uncertainty model, discussed in depth in Chapter 4. The proposed model avoids the use of GRID by generating and applying random errors in C. The choice of grid-cell spacing is avoided, ensuring all points, vertices, and nodes have independent random error. In addition the proposed model facilitates the application of any generic analysis and the automatic calculation and linkage of uncertainty information to GIS outputs.

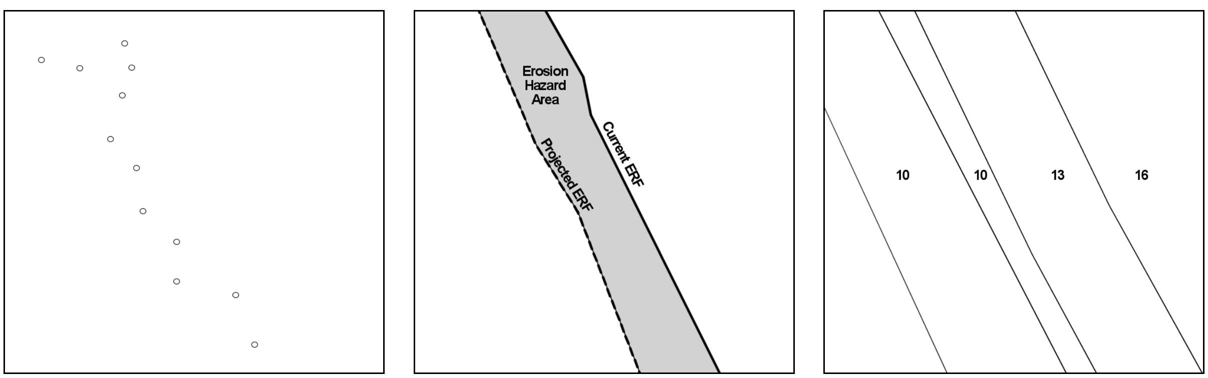
# CHAPTER 3
# EROSION HAZARDS STUDY

3001, Inc. recently completed a substantial GIS analysis as an integral part of the national *Evaluation of Erosion Hazards* study conducted by the Federal Emergency Management Agency (FEMA). The goals of the project were to evaluate the "economic impact analysis of erosion on coastal communities and on the NFIP [National Flood Insurance Program]" (FEMA, 1998, p. 1). This study compiled over 10 thematic layers for 27 counties in 18 coastal and Great Lakes states. 3001, Inc. collected GPS data on sampled coastal structures and attached over 100 descriptive attributes by overlaying assessment, parcel, flood zone, and erosion hazard layers. Using historical erosion rates, the study predicted relative positions of data layers over a 60-year period.

## Data Layers

This analysis focuses on three fundamental data layers from the Erosion Hazards Study: *Structures*, the *Erosion Hazard Area*, and *Flood Zones*. Figure 1 shows an aerial photograph over a sample study area in Dare County, North Carolina. The three data layers are overlaid on the image and depicted individually in Figure 2.


Figure 1:  Sample source data layers in Dare County, North Carolina.




a) Structures. b) Erosion Hazard Area. c) Flood Zones of Base Flood Elevation (BFE).

Figure 2:  Erosion Hazards Study source data layers.

The S*tructure* data set (Figure 2a) is a point coverage of sampled coastal buildings, collected with real time differentially corrected GPS (3001, 1998a).  This observation was usually made from the street, from which a bearing and offset was applied to shift the point to the building.  Although numerous structure attributes were collected for the study such as house number, address, and structure condition, only the x,y coordinates of the structure are of interest to this research.

The *Erosion Hazard Area (EHA)* (Figure 2b) is a polygon that represents "an area where erosion or avulsion is likely to result in damage to or loss of buildings and infrastructure within a 60-year period"(FEMA, 1998, p. 1). The EHA is bordered on the ocean-ward side by the *Current Erosion Reference Feature (CERF)*. The CERF line is a primary indicator of erosion that represents the "receding edge of a bluff or eroding frontal dune or the normal high-water line, or the seaward line of permanent vegetation" (3001, 1998b, p. 1). The landward extent of the EHA is the *Projected Erosion Reference Feature (PERF)* line representing the estimated 60-year position of the ERF based on historical erosion rates. It is expected that the *CERF* will migrate to the position of the *PERF* over the 60 year period.

Figure 2c shows the *Flood Zone* layer derived from FEMA's Flood Insurance Rate Maps (FIRM). FIRM maps identify degrees of risk in flood prone areas. This data set is a polygon coverage delineating zones of varying *Base Flood Elevation (BFE)*. BFE is an estimate of the water height in the event of a 100-year flood, decreasing in the landward direction from the coastline. For example, if a person was standing at a point with an elevation of 10 feet, and was in a zone with a BFE of 15, the person would be standing in five or more feet of water during a 100-year flood (3001, 1998c). All BFE values are in feet relative to the vertical datum of NGVD29.

## Erosion Hazards Analysis

The portion of the Erosion Hazards Study of relevance to this research involves the attachment of BFE values to coastal structures. Finding the BFE of each structure is a simple case of vector overlay: However, two additional analyses were carried out which complicated the study. The first task involved interpolating the location of flood zones over the 60-year span. The second task was an attempt to represent the stepped-nature of polygon BFEs as a more realistic smooth line.

Just as the CERF is expected to progress into the PERF over the 60-year span, the flood zone polygons are also expected to move accordingly. In order to estimate the BFEs of structures 60 years into the future, flood zone polygons needed to be shifted landward, the magnitude and direction of which being controlled by the erosion hazard progression. Ideally, this was to be accomplished by digitally migrating the flood zone polygons in 10-year intervals, resulting in six additional datasets (10,20,30,40,50, and 60-r), allowing each structure's BFE to be determined over the 60-year period by overlay. However, the algorithms involved in such a non-uniform migration of flood zone polygons proved to be too complex and generated numerous topological problems. Project members agreed that instead of shifting polygon zones landward, the same effect was accomplished by migrating the structures points ocean ward. This migration would also follow the erosion hazard progression although in the opposite direction. Although the structures do not in reality move toward the ocean, the relative temporal-spatial relationships between the structures and flood zones could be realized while allowing complex polygon layers to remain static.

To facilitate this relative structure migration, a *transect* was defined as a line through the structure perpendicular to the CERF. This was accomplished in ARC/INFO with the NEAR algorithm, which finds the position on the line nearest to the structure, therefore resulting in the perpendicular point of intersection. Each structure was assigned an EHA *width* defined as the transect length falling within the EHA (Figure 3). It is expected that a structures erosion-risk will progress along the transect expanded over the width of the EHA. Once the EHA width is found, and the azimuth of the transect line is known, projected structure positions are calculated along the transect such that the 60-year structure is the distance of the EHA width from the original position, and the 10-year interval positions are equally spread in between (Figure 4). Figure 5 shows how the geometry of the EHA controls the magnitude and direction of the progression, and in turn, the projected structure positions. By overlaying each structure with the flood zone polygons, the structures BFE can be determined over the 60-year period. During the design phase of this algorithm, tests were conducted that verified this method returned the same result as shifting flood zone boundaries landward.


Figure 3: Structure transects and EHA width.

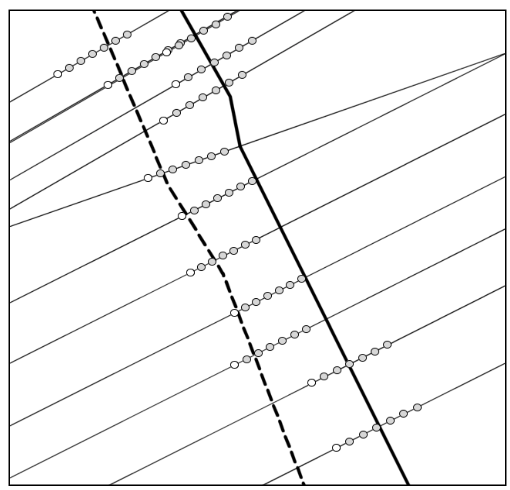


Figure 4: Projected structure positions for years 0, 10, 20, 30, 40, 50, and 60.

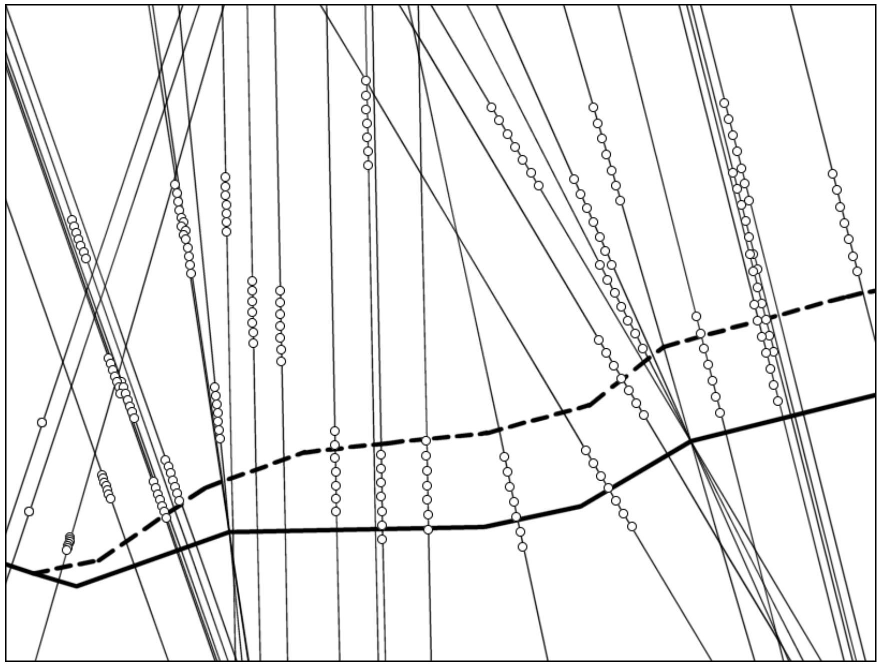


Figure 5: Projected structure positions from Glynn County, Georgia.

The representation of BFE as both integer and polygon-based values, incorrectly models its true nature. A sample profile along a structure's transect (Figure 6) reveals that at flood zone boundaries, the BFE will jump to a new level. In reality, however, it is expected that the BFE will gradually decrease as it progresses landward. A model was introduced to smooth the stepped nature of BFE into a more realistic representation, referred to as *interpolated* BFE. To distinguish between the two measures of BFE, the original, zone-based values will be referred to as *nominal* BFE (Figure 6).

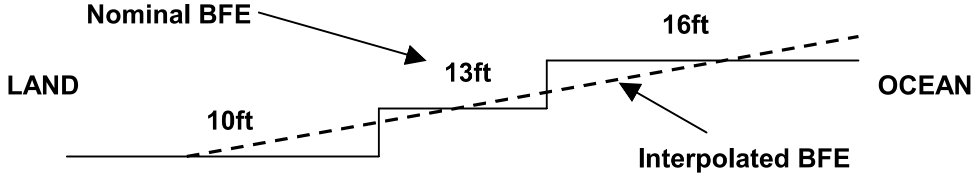



Figure 6: Profile along structure transect showing nominal and interpolated BFEs.

The calculation of interpolated BFEs was accomplished in ARC/INFO using a Triangulated Irregular Network (TIN). The coded polygon BFE was assumed to represent the elevation at the zone's *center*. Bordering polygon arcs were coded with the average of adjacent polygon BFE values and converted into a TIN with the ARCTIN function. Figure 7 shows the conversion of original BFE polygon codes to coded arc elevations (treated as contours) resulting in a representation of interpolated BFE. Because the TIN model is a continuous surface, it allowed the derivation of interpolated BFE anywhere in the study area.

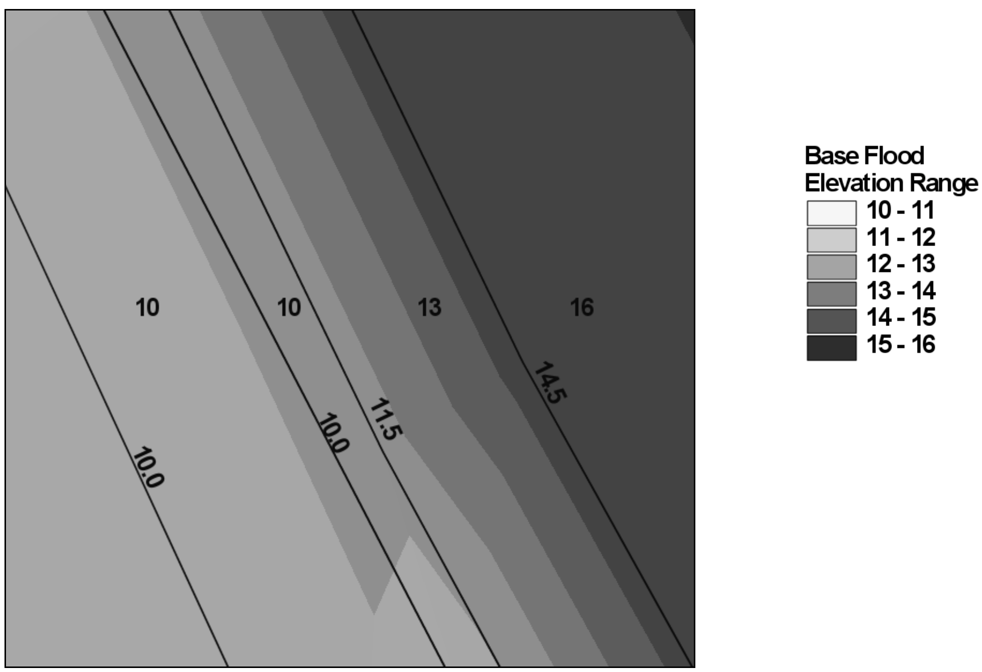



Figure 7: Interpolated BFE TIN surface and arc elevations.

By overlaying the TIN with projected structure positions at each 10-year interval, interpolated BFE values were derived for each structure over the 60-year period. Figure 8 shows the resulting spatial relationships of all output data. Once the projected structure positions are calculated nominal and interpolated BFE values are attached based on spatial overlay with the flood zones and TIN surface respectively. The script *fema.aml* provided in the Appendix was written to carry out the Erosion Hazards analysis. Given the three source coverages representing Structures, Flood Zones, and the Erosion Hazard Area, the AML calculates nominal and interpolated BFEs for each structure over the 60-year span at 10-year intervals. Table 1 gives sample output for a single structure. The next chapter describes the methods used to estimate uncertainty in these values.

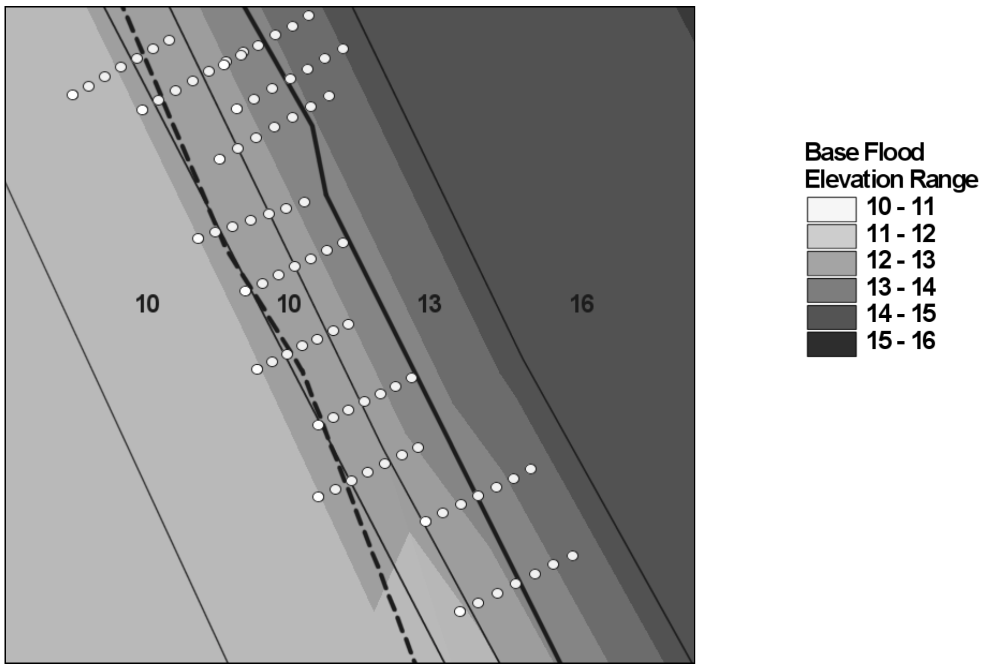



Figure 8: Spatial distribution of output data.

Table 1: Sample structure output data.

| | Year 0 | Year 10 | Year 20 | Year 30 | Year 40 | Year 50 | Year 60 |
|---|---|---|---|---|---|---|---|
| Nominal BFE | 10 | 10 | 10 | 10 | 10 | 13 | 13 |
| Interpolated BFE | 10.977 | 11.088 | 11.2 | 11.315 | 11.429 | 11.617 | 11.925 |

# CHAPTER 4
# METHODS

In order to estimate the uncertainty in nominal and interpolated BFEs, an error model was developed and formalized into an application referred to as the *Positional Error Simulator (PES)*. PES is a menu-driven application for NT ARC/INFO version 7.2 written primarily in the Arc Macro Language (AML). Although this application will be applied specifically to the Erosion Hazards Study, it is a generic error model designed to work with any set of vector coverages, and any analysis driven by AML. Various component scripts were written in AML, C, and Avenue to create the PES model. AML and C source code is provided in the Appendix. All code and a brief tutorial is provided on the author's website at http://www.surv.ufl.edu/~sam or by email at samhhenderson@hotmail.com.

This research adopts Goodchild's definition of an *error model* as "a stochastic process capable of generating a population of distorted version of the same reality" (Goodchild, Guoqing, and Shiren, 1992, p. 87). Like Hunter and Goodchild's vector uncertainty model, the PES model allows for the creation of random data sets. However, the proposed algorithm is applied directly to the vector data, avoiding the choices of grid cell spacing, and insuring the creation of normally distributed random data. The PES model also improves on Hunter and Goodchild's model by applying a user-defined analysis to the perturbed data and summarizing variability in derived values.

This chapter describes the methods used for estimating uncertainty in GIS-derived values due to positional error propagation. The first section describes algorithms in the

PES model for creating data realizations, carrying out the simulations, and statistically summarizing uncertainty in outputs. The second section describes how this application was applied to the Erosion Hazards Study.

## Creating Data Realizations

The PES model assumes only independent random measurement error in the position of the points, nodes, and vertices that make up vector based digital data. Like the models used by Hunter and Goodchild (1996), Leung and Yan (1998), and Shi (1998), the same uncertainty existing around point features is extended to all vector data by applying identical algorithms to the nodes and vertices of line and polygon data. Error is introduced into each of *N* realizations, producing a population of hypothetical observations. At a single point, node, or vertex, the collection of all realizations (*N*) will return a circular normal distribution centered on the input coordinates.

In order to apply this model, the true nature of the digital representation must be overlooked. The digitization models mentioned in Chapter 2 have concluded that repeated digitization trials will generally reveal point-based circular normal distributions. Assuming the absence of systematic errors, by repeating the digitization trials, the mean coordinates can be calculated and used as the most probable estimate of the true value. However, it is highly unlikely that any GIS user has the time or budget to digitize a map a sufficient amount of times to attain most probable coordinates. Rather, a map is digitized once, the digital version becoming one observation of a population of possible representations. Therefore, it is incorrect to assume that by simply digitizing a line once, one can adequately describe the error around each point or vertex by centering on it a probability distribution. In some cases, as in surveyed parcel locations, redundant measurements are taken and adjustments made to insure the coordinate represented are in fact the most probable values. However, the vast majority of GIS data has not been collected through a rigorous surveying process. It must be assumed that the majority of digital data are single observations of the real world. To apply the error model, this fact must be overlooked. Although the consequences of this assumption could be substantial, they cannot be avoided without knowledge of the actual errors existing in spatial data.

Data realizations are created in ARC/INFO through the use of *generate* files. A generate file is an ASCII representation of an ARC/INFO coverage. Generate files consist of point, vertex, node, and label identifiers, and x,y coordinate pairs. The C program *randgen*.c (see Appendix) was written to create $N$ error perturbed generate files based on a user-specified 1-sigma ($\sigma$) error level.

Each point, line, or vertex in the database is simulated by $N$ possible realizations. For each realization, a random coordinate ($X_R$, $Y_R$) is computed from a random *azimuth* and *distance* applied to the original position ($X_O$,$Y_O$). *Azimuth ($\alpha$)* is a random variable generated by the C function *rand()* whose possible values are uniformly distributed integers from 0 to 359. Theoretically, a truly random azimuth should not be restricted to integer values. However, the 360 possible azimuths derived from this method prove to be sufficient in the realistic application of such an error model. *Distance (d)* is a normally distributed random variable calculated with the Box-Muller algorithm and scaled by the user's input error tolerance. The random coordinates are computed as follows.

$$X_R = X_O + d(\sin(\alpha)) \qquad Y_R = Y_O + d(\cos(\alpha))$$

Figure 9 shows 100 point realizations created with the proposed algorithm. Assuming a $1\sigma$ error of 10ft, approximately 68 points should fall within a 10ft radius of the original point. Applying standard multipliers for probable errors, 95 points should fall within about 20ft and 99 within about 26ft. The cross hair in the figure shows the calculated mean coordinates.

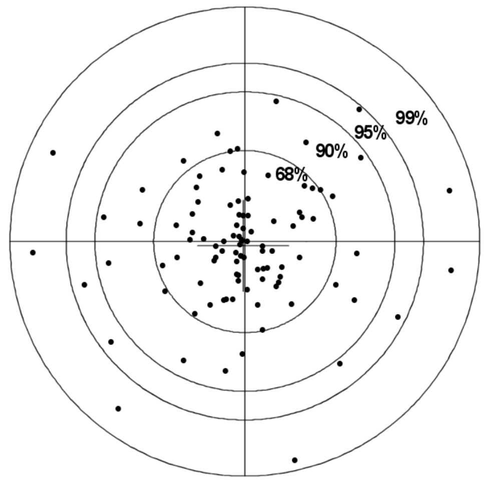


Figure 9: 100 point realizations generated by PES model.

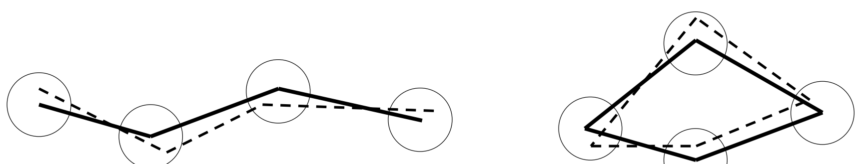


Figure 10: One realization of the point based model applied to lines and polygons.

The point-based approach above is similarly applied to the nodes and vertices of line and polygon datasets (Figure 10). Additional functions are added to the algorithm to retain topological relations of line and polygon data such as connectivity and adjacency. Polygon labels are not perturbed. The original labels are copied from the source dataset and *put* into the realized coverage to retain polygon topology.

Once *N* realized generate files are created, the feature identifier numbers are used to relate the original attribute information (not stored in the generate files) to the realized datasets. The end result is *N* copies of the original coverage complete with the original attribute information but perturbed by positional error.

The model requires apriori knowledge of existing positional errors in source data. This is accomplished by estimating the radius of a circle (centered on the input point) within which 68% of the error-introduced points will fall. Four methods for deriving such an error estimate were outlined in Chapter 2: deductive estimate, internal evidence, comparison to source, and independent source of higher accuracy. Where rigorous accuracy assessment tests have been conducted, these results should be used as inputs to the simulation. When higher accuracy data are not available, estimates of input data quality must be made.

The application of this point model to the nodes and vertices of lines and polygons may result in the creation of topological inconsistencies or unwanted overlaps. Hunter and Goodchild refer to these overlaps as *rips* (1995a) and *fractures* (1999a) (Figure 11).

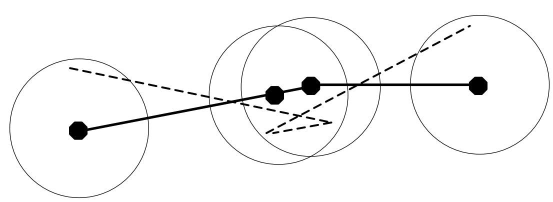


Figure 11: Rips in realized line data.

Rips may arise when estimated positional accuracies are greater than the distances between line vertices. In some cases, this situation will arise as the digital data are essentially misrepresenting the real world. For example, when stream-mode digitizing, the operator records points at equal intervals regardless of whether or not they represent actual changes in direction. In this case, rips can be avoided by removing extraneous details or *generalizing* the data prior to creating realizations. This technique is shown in Figure 12.

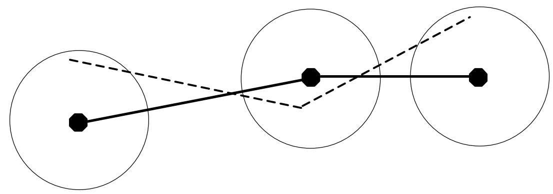


Figure 12: Application of feature generalization to remove rips.

In other cases, when estimated accuracies are large compared to line segment lengths and line vertices do represent actual feature changes, spatial autocorrelation must be present. To avoid rips, Hunter and Goodchild (1995a and 1996) proposed the introduction of autocorrelation into the randomization process. By design, their method would adjust random coordinates until a specific level of autocorrelation is reached. Figure 13 shows a sample application of this technique.

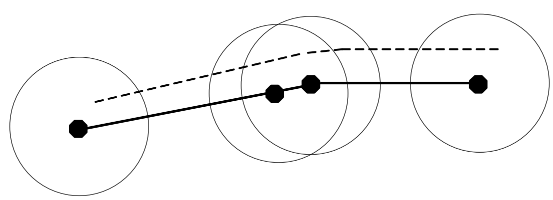


Figure 13: Introduction of autocorrelation to avoid rips.

However, since the 1996 paper, Hunter and Goodchild have not continued development of this technique (Hunter, 2000). In 1999b, they overcame the problem of rips by adjusting random coordinates until no overlaps exist, without constraining the results to a specific level of autocorrelation.

When errors are spatially autocorrelated, the realizations produced cannot be represented by a circular normal distribution, and the assumption that only random error exists cannot be made. Lacking practical methods to ascertain actual degrees and extent of spatial autocorrelation, the PES model assumes only the presence of independent random error. Therefore, data sets consisting of details more refined than error estimates must be generalized before applying the error model.

## Deriving Uncertainty Estimates

After randomization, each of *N* simulated data sets is copied into a unique workspace based on the realization number. To carry out the analysis, a user-defined AML is then run in each workspace. This produces *N* equally likely GIS results.

Outputs may be in the form of new or modified attributes attached to one or more source coverages. The PES model refers to these attributes as *items of interest*. Typical items may include x,y coordinates, length, area, and perimeter. Drawing on the work of Hunter and Goodchild (1995a, 1996, 1999a, and 1999b), the proposed model also summarizes variability in reported attributes with the mean and standard deviation statistics. Mean values are calculated in the model to reveal error distributions not centered on the original GIS values.

Once all realizations are created and the analysis is performed, coverage attribute tables are exported as ASCII files to facilitate the calculation of summary statistics. The C program *fat2rrt.c* (see Appendix) was written to sort attribute ASCII files into *record realization tables (RRT)*. An *RRT* table is created for every record or object in the database. Each row of the *RRT* table lists a simulated output from the model. There are therefore *N* rows per *RRT* table. Once *RRT* tables are created, the files are converted back to INFO tables and passed through the STATISTICS function to calculate mean and standard deviation. No mechanism is included for describing uncertainty in character fields. Figure 14 diagrams how uncertainty estimates are derived from data realizations in the case of polygon area queries.

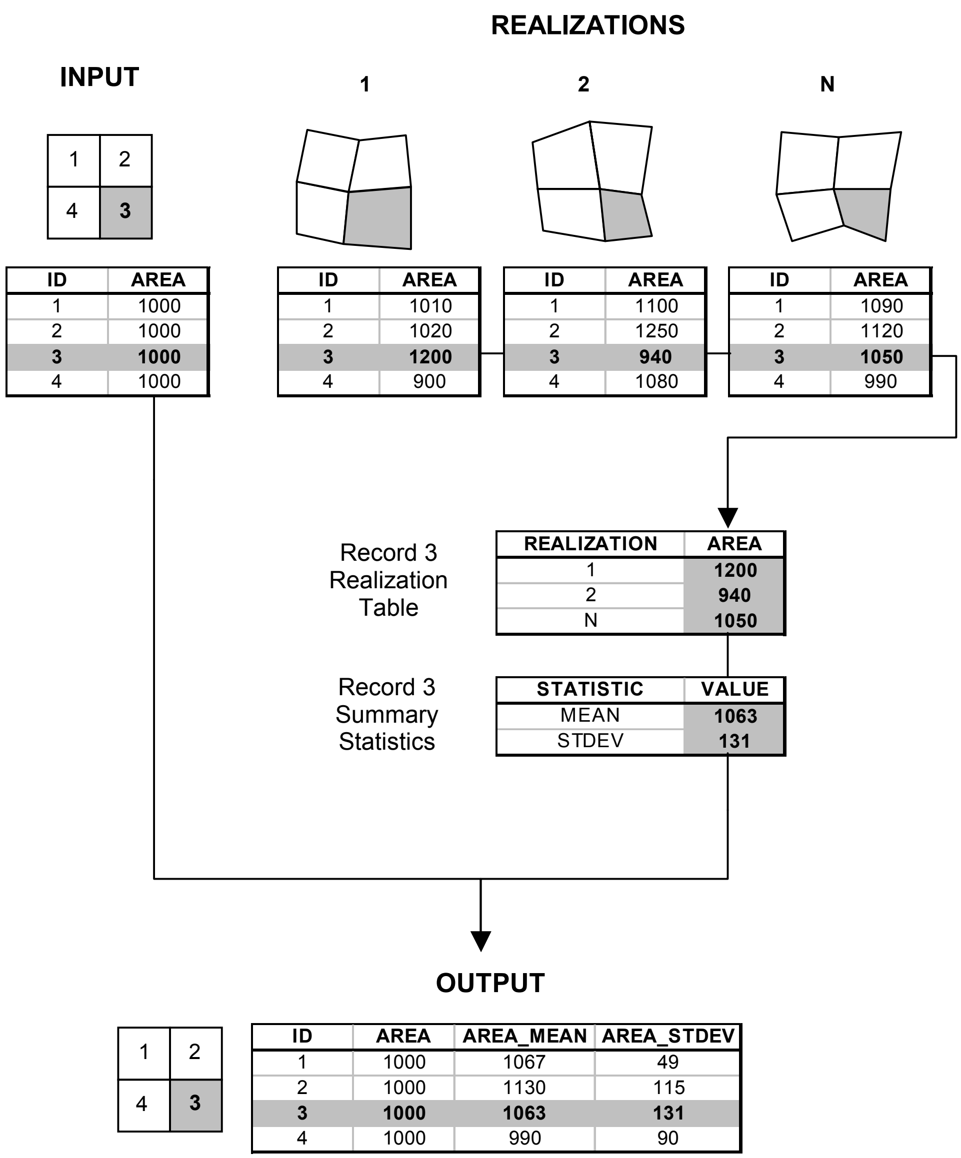



Figure 14: Calculation of summary statistics.

The PES model assumes only normally distributed error in derived attributes. When the resulting error distributions are non-normal it may be inappropriate to apply typical assumptions made about standard deviations, and the term should be regarded not as a probabilistic function but as a simple indicator of error. Once the outputs are summarized from all realizations, mean and standard deviation statistics are attached to the attribute tables of the original data sets. This allows for the direct link of the original GIS output and uncertainty information.

## Application to Erosion Hazards Study

This research focuses on 10 sites from the Erosion Hazards study spanning three counties: Dare, North Carolina; Glynn, Georgia; and Sussex, Delaware. These 10 sites consist of 216 structures. The PES application was used to estimate uncertainty in nominal and interpolated BFEs calculated for each structure over the 60-year period at 10-year intervals. Table 2 lists the estimated positional accuracies of data layers used as input to the PES model.

Table 2: Erosion Hazards Study estimated input errors.

| Coverage | $1\sigma$ Error |
|---|---|
| Structures | 15ft |
| Erosion Hazard Area (EHA) | 5ft |
| Flood Zones | 5ft |

Metadata produced by 3001, Inc, states that each structure is within 15ft feet of the structure centroid, without suggesting a particular level of confidence. The metadata also recognizes that because the structure is in reality, 3-dimensional, and in the digital domain, 2-dimensional, there is some abstraction needed to represent the structure as a point. Due to the difficulties of estimating the centroid, and based on the authors personal experience with the structure data, it is more plausible to assume that the structure falls within 15ft of the centroid at the 1$\sigma$ level (i.e. 68% of the time). The flood zone and EHA layers were stream-mode digitized from various scale documents. In the case of the *PERF* and flood zones, it is difficult to assess the accuracy of a data layer that represents features not actually existing on the ground. Rather, a reasonable estimate of 5ft was used for these layers based on comparisons of the CERF to higher accuracy imagery.

Before carrying out the simulations it was necessary to remove extraneous detail in the arcs of the EHA and ZONE coverages resulting from stream mode digitizing. By generalizing and/or cleaning the data with three times the error tolerance, it is highly unlikely that the algorithm will produce overlaps or rips in realized data. The EHA and ZONE coverages were cleaned and generalized at 15ft using the default ARC/INFO *point-remove* function based on the Douglas-Peucker algorithm.

The PES application was used to simulate the Erosion Hazards Study 100 times. This produced 100 equally possible GIS outputs in the form of nominal and interpolated BFEs for years 0, 10, 20, 30, 40, 50, and 60. For each of the attributes, the mean and standard deviation were calculated as key measures of the resulting attribute uncertainty.

CHAPTER 5
RESULTS AND DISCUSSION

## Sample Results

As an introduction to the simulation output, the results from structure #1 are first presented. Figure 15 shows the source data layers around structure #1 and the projected structures created by the initial run of the analysis. Figure 16 shows the initial output and the 100 realizations of flood zones, EHA polygons, and projected structure locations. The initial GIS output and calculated uncertainty measures are presented in Table 3. The legend associated with Figure 16 is adopted for the remainder of this thesis.

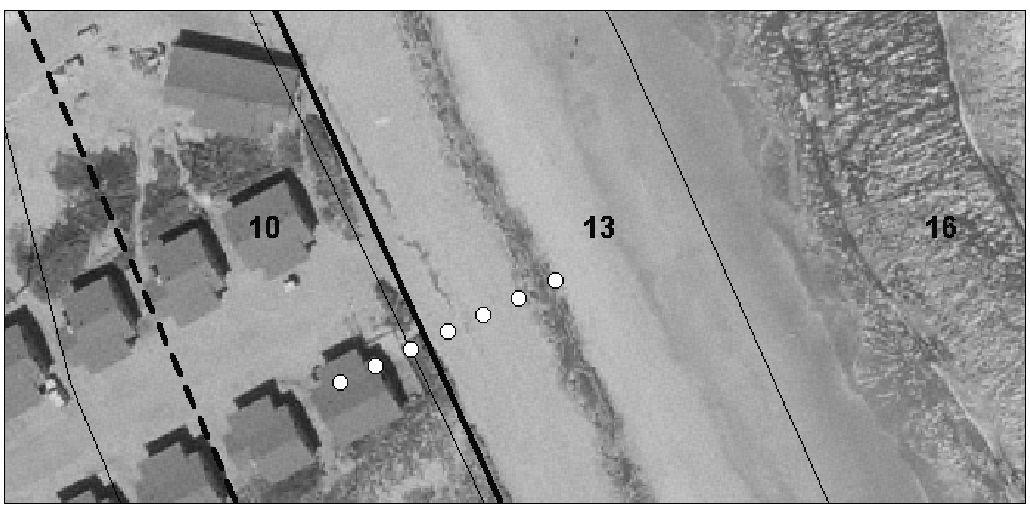


Figure 15: Structure #1 initial GIS output.

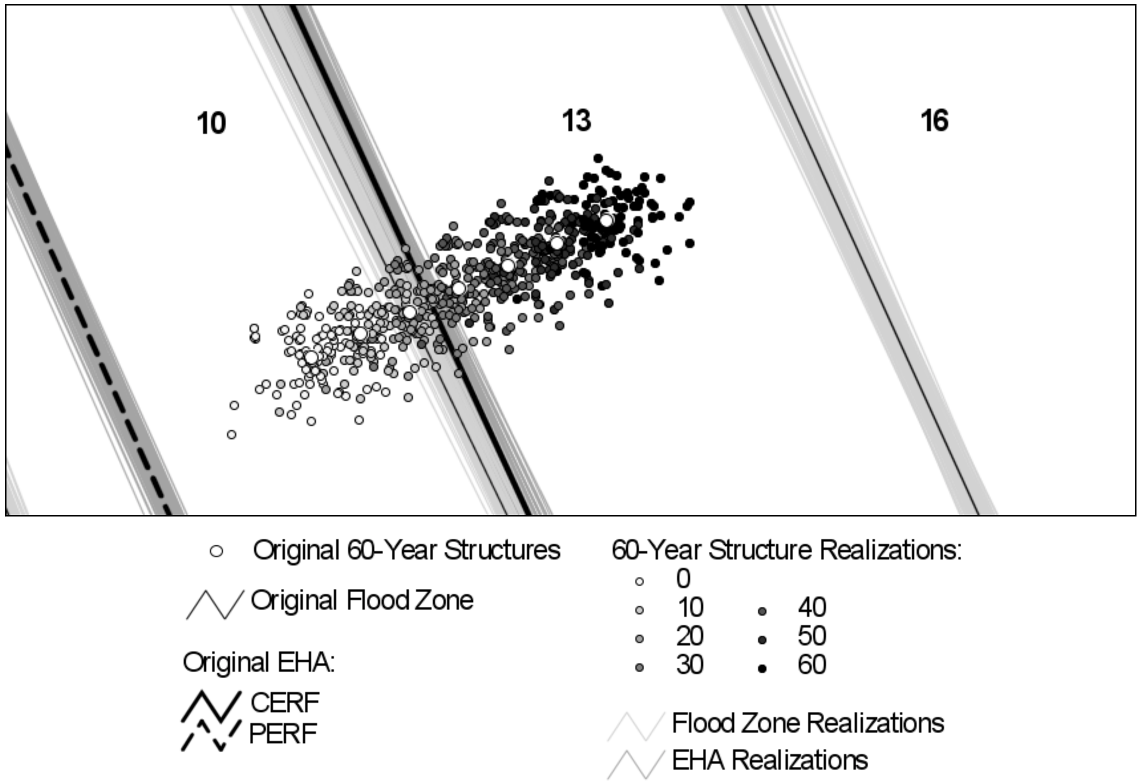



Figure 16: Structure #1 initial GIS output and 100 data realizations.

Table 3: Structure #1 attribute values and estimated uncertainty.

| | Year | Value | Mean | Standard Deviation |
|---|---|---|---|---|
| Nominal BFE | 0 | 10 | 10 | 0 |
| | 10 | 10 | 10.18 | 0.72 |
| | 20 | 13 | 11.56 | 1.51 |
| | 30 | 13 | 12.88 | 0.59 |
| | 40 | 13 | 13 | 0 |
| | 50 | 13 | 13 | 0 |
| | 60 | 13 | 13 | 0 |
| Interpolated BFE | 0 | 11.727 | 11.74 | 0.16 |
| | 10 | 12.005 | 12.02 | 0.16 |
| | 20 | 12.282 | 12.3 | 0.17 |
| | 30 | 12.56 | 12.57 | 0.17 |
| | 40 | 12.838 | 12.85 | 0.17 |
| | 50 | 13.116 | 13.13 | 0.18 |
| | 60 | 13.394 | 13.4 | 0.18 |

Structure #1 progresses from a nominal BFE of 10ft at years 0 and 10 to 13ft for years 20 through 60. The largest nominal BFE error of 1.51ft occurs at year 20. The remainder of this thesis will use the terms *error* and *uncertainty* interchangeably to refer to the standard deviation calculated by the PES model. The initial year 20 structure falls very close to the border of the two flood zone polygons and is therefore subject to point-in-polygon effects when data layers are perturbed. Figure 17 shows that 48 of the Year-20 structure realizations return a 10ft BFE, while the remaining 52 structures return 13ft. This is also presented by the histogram in Figure 18.

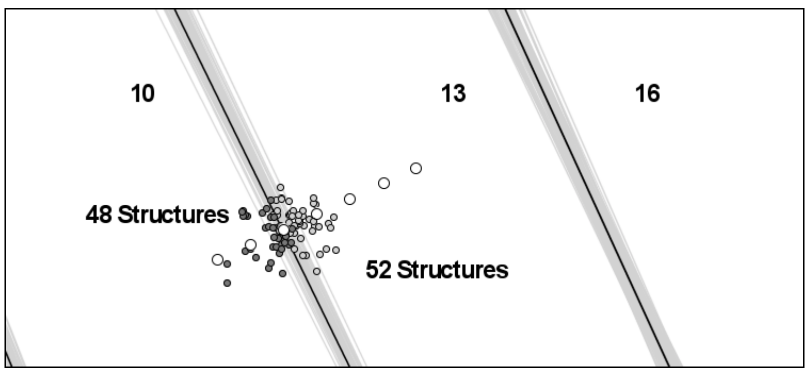


Figure 17: Structure #1 year 20 point-in-polygon errors.

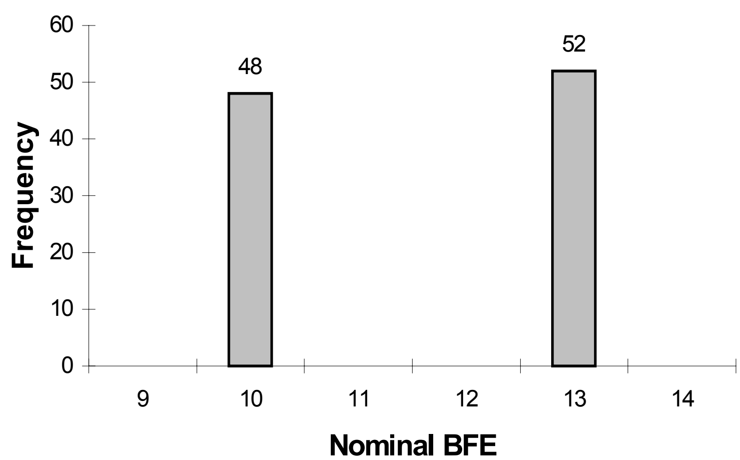


Figure 18: Structure #1 distribution of year 20 realized nominal BFEs.

As discussed in the previous chapter, application of the standard deviation function to non-normally distributed data may be misleading. In the case of nominal BFE, the resulting errors are not normally distributed and the probabilistic statements usually associated with the standard deviation statistics cannot be made. For example, it cannot be properly stated that 68% of the time, the structure returned a BFE within plus or minus 1.51ft of the mean BFE of 11.56ft. In addition, standard multipliers cannot be applied to attain error estimates at the 95% or 99% confidence levels. In reality, the structure will only return one of two possible values: 10 or 13ft. Furthermore, it may also underestimate the amount of error possible in nominal BFEs. Even though the calculated error is 1.5ft, the possible values actual span 3ft, and are almost equally likely to return either result. However, because the resulting distributions of possible field values may not be known, the statistic is useful as a generic indicator of error levels. Care must be taken in the interpretation of results. The errors of 0.72 and 0.59, produced at years 10 and 30 respectively, are also the result of point-in-polygon effects, although less pronounced. All other years return zero error in nominal BFEs.

The interpolated representations of BFE returned smaller errors. Point-in-polygon effects do not occur. Rather, errors in interpolated BFEs are the result of uncertainty in structure positions and variations in the TIN surface derived from errors in flood zone arcs. For visualization purposes, the 100 TINs were converted to 1ft contours as shown in Figure 19. The variability in contour locations closely resembles the uncertainty in parent flood zone boundaries. Any magnification of the uncertainty in these contours is attributable to uncertainties in the derivation of the TIN model. As shown in Table 3 errors in the interpolated BFEs increase slightly with each 10-year interval. Figure 20 shows a histogram of the realized interpolated BFE values for structure #1 at year 60 (corresponding to Figure 19). The returned interpolated BFEs approach a normal distribution, indicating that the standard deviation statistic does provide a useful estimate of error. Other interpolated BFE error distributions varied from normal to skewed to bimodal.

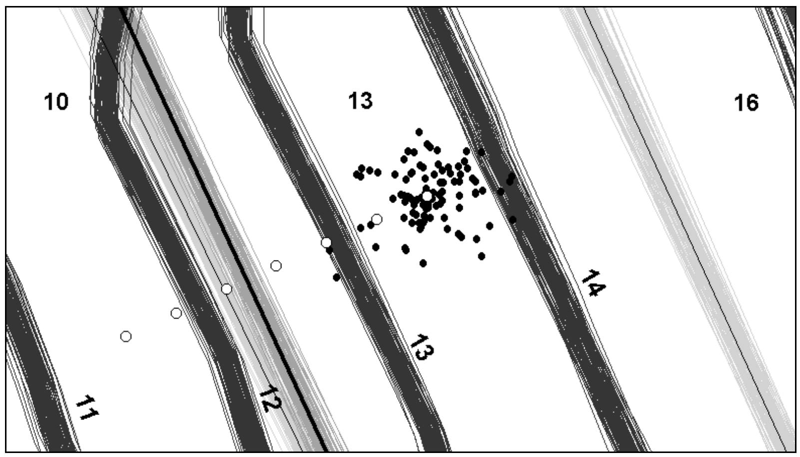


Figure 19: Structure #1 year 60 realizations with interpolated 1ft contours.

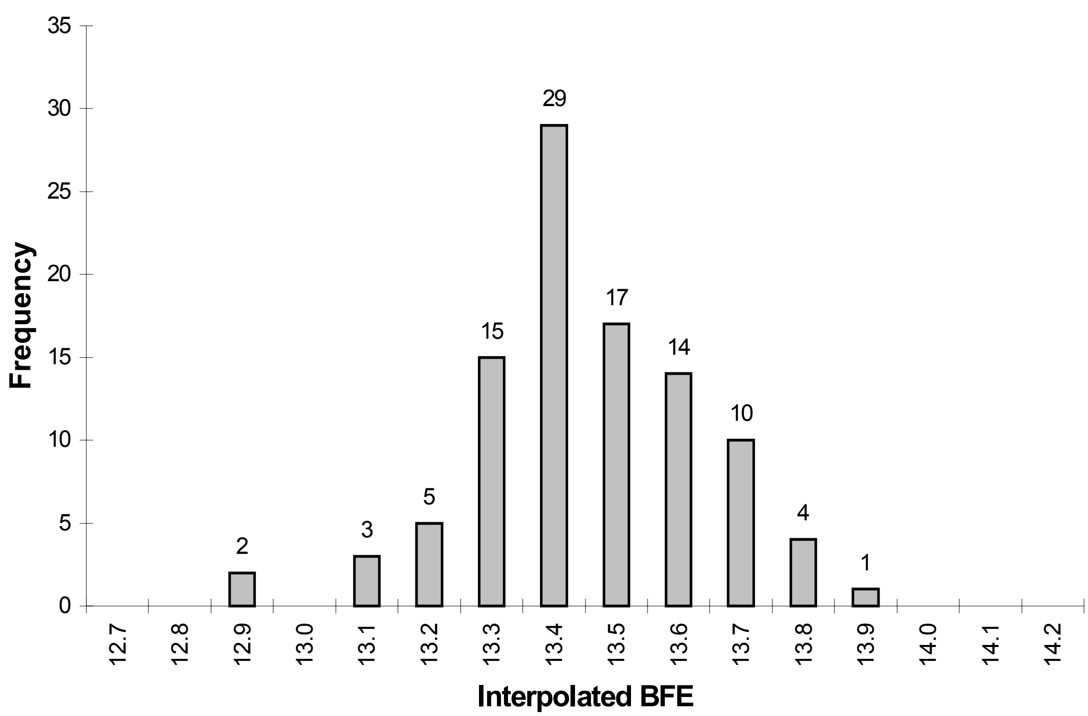
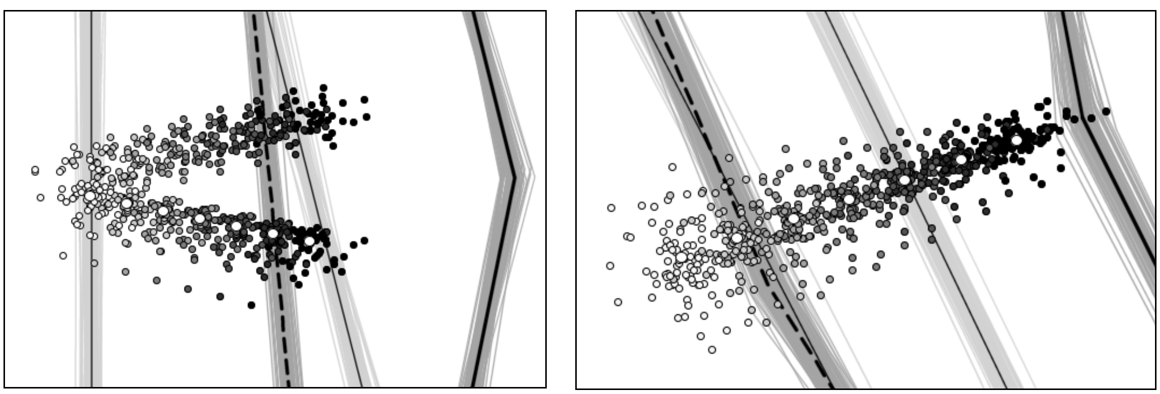


Figure 20: Structure #1 distribution of year 60 realized interpolated BFEs.

Uncertainty in the nominal and interpolated BFEs is largely affected by the dispersion of projected structures realizations. The geometry of the EHA and the structure's proximity to the CERF determine projected structure dispersion or *spread*. Figure 21 shows examples of increasing (a) and decreasing (b) projected structure spreads. Large spreads increase the likelihood of error propagation into BFE values. In Figure 21a, the calculation of the NEAR point (refer to Chapter 3) on the CERF returns equally possible locations on different line segments, causing realized transects to point in two different directions. In Figure 21b, realized structures share a common near point (the vertex of the CERF line segments) causing projected structures to converge with each 10-year interval.



a) Structure #185, increasing projected structure spread.

b) Structure #53, decreasing projected structure spread..

Figure 21: Increasing and decreasing spread of projected structure realizations.

In order to compare values, a measure of circular error (similar to $RMSE_{CIRCULAR}$) was calculated for each 10-year interval based on the standard deviations of the x and y coordinates.

$$\sigma_C = \sqrt{{\sigma_X}^2 + {\sigma_Y}^2}$$

When $\sigma_X = \sigma_Y$, $\sigma_C$ provides the radius of a circle (centered on mean coordinates) within which approximately 63% of realized points fall (Mikhail and Ackerman, 1976). When $\sigma_X$ and $\sigma_Y$ are not equal, the resulting radial probability is variable. Furthermore, when structure realizations do not follow the circular normal distribution, the circular error looses its probabilistic definition and is only retained as a basis for comparison. Circular errors were calculated for projected structures for each 10-year interval. At year 0, the circular error is expected to approximate the input uncertainty of 15ft. The PES algorithm produces a circular normal distribution such that 68% of realized structures should fall within the estimated input error (15ft for structures). Since $\sigma_C$ is at the 63% level, calculated values at year 0 should be slightly less than 15ft. In the case of structure #1, the circular error slightly increases with each 10-year interval (Table 4). Table 4 also shows calculated circular errors corresponding to the structure #s 185 (increasing spread) and 53 (decreasing spread) from Figure 21.

Table 4: Sample circular errors of projected structures.

| Year | 0 | 10 | 20 | 30 | 40 | 50 | 60 |
|---:|---|---|---|---|---|---|---|
| Structure #1 | 14.8 | 15.0 | 15.2 | 15.4 | 15.7 | 16.0 | 16.3 |
| Structure #185 | 15.1 | 18.6 | 22.2 | 26.1 | 30.3 | 34.6 | 39.1 |
| Structure #53 | 14.9 | 13.9 | 12.9 | 12.2 | 11.6 | 11.3 | 11.2 |

## Study-Wide Results

Initial output from the PES model is the mean and standard deviation for each item of interest linked to the structure record. However, a general idea of the uncertainty present in these attributes, across all structures, is desirable. This section will report on the study-wide uncertainty levels present in nominal and interpolated BFEs.

Histograms were created for each 10-year interval over the 60-year period based on the 216 observations of nominal BFE error. Figure 22 shows the results for year-0. About 82% of structures returned zero error in the BFE calculation. About 93.5% of structure BFE errors fell between 0 and 0.5ft and all BFE errors were less than 1.5ft.

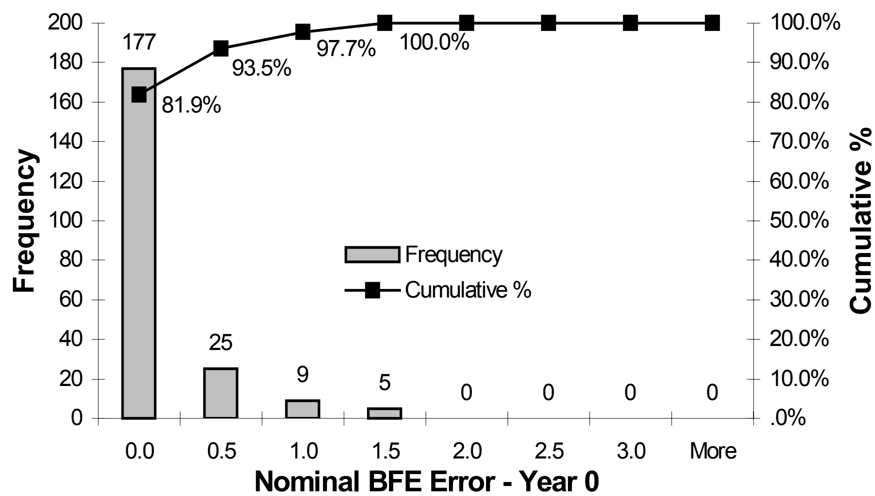



Figure 22: Study-wide histogram of nominal BFE error at year 0.

Histograms were combined to show the cumulative percentages of structures falling within nominal BFE error ranges for each 10-year interval (Figure 23).

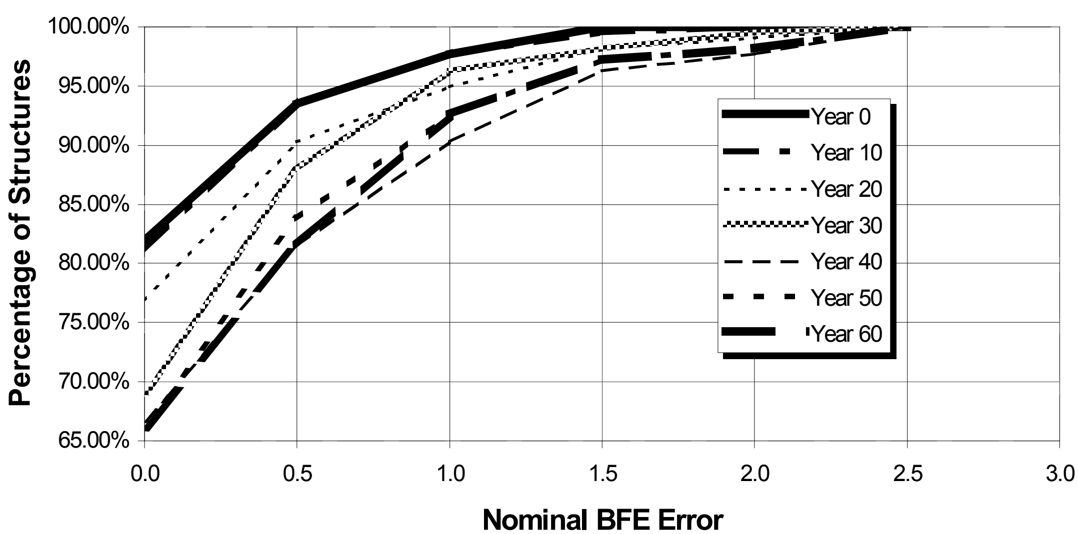



Figure 23: Study-wide nominal BFE uncertainty.

In general, nominal BFE error increases with each 10-year interval. This is expected as the likelihood of point-in-polygon effects increases with structure spread. 80% of nominal BFE errors are within 0.5ft, 90% within 1ft, and all within 2.5ft

Similarly histograms were created for interpolated BFE errors and were combined into a plot of cumulative percentages (Figure 24). The figure leaves off the lower 80% for clarity. From 0.25 the plots for all years go to zero error at 0%. In other words, no structures were free from interpolated BFE errors. All projected structures returned an interpolated BFE of less than 1.25ft and about 96% of structures had an error of 0.5ft or less. As observed with nominal BFEs, the interpolated BFE errors also increase with each 10-year interval.

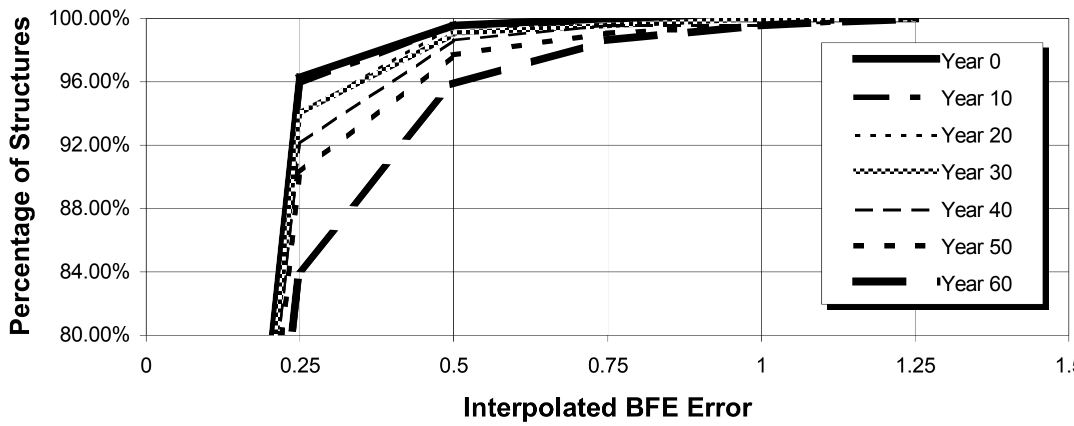



Figure 24: Study-wide interpolated BFE uncertainty.

Figure 25 is a histogram of calculated circular errors for year-60. 70 of the 216 year-60 structures did not exhibit any more than 15ft circular errors, attributable to the estimated uncertainty in year 0 structures. 108 structures displayed minor structure spreads between 15 and 20ft. Approximately 10% of the 60-year structures spread significantly at 30ft or more. In large structure spread areas, it is highly likely that uncertainty will propagate into nominal and interpolated BFE values.

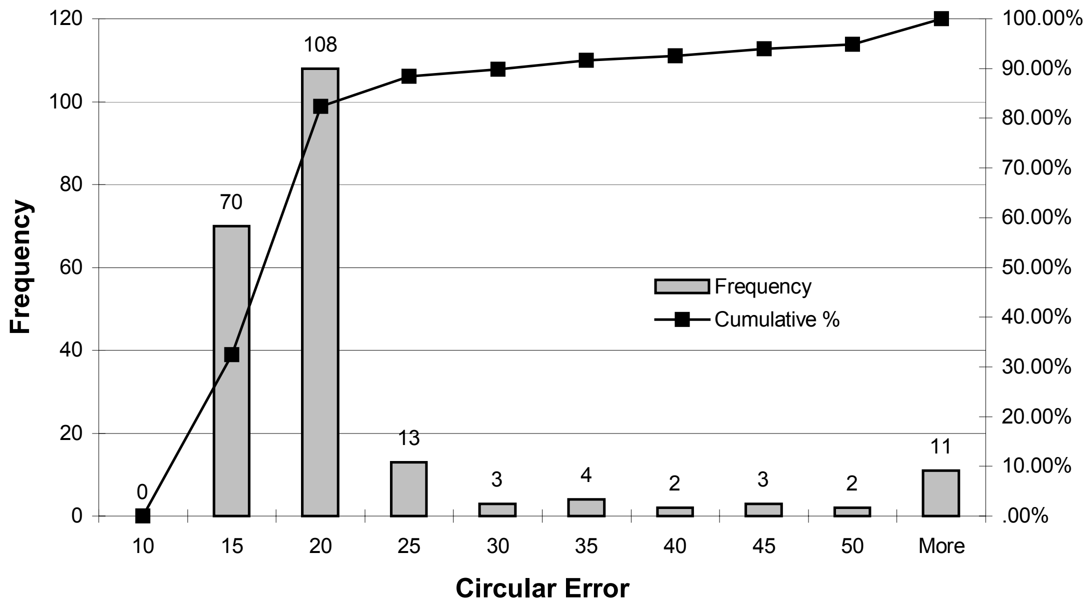



Figure 25: Study-wide structure spread - year 60

## High Uncertainty Cases

Nominal BFEs will contain the most uncertainty when there is high probability that perturbations in the structure and zone polygons will return different elevations during polygon overlay. This is increasingly likely in large structure spread areas. The highest nominal BFE error in the database was 2.5ft occurring at structure #132 in year-40 where 60% percent of the realizations return a BFE of 15ft while the remaining 40% return a BFE of 20ft (Figure 26). There are two primary reasons for such a high uncertainty at this point. First, due to the initial projected structures proximity to the zone border, the realized structures are evenly dispersed into neighboring zones. The second factor is the large step across the zone boundary from a BFE of 15 to 20ft.

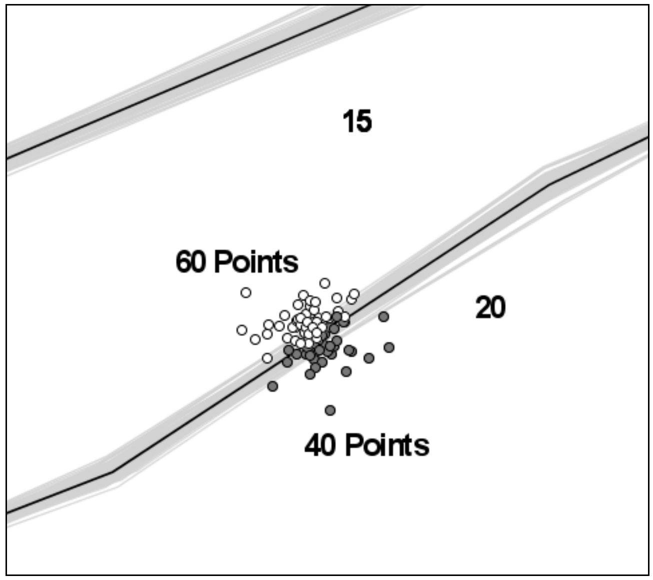


Figure 26: Structure #132,year 20 realizations.

High uncertainty in interpolated BFEs can be attributed to large structure spreads making it more likely for projected structures to fall in different elevation ranges. The largest interpolated BFE error of 1.25ft occurred at structure #26 in year 60 (Figure 27). In the figure, nominal BFE values are shown in black while coded arc elevations for the interpolated TIN are shown in white. The projected structure realizations are sent in two different directions. At year 60, the majority of structures fall in the center of a BFE zone of 12ft. Interpolated BFE values for this lower group are clustered around 12.0ft. The upper group of year-60 structures returned interpolated BFEs varying only slightly from 15.9ft. The histogram in Figure 28 shows the distribution of realized interpolated BFEs for year 60. Although each cluster of 60-year structures revealed relatively precise measurements of interpolated BFE independently, taken together, the interpolated BFE error is 1.2ft. The 10% of circular errors over 30ft (as shown in Figure 25) were caused by similar cases of structure spread.

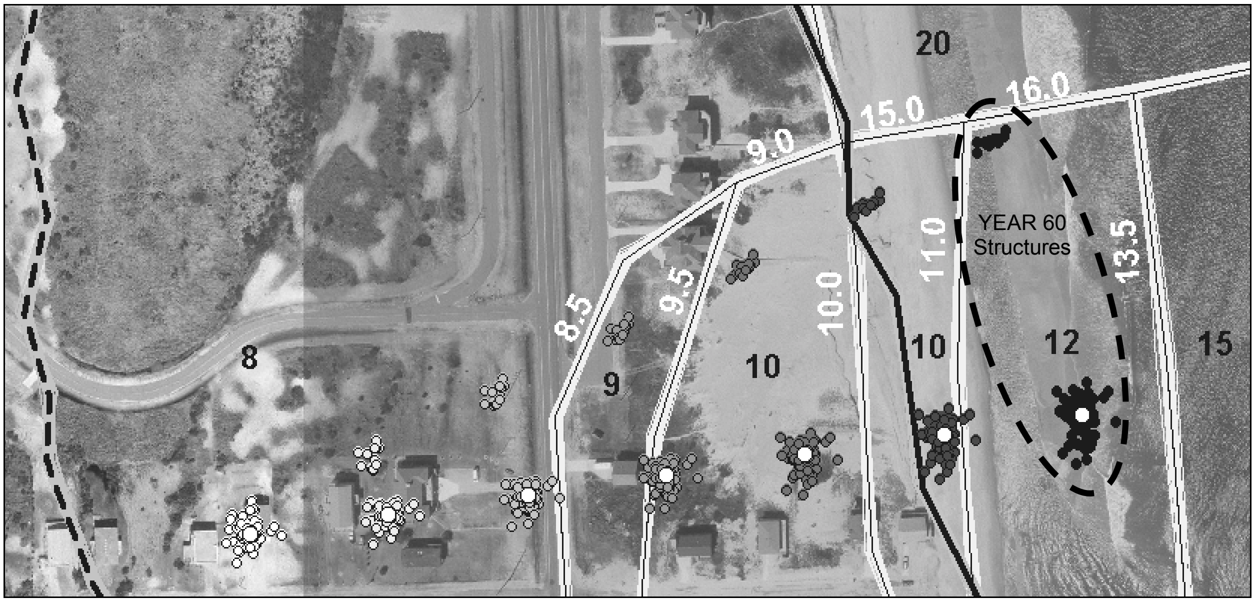



Figure 27: Structure #26 realizations exhibiting large year-60 interpolated BFE error.

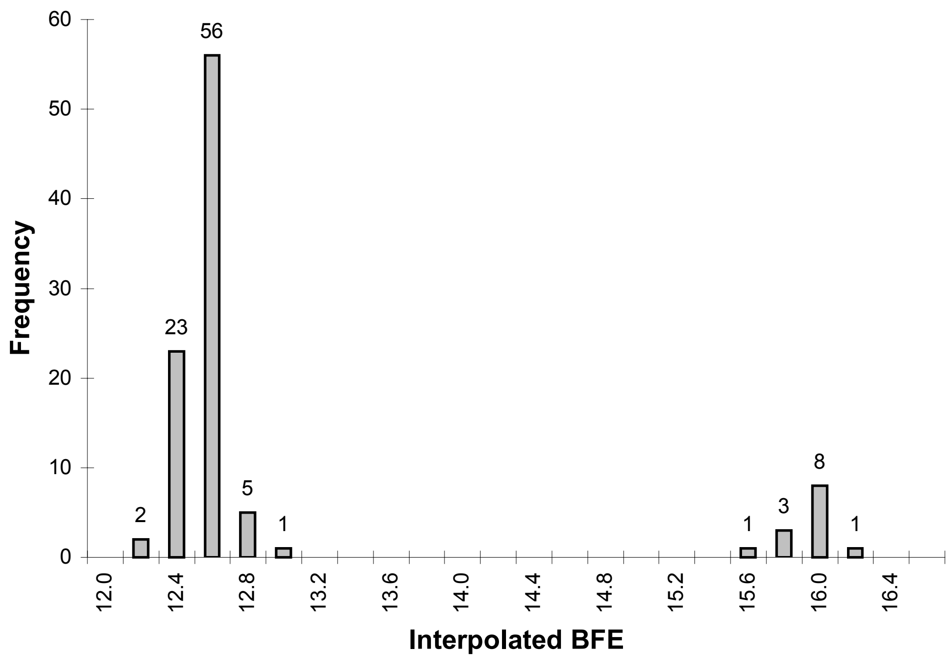



Figure 28: Structure #26 distribution of year 60 realized interpolated BFEs.

As previously observed with nominal BFEs, the error distribution as shown in Figure 28 is not normal and the use of standard deviation as a summary statistic disguises the actual nature of error at this point. Although the calculated standard deviation is 1.25ft, it is actually likely that interpolated BFEs as far apart as 4ft will be returned.

## Usefulness

The previous section has shown that simulation methods can be applied to complex GIS analyses to derive estimates of uncertainty in reported values. In the Erosion Hazards Analysis, 90% of nominal BFEs returned errors of 1ft or less and 90% of interpolated BFEs had less than 0.5ft of error. Certain structures in the analysis revealed large and unanticipated errors. Nominal and interpolated errors were as high as 2.4 and 1.25ft respectively. However, the realized error distribution must also be considered when evaluating the usefulness of estimated uncertainty measures. When output errors are not normally distributed, the mean and standard deviation statistics fail to adequately describe the nature of uncertainty in derived attributes.

A distinction must be drawn between absolute uncertainty and uncertainty existing within the GIS model. Output from the PES application reports only variability arising from estimated uncertainties in the positions of input data. In reality, there are numerous other sources of error also propagating into final values. Furthermore, the models themselves may be misrepresenting the real world. For example, although the majority of interpolated BFEs were found to contain less than 0.25ft of error, these values are just smoothed representations of, and derived directly from, nominal BFEs. It is incorrect to assume that interpolated values are *more accurate* than nominal values.

The largest encumbrance of the proposed error model is the required computer processing time. The Erosion Hazards error analysis was performed on 600MHZ Pentium III computers with 128mb's of memory running Windows NT and ARC/INFO version 7.2.1. The original analysis, carried out by *fema.aml* took from 5 to 10 minutes depending on the site. Running the PES model with 100 iterations took from 8 to 18 hours per site. As expected, the time required for the error analysis is expanded *N* times plus the time for creating data realizations and summarizing output. Although the time added for perturbing data sets was minor, the ARC/INFO STATISTICS function, which operated on INFO tables, was considerably slower than desired. The storage required for the PES realizations and summary statistics is approximately *4N* times the original disk space. This will vary by data type and complexity. Although time and storage restrictions are considerable, these constraints are becoming less problematic with advances in computer hardware.

Simulation results can improve the usefulness of data quality statements now required in metadata standards. Where no other basis for comparison exists, as in the Erosion Hazards Study, the data producer usually estimates the uncertainty in derived values. Although estimation may be adequate in many analyses, complex layer interactions may produce large and unexpected errors. In these cases, a realistic representation of output error can only be attained through simulations.

Uncertainty data can be used to appropriately represent data variability by limiting the *reporting precision* of output values. For example, in the Erosion Hazards Analysis, the TINSPOT function returned interpolated BFE values to one thousandths of a foot. After simulating the data and quantifying the variability in elevations it might be more appropriate to display the data to only the hundredths or tenths of a foot. Subsequent users of the data then have a rough idea of the data's inherent accuracy simply by the number of significant figures.

As proposed by Veregin (1994), simulation modeling can be used as the basis for mathematical error propagation models. By developing descriptive models based on simulation output, error propagation can be assessed in similar systems, without performing time-consuming simulations. The application of the PES model to the Erosion Hazards Study has lead to several observations that support this technique. For example, output uncertainly was largely based on projected structure spread values. Structure spreads could be predicted by variables such as distance to the CERF and the concavity or convexity of the CERF line. Another model parameter could be the distance to zone boundaries where larger errors are more likely. In addition, these parameters could be introduced as dummy variables.

The error model can also be used during the design phases of a GIS project. By applying such a model in a pilot study, expected levels of output uncertainty can be estimated and data collection and/or processing methods can be revised to more directly meet project requirements. In addition, the error analysis can be used in a suitability framework. By using a trial-and-error approach, input errors levels and algorithms can be modified until required accuracy levels are reached. Heuvelink, Burrough, and Stein (1989, p. 13) recognize that this information can be "potentially of great value for resource managers who need to get reliable information for realistic budgets". This research specifically looks into one such application proposed by Openshaw (1989): the isolation of output uncertainty derived from individual data layers.

Openshaw (1989) states that by considering error in a single source layer and holding all other layers error-free, the contribution of that layer to output uncertainty levels can be identified. The goal of such a study is to determine if the improvement of a single data layer could significantly reduce uncertainty in GIS outputs. Such information could prove invaluable for determining appropriate data collection techniques. If levels of output uncertainty are unacceptable, this approach could identify data layers whose improvement would most benefit study results. Conversely, if errors in outputs are negligible, the project may benefit from the degradation of input data quality, by substituting more cost effective data collection methods.

## Isolating Error Contributors

To isolate the degree of output uncertainty contributed by the presence of positional error in each source coverage, the 100 simulations were repeated three times, each time only considering positional error in one of the source coverages. The original error analysis, considering errors in all 3 layers will be referred to as the *complete* analysis. Results from the *complete* model were compared with the individual layer analyses, termed *onlyeha*, *onlystru*, and *onlyzone*.

When only considering positional error in the structures layer, all uncertainty in nominal and interpolated BFEs is derived from the structure spread. BFE errors will magnify with the structure's proximity to zone boundaries and in areas of high structure spread.

In the *onlyeha* analysis, uncertainty in nominal BFEs will depend on shifts in the EHA polygon and the resulting projected structure spreads. Even through the structures layer is not perturbed, shifts in the CERF line of the EHA polygon will cause the NEAR algorithm to produce variable projected structure positions (Figure 29). Because flood zones are held error-free, uncertainty in the interpolated BFEs will result only from this slight structure spread.

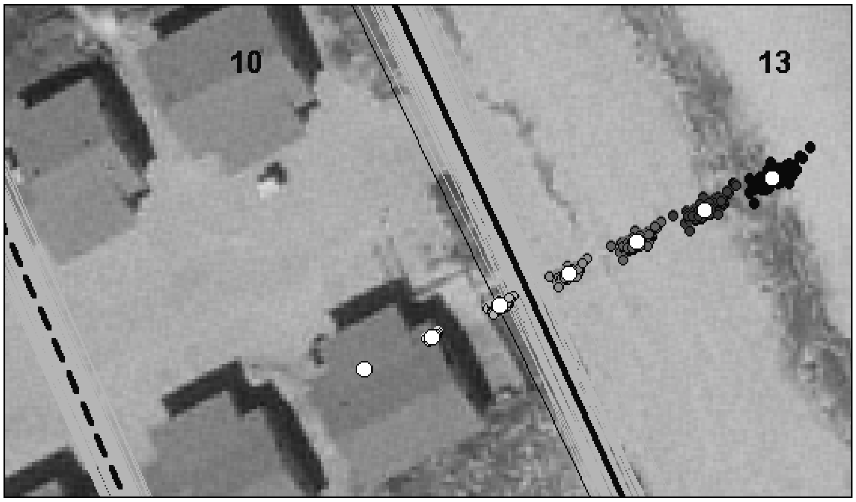



Figure 29: Structure #1 *onlyeha* realizations.

In the *onlyzone* analysis, structures and the EHA are error free, producing no variability in the positions of projected structures. Any uncertainty in nominal and interpolated BFEs are caused by the shifting of flood zone boundaries.

Study-wide average BFE errors were calculated for the 4 runs of the error analysis (*complete*, *onlyzone*, *onlyeha*, and *onlystru*). A plot of average error against year (Figure 30) reveals error in nominal BFEs is almost completely explained by positional errors in the structures layer. The inclusion of zone and EHA errors contribute minimal amounts of uncertainty to final BFEs.

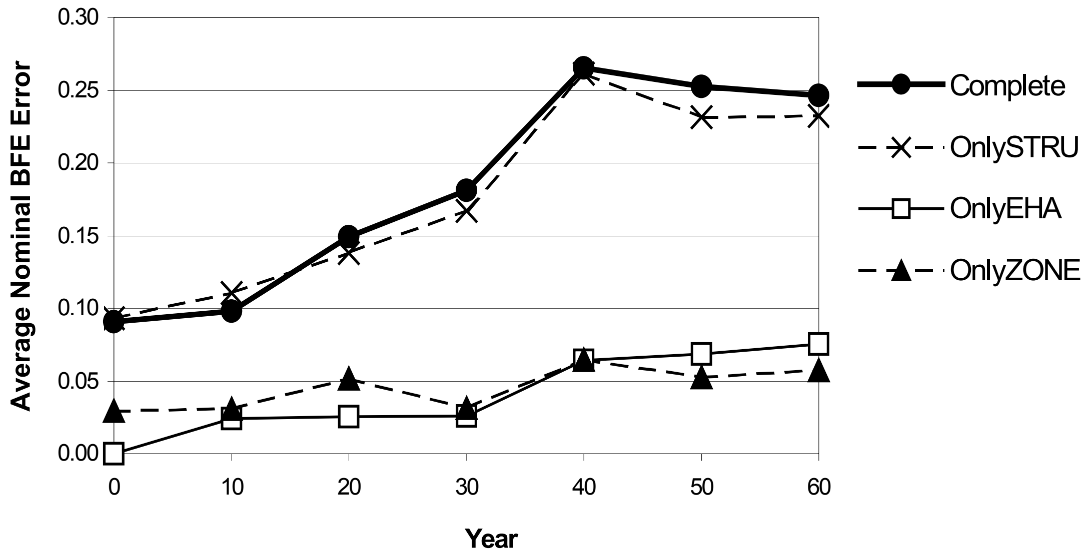



Figure 30: Average nominal BFE uncertainty for isolated runs.

Similarly, study-wide averages of interpolated BFE errors were calculated for each run (Figure 31). Interpolated BFE uncertainty is also explained predominantly by uncertainty in the structures layer. The effects from zone and EHA layers are smaller but significant. When only error in structures is considered, variability in interpolated BFEs is attributable only to structure spread. Because zones are held fixed, the 100 realized TIN surfaces will be exactly the same throughout the analysis. When only the EHA layer is considered, the structure spread is the only contributor to interpolated BFEs. When only zones are considered, the TIN surfaces are themselves shifted while structures remain fixed.

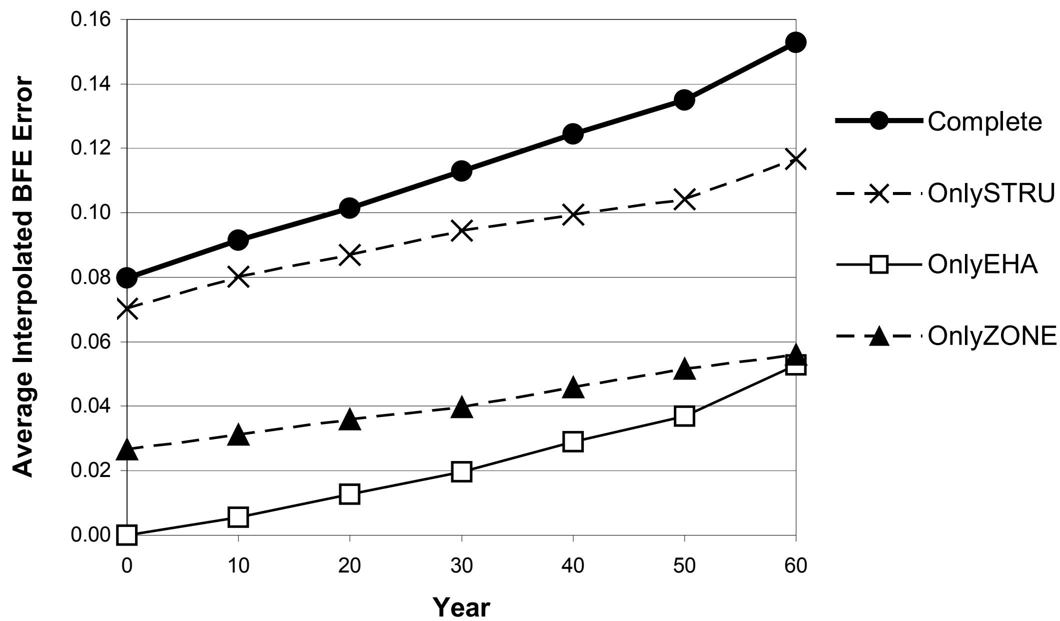



Figure 31: Average interpolated BFE uncertainty for isolated runs.

To compare structure spread values, study-wide averages of circular error were calculated for each run (Figure 32). In the *complete* case, circular errors progress from 15ft at year 0 (as expected) to approximately 21ft at year 60. When only structure errors are considered, circular errors follow the complete case very closely, differing only in the fact that the *complete* run contains and additional level of uncertainty due to variability introduced by perturbation of the EHA. Considering only errors in the EHA, structure-spread progress from no error at year 0 to about 8ft at year 60. The *onlyzone* run does not alter positions of projected structures.

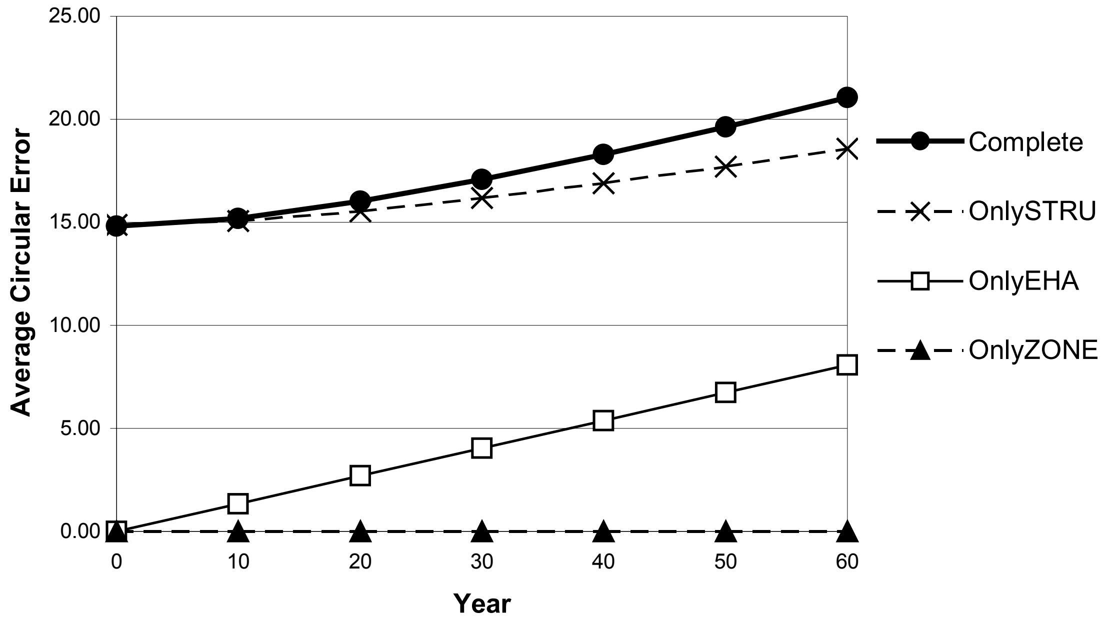



Figure 32: Average structure spreads for isolated runs.

Assuming only the presence of uncorrelated random error in derived uncertainty estimates, the square root of the sum of squared errors from individual layers can approximate how uncertainty is propagated through interaction effects during the *complete* analysis.

$$\sigma_{EXPECTED} \approx \sqrt{\sigma_{ONLYSTRU}{}^2 + \sigma_{ONLYEHA}{}^2 + \sigma_{ONLYZONE}{}^2}$$

Although the actual degree of correlation between layers is unknown, comparing the *expected* combination of errors against the *observed* or *complete* value allows the identification of cases where errors accumulate in an unexpected manor. To facilitate this comparison, the parameter Δ was calculated as follows.

$$\Delta = \sigma_{COMPLETE} - \sigma_{EXPECTED}$$

Using the results from structure #1 as shown in Table 5, the complete run returned an error of 0.18ft for the interpolated BFE at year 60. Applying the formulas above,

$\sigma_{EXPECTED}$ = 0.16, and $\Delta$ = 0.02. Because Δ is positive, the actual interaction of data layers returns error levels slightly larger than expected.

Table 5: Structure #1, year 60 interpolated BFE errors for complete and isolated runs.

| | |
|---|---|
| $\sigma_{COMPLETE}$ | 0.18 |
| $\sigma_{ONLYEHA}$ | 0.05 |
| $\sigma_{ONLYSTRU}$ | 0.15 |
| $\sigma_{ONLYZONE}$ | 0.03 |
| $\sigma_{EXPECTED}$ | 0.16 |
| $\Delta$ | 0.02 |

Δ values were calculated for all nominal and interpolated BFEs. A histogram of Δ values for the interpolated BFE at year 60 is presented in Figure 33. Approximately 90% of structure errors accumulate as expected, as their Δ values are reasonably close to zero (+/- 0.05). The normally distributed nature of values indicate it is equally likely for errors to accumulate or compensate during the actual interaction of data layers. Years 0 - 50 produced similar distribution of Δ values for nominal and interpolated BFEs.

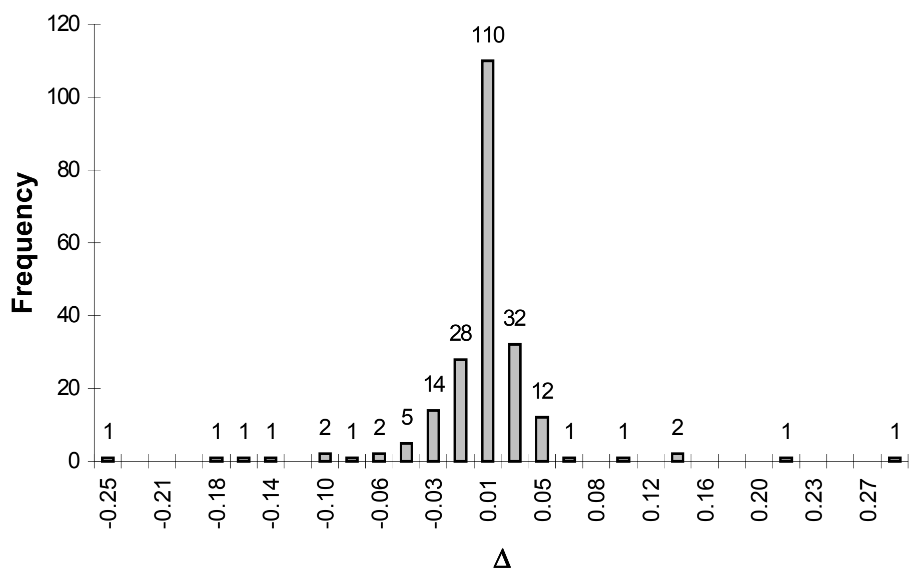


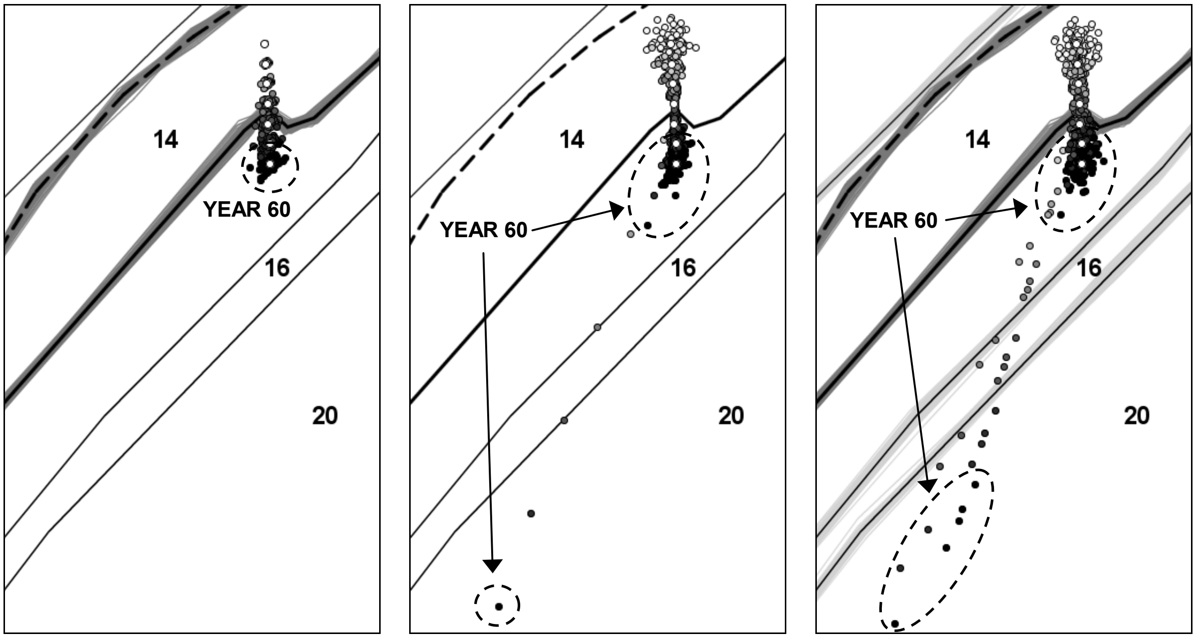

Figure 33: Histogram of Δ values for interpolated BFE at year 60.

Positive Δ values reveal cases where expected errors are less than those observed in the *complete* run. In these cases, interaction effects magnify output uncertainty. Conversely, negative Δ values occur when the expected error is greater than observed levels, revealing situations where layer interactions diminish output uncertainty. In these cases, errors from different layers tend to compensate.

Using the metrics introduced above, structure #79 exhibits the largest case of error expansion. Based on the results given in Table 6 the expected error is 0.44ft. However, layer interactions of the *complete* model produced the observed value of 0.65ft. The Δ value is 0.21. Figure 34 shows projected structure positions created for the isolated runs of structure #79. The *onlyzone* analysis does not produce variability in structure positions. The *onlyeha* analysis (Figure 34a) returns projected structures falling within the 14ft BFE zone and revealing little structure spread. The calculated error is 0.05ft. When structure errors are considered in the *onlystru* model (Figure 34b), the data reveals a large erosion hazard progression, projecting a single structure into neighboring zones and magnifying the resulting error. When all layers are randomized in the *complete* model (Figure 34c), several projected structures exhibit the extended progression, returning a final uncertainty of 0.65ft, considerably larger than the expected value of 0.44ft.

Table 6: Structure #79, year 60 interpolated BFE errors for complete and isolated runs.

| | |
|---|---|
| $\sigma_{COMPLETE}$ | 0.65 |
| $\sigma_{ONLYEHA}$ | 0.16 |
| $\sigma_{ONLYSTRU}$ | 0.41 |
| $\sigma_{ONLYZONE}$ | 0.05 |
| $\sigma_{EXPECTED}$ | 0.44 |
| Δ | 0.21 |



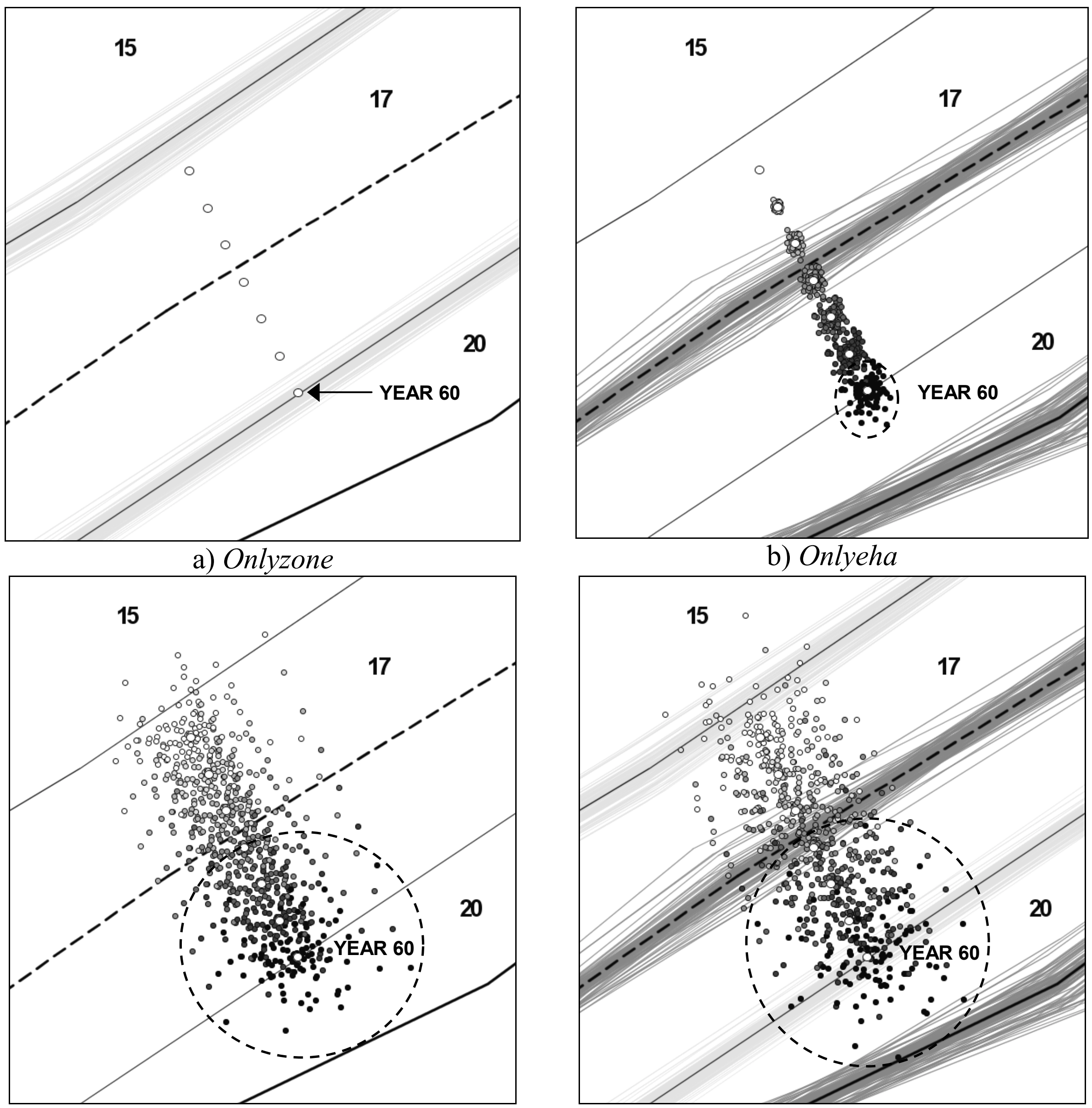

a) *Onlyeha* b) *Onlystru* c) *Complete*

Figure 34: Structure #79 comparison of complete and isolated realizations.

Structure #102 exhibits the opposite effect. The interaction of data layers in the *complete* model tends to compensate expected errors from individual coverages. The results for interpolated BFE errors in year 60 are shown in Table 7 and Figure 35. The *onlyzone* run (Figure 35a) produces a large uncertainty of 1.07ft. The derived TINs produce large variability since the 60-year structure is very near the zone boundary. The *onlyeha* run (Figure 35b) returns very little error as the zones are held fixed and the structure spread is not considerable. In the *onlystru* run (Figure 35c), a large structure spread produces 0.17ft of error. The expected accumulation of errors produces a value of 1.08, primarily due to uncertainty from the *onlyzone* model. However, the observed model only revealed an error of 0.89, resulting in a Δ of –0.19. In the *complete* case (Figure 35d), the combination of structure spread and zone shifts tends to cancel the amount of expected uncertainty.

Table 7:  Structure #102, year 60 interpolated BFE errors for complete and isolated runs.

| | |
|---|---|
| $\sigma_{COMPLETE}$ | 0.89 |
| $\sigma_{ONLYEHA}$ | 0.07 |
| $\sigma_{ONLYSTRU}$ | 0.17 |
| $\sigma_{ONLYZONE}$ | 1.07 |
| $\sigma_{EXPECTED}$ | 1.08 |
| $\Delta$ | -0.19 |




a) *Onlyzone*　b) *Onlyeha*

c) *Onlystru*　d) *Complete*

Figure 35:  Structure #102 complete and isolated outputs.

Despite these unusual cases of error expansion and compensation, approximately 90% of BFE errors accumulated as expected. Therefore, refining any of the input data layers will improve the precision of computed quantities for the majority of structures. However, the conclusion sought is which data layer's improvement would most benefit the study's results. Although sophisticated models could be developed to reveal actual parameters of expected error contributions, the simple comparison of average errors and the fact that errors do accumulate as expected, shows that the structures layer contributes the most to output uncertainty. Study results could best be improved by increasing the accuracy of structure locations, and to a lesser but significant degree by the zone and EHA layers. However, practical limitations of the object's representation must also be considered. Although structure locations could be collected to centimeter accuracy level if desired, there is still the unavoidable problem of representing a 2-dimensional building footprint as a single point. Arguably, the consequent reduction in output uncertainty could be meaningless.

# CHAPTER 6
# CONCLUSION

The Erosion Hazards Study was a comprehensive GIS analysis that predicted Base Flood Elevations of coastal structures in the event of a 100-year flood 60 years into the future. In order to estimate uncertainty in reported BFEs, the Positional Error Simulator (PES) application was written for ARC/INFO to implement Monte Carlo simulations. The PES application simulates error in data sets, carries out spatial analyses, and summarizes observed variability with basic statistics. The model assumes normally distributed independent positional errors in the points, nodes, and vertices that make up vector data.

Although many researchers have identified the ability for the error model to be applied to complex analyses, few tests have been presented. Applying the PES model to the Erosion Hazards Study has shown that the simulations can indeed be used in a substantial spatial analysis. Furthermore, simulations are perhaps the only practical method for obtaining realistic uncertainty estimates. Openshaw (1989, p. 265) recognizes "what many applications seem to need is not precise estimates of error but some confidence that the error and uncertainty levels are not so high as to render in doubt the validity of the results." The proposed PES software provides a practical method for estimating the variability of GIS outputs due to positional errors, allowing GIS users to evaluate the data's fitness for use.

Application of the PES error model to the Erosion Hazards Study allows the original research questions to be addressed.

1. *What levels and distributions of error exist in derived Base Flood Elevations?*

Summary statistics showed that 90% of nominal BFEs returned errors of 1ft or less and 95% of interpolated BFEs had less than 0.5ft of error. Nominal BFE errors were largely due to point-in-polygon occurrences during polygon overlay. Consequently, the resulting distributions were highly non-normal, usually returning only two possible elevations. Interpolated BFE errors arose during the conversion of polygon zones to contours needed for the derivation of TIN surfaces. Observed distributions varied from normal to skewed to bimodal. Several cases of high uncertainty were introduced and shown to arise in areas of high structure spread. Nominal and interpolated BFE errors were as high as 2.5 and 1.25ft respectively.

2. *What are the potential uses and limitations of the proposed error model and the resulting uncertainty data?*

The calculated uncertainty information has multiple uses. These can be the basis for reliable quality statements required in metadata documents. The mean and standard deviation measures can be reported at the record-level, attaching individual measures of data quality to geographic objects. These values can be summarized to achieve overall data quality statements.

The variability in reported values can be used to limit the reported precision of GIS attributes. GIS functions will always produce data stored beyond their inherent precision. Displaying an appropriate number of significant figures will allow users to quickly assess data quality, and curb inappropriate data usage.

Simulations not only provide quantitative estimates of uncertainty but also allow for the visualization of error. Arguably, visual output is possibly more valuable than quantitative summary statistics. Being able to see hypothetical data paths allows the user to identify how errors occur and impact the study. Furthermore, the visualization of error may lead to the identification of alternative, less error-prone, processing methods.

Simulation output can be used to develop formal models of error propagation (Veregin, 1994). By making general observations about the nature of error, parameters can be introduced to model situations in which errors arise. These models can then be applied to similar functions to develop error estimates without performing time consuming simulations.

The proposed simulation method and uncertainty data can be used in a suitability framework. By using a trial-and-error approach, input errors and processing algorithms can be varied until desired levels of data quality are reached. This usage of the error model is a valuable resource for project members who must determine appropriate and cost-effective methods of data collection and processing. A related study, the isolation of maximum error contributors, was investigated in depth and discussed below for research question #3.

Several limitations of the proposed error model and resulting uncertainty data have also been revealed. The assumption of independent random error cannot accurately be made for all data sets. Currently, the model requires the use of generalization techniques to prevent rips in linework. Several data collection and conversion processes will lead to correlated errors, and could better be modeled by accounting for systematic effects in realized data.

Another limitation of the proposed error model is the generic application of the standard deviation as a summary statistic. Care must be taken in the interpretation of output uncertainty estimates as the value of summary statistics are largely based on unpredictable realized error distributions. When errors in GIS-derived values are not normally distributed, probabilistic statements about the nature of error cannot be made. In the case of nominal BFEs, most uncertainty was derived from point-in-polygon effects returning two possible elevations. Clearly non-normal, these errors might be better represented by an error of confusion matrix. Many errors in interpolated BFEs approximated the normal distribution while others were highly skewed. In rare cases such as structure #26, realizations of interpolated BFE returned two clusters of normally distributed data. In this case, the uncertainty might better be represented by bimodal statistics. The error model would benefit from more useful summary statistics based on observed distributions.

The error model suffers from significant increases in processing time and data storage requirements. There is therefore a need to streamline algorithms and data storage methods. By handling summary statistics within ARC/INFO additional and avoidable processing times were encountered by the STATISTICS routine and the storage of tabular data in the INFO format. By coding summary routines in C, the model would speed up considerably. In addition, storing realization and summary tables in ASCII format would reduce disk space requirements.

The reported error levels are only due to estimated positional accuracies of source layers. Other factors such as attribute accuracy and model misspecification error must also be considered to fully express uncertainty in reported values. In the Erosion Hazards

Study, simulating attribute uncertainty would involve coding polygons with different BFE values, requiring knowledge of the expected probability distribution and magnitude of attribute error.

3. *Which data layer contributes the most to output uncertainty?*

By running the simulation multiple times, each time only considering positional errors in one layer, resulting uncertainty measures arising from source layers were calculated. Study wide averages showed error in the structure layer contributed most to output uncertainty.

4. *How do errors from individual inputs accumulate during spatial analysis?*

The isolated studies allowed an investigation into the accumulation of errors during spatial analysis. An assumption was made that the total error could be approximated by the square root of the sum of the squared errors from individual coverages. Comparing expected errors with those actually observed revealed that errors did accumulate as expected in approximately 90% of cases. The remaining observations reveal instances of error expansion (observed values were higher than expected) and error compensation (observed values were less than expected).

Addressing the above research questions has revealed that error propagation in GIS is a complex problem. Currently, this problem can only practically be addressed with Monte Carlo simulations. Additional methods are needed for handling correlation in input data and presenting meaningful uncertainty information based on observed error distributions.

APPENDIX
CODE

Fema.aml

```
/*----------------------------------------------------------------------
/*
/*    PROGRAM: FEMA.AML
/*
/*            This program carries out the reduced Erosion Hazards
/*            Analysis by attaching nominal and interpolated BFEs to
/*            structures and their projected 60-year positions.
/*
/*            This program is based on a similar application written by
/*            the author for 3001, Inc.
/*
/*
/*      INPUT: Program assumes 3 source coverages exist in current
/*            workspace: stru, zone, eha
/*
/*     OUTPUT: Stru coverage with positions of projected structures
/*            nominal and interpolated BFES at 10-year intervals
/*
/*     AUTHOR: Sam Henderson
/*
/*       DATE: 4/2000
/*
/*      CALLS:
/*
/*      NOTES:
/*
/*----------------------------------------------------------------------
&severity &error &routine BAILOUT

precision double

&call EHA_TO_CERF_PERF_BOX

&call DEFINE_TRANSECT

&call CALC_EHA_WIDTH
```

```
&call CREATE_PROJECTED_STRUCTURES

&call CREATE_BFE_SURFACE

&call OVERLAY

&call EXIT

&return

/*---------------------------
&routine EHA_TO_CERF_PERF_BOX
/*---------------------------
ae
&call SMALLTOL
ec eha
ef arc
sel type = 'cerf'
put cerf
sel type = 'perf'
put perf
sel type = ''
put box
save all yes
quit
build cerf line
build perf line
build box poly


&return

/*----------------------
&routine DEFINE_TRANSECT
/*----------------------
/* STEP 1: Calculate structure transect line through
/* structure perpendicular to the CERF using proximity anlysis (NEAR)

/* first, addxy to structures cover
addxy stru
tables
sel stru.pat
alter x-coord x0;;;;;;;;
alter y-coord y0;;;;;;;;
quit
```

```
/* tack on coords of near point
near stru cerf line 100000 # location
tables
sel stru.pat
alter x-coord x_near;;;;;;;;
alter y-coord y_near;;;;;;;;
alter distance disterf;;;;;;;
quit
dropitem stru.pat stru.pat cerf#

/* store vars for transect from and to coordinates
ae
&call SMALLTOL
ec stru
ef point
sel all
cursor open
&s i = 0
&do &while %:edit.aml$next%
  &s id%i% = %:edit.stru-id%
  &s fromx%i% = %:edit.x0%
  &s fromy%i% = %:edit.y0%
  &s tox%i% = %:edit.x_near%
  &s toy%i% = %:edit.y_near%
  &s i = [calc %i% + 1]
  cursor next
&end
&s numis = [calc %i% - 1]


/* Create line cover for each one, extend and put into transect coverage
create tran cerf
&do i = 0 &to %numis%
  &s linecov = [scratchname -prefix xxfema -dir]
  create %linecov% cerf
  coordinate key
  ef arc
  createattributes
  add
  2,[value fromx%i%],[value fromy%i%]
  2,[value tox%i%],[value toy%i%]
  9
  coordinate mouse
  cal [entryname [show ec]]-id = [value id%i%]
```

```
  /* extend to meet edges
  get box /* brings in box arcs
  nsel
  extend 10000

  put tran
  &if %i% gt 0 &then
    y;
  removeedit all yes

&end
coord mouse
save all yes
quit

build tran line

&return


/*---------------------
&routine CALC_EHA_WIDTH
/*---------------------
/* trim transects to erosion hazard area and find width.

/* overlay to set vars for lengths
identity tran eha tranout line .001
ae
&call SMALLTOL
ec tranout
ef arc
sel ineha = 'y'
cursor open
&do &while %:edit.aml$next%
  &s id = %:edit.tranout-id%
  &s width%id% = %:edit.length%
  cursor next
&end
sel ineha = 'n'
cursor open
&do &while %:edit.aml$next%
  &s id = %:edit.tranout-id%
  &if not [variable width%id%] &then /* only set if not already defined
    &s width%id% = 0
  cursor next
&end
```

```
save all yes
quit

/* attach widths from vars to structures
ae
&call SMALLTOL
ec stru
ef point
additem ehawidth 8 18 f 5
sel all
cursor open
&do &while %:edit.aml$next%
  &s id = %:edit.stru-id%
  &s :edit.ehawidth = [value width%id%]
  cursor next
&end

save all yes
quit


&return



/*---------------------------------
&routine CREATE_PROJECTED_STRUCTURES
/*---------------------------------

ae
&call SMALLTOL
ec stru
ef point
sel all
cursor open
&s num = 0
&do &while %:edit.aml$next%
  &s num = [calc %num% + 1]
  &s width = %:edit.ehawidth%
  &do i = 1 &to 6
    &s dist_yr%i%0 = [calc [calc %width% / 6] * %i%]

        &s id = %:edit.stru-id%
        &s link%num% = %id%
```

```
      &s x1 = %:edit.x0%
      &s y1 = %:edit.y0%
      &s x2 = %:edit.x_near%
      &s y2 = %:edit.y_near%



      /* With these two points, INVERSE for dist, az, new point xy.
      &s distance = [invdistance %x1%,%y1%,%x2%,%y2%] /* WHO CARES,.
      &s polar = [radang [invangle %x1%,%y1%,%x2%,%y2%]]
      &s az = [calc 360 - [calc %polar% - 90]]
      &if %az% gt 360 &then
        &s az = [calc %az% - 360]

      /* move to new point down line.
      &s progdist = [value dist_yr%i%0]
      &s dep = [calc %progdist% * [sin [angrad %az%]]]
      &s lat = [calc %progdist% * [cos [angrad %az%]]]
      &s newx%num%_%i%0 = [calc %x1% + %dep%]
      &s newy%num%_%i%0 = [calc %y1% + %lat%]
  &end

  cursor next

&end
&s totalnums = %num%
save all yes
quit


/* now, read var coords and create points.
copy stru pstr
ae
&call SMALLTOL
ec pstr
ef point
additem year 4 5 i
additem x 8 18 f 5
additem y 8 18 f 5
sel all /* just structures
cal year = 0
coord key
&do num = 1 &to %totalnums%
  &do i = 1 &to 6
    add
    1,[value newx%num%_%i%0],[value newy%num%_%i%0]
```

```
    ;
    cal year = %i%0
    cal x = [value newx%num%_%i%0]
    cal y = [value newy%num%_%i%0]
    cal pstr-id = [value link%num%]
  &end
&end
coord mouse
save all yes
quit

&return

/*-------------------------
&routine CREATE_BFE_SURFACE
/*-------------------------

build zone line

/* Add items.
&do item &list left_bfe right_bfe bfe_avg
  additem zone.aat zone.aat %item% 8 12 f 3
&end

/* relate to get c_bfe or p_bfe on the aat for left and right poly
relate add
left
zone.pat
info
lpoly#
zone#
linear
rw;;;
relate add
right
zone.pat
info
rpoly#
zone#
linear
rw;;;;

/* populate left and right bfes through relates
arcedit
&do type &list left right
  ec zone
```

```
  ef arc
  sel all
  cal %type%_bfe = %type%//bfe
&end
save all yes
quit

/* Calculate bfe averages on arcs.
cursor aatcur declare zone arc rw
cursor aatcur open  /*autoselects all

&do &while %:aatcur.aml$next%

  /* Normal case. (interior)
  &if %:aatcur.left_bfe% gt 0 and %:aatcur.right_bfe% gt 0 &then
    &s :aatcur.bfe_avg = [calc %:aatcur.left_bfe% / 2 + %:aatcur.right_bfe% / 2]

  /* Outside/border case. left poly -9999
  &if %:aatcur.left_bfe% le 0 and %:aatcur.right_bfe% gt 0 &then
    &s :aatcur.bfe_avg = %:aatcur.right_bfe%

  /* Outside/border case. Right poly -9999
  &if %:aatcur.left_bfe% gt 0 and %:aatcur.right_bfe% le 0 &then
    &s :aatcur.bfe_avg = %:aatcur.left_bfe%

  /* both negative case (nodata)
  &if %:aatcur.left_bfe% le 0 and %:aatcur.right_bfe% le 0 &then
  &s :aatcur.bfe_avg = -666


  /*go to next record
  cursor aatcur next

&end

/*close & remove cursor
cursor aatcur close
cursor aatcur remove

/* If any nodata arcs exist, delete them (-666)
arcedit
ec zone
ef arc
sel bfe_avg = -666
&if [show num sel] gt 0 &then
  delete
```

```
save all yes
quit
build zone poly

/* Convert arc cover to tin.
arctin zone bfetin line bfe_avg .001

relate drop
left;;
relate drop
right;;



&return


/*--------------
&routine OVERLAY
/*--------------

/* attach interpolated bfe to projected structures
tinspot bfetin pstr intbfe

/* attach nominal flood zone and bfe info to projected structures
identity pstr zone pstr2 point .001

/* relate all new items back to structures cover
&do year &list 0 10 20 30 40 50 60
  additem stru.pat stru.pat bfe%year% 3 4 i
&end
&do year &list 0 10 20 30 40 50 60
  additem stru.pat stru.pat intbfe%year% 8 12 f 3
&end
&do year &list 10 20 30 40 50 60
  additem stru.pat stru.pat x%year% 8 18 f 5
  additem stru.pat stru.pat y%year% 8 18 f 5
&end


/* make temp coverages for each year from projects structures
ae
&call SMALLTOL
ec pstr2
ef point
&do year &list 0 10 20 30 40 50 60
```

```
  sel year = %year%
  put pstr2_%year%
&end
save all yes
quit

/* relate items back to structures
&do year &list 0 10 20 30 40 50 60
  relate add
  relate%year%
  pstr2_%year%.pat
  info
  stru-id
  pstr2_%year%-id
  linear
  rw;;;
&end
tables
sel stru.pat
&do year &list 0 10 20 30 40 50 60
  &do item &list bfe intbfe
    cal %item%%year% = relate%year%//%item%
  &end
&end
&do year &list 10 20 30 40 50 60
  &do item &list x y
    cal %item%%year% = relate%year%//%item%
  &end
&end

quit

pullitems stru.pat stru.pat area,perimeter,stru#,stru-id,x0,y0,~
x_near,y_near,disterf,ehawidth,x10,y10,x20,y20,x30,y30,x40,y40,~
x50,y50,x60,y60,bfe0,bfe10,bfe20,bfe30,bfe40,bfe50,bfe60,intbfe0,~
intbfe10,intbfe20,intbfe30,intbfe40,intbfe50,intbfe60

kill pstr all
kill box all
kill tran all
dropitem tranout.aat tranout.aat tran-id tran# eha# eha-id area perimeter
rename tranout tran
dropitem pstr2.pat pstr2.pat pstr# pstr-id
rename pstr2 pstr
&do cov &list [listfile pstr2* -cover]
  kill %cov% all
```

```
&end
kill cerf all
kill perf all
&do cov &list [listfile xx* -cover]
  kill %cov% all
&end

&return


/*---------------
&routine SMALLTOL
/*---------------
weedtol .1
grain .1
editdistance .1
nodesnap first .1

&return

/*------------
&routine USAGE
/*------------
&type USAGE: FEMA.AML
&return &inform

/*-----------
&routine EXIT
/*-----------
/* Clean up and exit menu

/* Delete global vars

&return


/*--------------
&routine BAILOUT
/*--------------
&severity &error &ignore
&call exit
&return FEMA.AML is bailing out-;&type
```

Pes.aml

```
/*----------------------------------------------------------------------
/*          Positional Error Simulator
/*----------------------------------------------------------------------
/*
/*    PROGRAM: PES.AML
/*          "Positional Error Simulator"
/*          This program is a product of research conducted by
/*          Sam Henderson, University of Florida Geomatics Program -
/*          Read the online documents & tutorials before using:
/*          http://www.surv.ufl.edu/~sam
/*
/*     INPUT: Names, paths of input coverages
/*          1-sigma error radius for each coverage
/*          Items of interest for each coverage
/*          Number of iterations (realizations) (N)
/*          Name of AML with arguments
/*          Location of output directory
/*
/*    OUTPUT: Original covers with mean & standard deviation for items
/*          of interest.
/*
/*    AUTHOR: Sam Henderson
/*
/*      DATE: 01/2000 - 4/2000
/*
/*     CALLS: pes_main.menu, pes_main.aml, pes_add.aml, pes_add.menu
/*          randgen.exe, fat2rrt.exe
/*          (Expects NT system variable "PESCODE" = "x:\dir" that
/*          points to location of .exe files)

/*     NOTES:
/*          1. Tested only on Windows NT 4.0, ARC/INFO 7.2.1
/*          2. AML and MENU files should be located within the search
/*             directories specified by amlpath and menupath.
/*          3. An NT system variable should be set to the path of the
/*             EXE and APR files: Control Panel - System - Environment -
/*             and set a User variable of pescode equal to the value
/*             "x:\code" where x:\code is the full path to the directory that
/*             contains the EXE and APR files.
/*             (Do not include a backslash after the path)
/*          4. Items of interest must be numeric
/*          5. Items of interest: do not include static items (stdev = 0) -
/*             value must vary with each realization
/*          6. Input covers must be clean
/*          7. Number of items in any coverage limited by list function (1024 chars)
```

```
/*          8. Avoid numbers and periods "." in input cover names
/*          9. Coverages must be poly, line, or point only (
/*             multiple feature types not supported)
/*          10. Max records in any feature attribute table: 1000
/*          11. Max width for FAT unload including commas: 1000
/*          12. Max number of arcs, points, nodes, and vertices is 5000
/*          13. All point coverages will have x-coord, y-coord added/updated
/*          14. Polygon coverages: must have 1 label per polygon
/*          15. Polygon coverages: labels are cut and pasted into realizations
/*             (may cause problem with small polys, relative to error)
/*          16. User defined AML string should contain AML name and arguments
/*              (no ".aml")
/*          17. User defined AML should be simplified to assume covers in current
/*              workspace (no absolute paths)
/*          18. Cover-ids in "input" are original, those in "input+" and other
/*              output directories are modified
/*          19. Clean process assumes rectangular coordinate system
/*              (don't use with geographic)
/*          20. Error is at one-sigma level
/*              (68% of points/vertices fall within error radius)
/*          21. Line and Polygon coverages: if any two vertices or nodes are within
/*              4 times the estimated error, there is a chance of creating new
/*              records in realized coverages.  This will cause scrambling of the
/*              RRTables and invalid summary statistics.  To avoid this problem data
/*              should be cleaned, generalized or error estimates reduced.
/*----------------------------------------------------------------------
&severity &error &routine BAILOUT

/* set up environment
&call SET_ENV

/* get user input
&call GET_USER_INPUT

/* create output workspaces
&call CREATE_WORKSPACES

/* Query coverages and set vars, write file for ArcView
&call DESCRIBECOVERS
&call WRITE_AVFILE

/* randomize coverages, create coverages with error introduced
&call INTRODUCE_ERROR

/* run user aml on simulated data
&call RUN_USER_AML
```

```
/* re-organize all simulated data into single workspace
&call PERSIM_TO_ALLSIMS

/* export all attributes tables to ASCII files
&call UNLOAD_TABLES

/* reorganize per-cover to per-record (feature) txt files
&call FAT2RRT

/* conver RRT files into info tables
&call RRT2INFO

/* Run stats on rrt info files
&call STATS

/* Add record-id to summary tables
&call ADD_REC-ID

/* join summary data to input data
&call JOIN_STATS

/* rebuild topology
&call REBUILD

/* clean up and exit
&call EXIT

&return

/*--------------
&routine SET_ENV
/*--------------

/* Check program
&if [show program] ne ARC &then
  &return Run [before %AML$FILE% .aml] from ARC.

/* clean up any old vars
&dv .pes*

/* Start log file
&s logfile = [scratchname -prefix xxpeslog -file]
&watch %logfile%
&type start time: [date -full]
```

```
/* Store current environment.
&s curdisplay = [show display]
display 0
&term 9999 &mouse
&s .pes$homews = [show workspace]
&if not [null [show &amlpath]] &then
  &s curamlpath [show &amlpath]
&if not [null [show &menupath]] &then
  &s curmenupath [show &menupath]


/* Sets arctools paths.
&run [joinfile [joinfile $ATHOME lib -sub] setpaths -file]

/* search for all PES amls and menus in PATH>...

&return


/*--------------------
&routine GET_USER_INPUT
/*--------------------

/* create file to hold input data list
&if not [variable .pes$mainfile] &then &do
  &s .pes$mainfile = [scratchname]
  &s fileunit = [open %.pes$mainfile% openstat -write]
  &s closestat = [close %fileunit%]
&end

/* open pes main menu to get user input
&run pes_main init # # modal
&if %.pes_main$cancel% &then
  &call BAILOUT

/* read main menu file to get coverage name, error , items of interest
&s menufile = [open %.pes$mainfile% readstat -read]
&s i = 1
&do &while %readstat% = 0
  &s theline = [read %menufile% readstat]
  &if not [null %theline%] &then &do
    &s .PES$cover%i% = [extract 1 [unquote %theline%]]
    &s .PES$error%i% = [extract 2 [unquote %theline%]]
    &s .PES$dangle%i% = [extract 3 [unquote %theline%]]
    &s .PES$fuzzy%i% = [extract 4 [unquote %theline%]]
    &s .PES$genweed%i% = [extract 5 [unquote %theline%]]
```

```
   &s sofar = [value .PES$cover%i%],[value .PES$error%i%],[value
.PES$dangle%i%],[value .PES$fuzzy%i%],[value .PES$genweed%i%],
   &s .PES$ioilist%i% = [after [unquote %theline%] %sofar%]
   &s i = [calc %i% + 1]
  &end
&end
&s closestat = [close %fileunit%]
&s .pes$numcovs = [calc %i% - 1]

/* get number of iterations
&s .PES$iterations = %.pes_main$iterations%
&s .PES$0iterations = [calc %.PES$iterations% - 1]

/* check that aml exists
&s .PES$aml = %.pes_main$aml%
&s .PES$amllocation
&if not [null %.PES$aml%] &then &do
  &s theaml = [unquote %.PES$aml%]
  &if [token [unquote %.PES$aml%] -count] gt 1 &then  /*args exist
   &s theaml = [extract 1 [unquote %.PES$aml%]]
  &run [joinfile [joinfile $ATHOME misclib -sub] searchpaths -file] ~
   aml %theaml%.aml .PES$amllocation
  &if [null %.PES$amllocation%] &then
   &call BAILOUT
&end

/* system var PESCODE must be set before running this aml: "x:\dir"
&s .PES$randgen [joinfile %PESCODE% randgen.exe]
&s .PES$fat2rrt [joinfile %PESCODE% fat2rrt.exe]
&s .PES$aprfile [joinfile %PESCODE% pes.apr]

/* set var for output directory
&s .pes$outdir = %.pes_main$outdir%
&if not [exists %.pes$outdir% -dir] &then
  &sys mkdir %.pes$outdir%

/* Send user choices to screen.
&type -------------------
&type Input Data Summary:
&type -------------------
&type Number of coverages: %.PES$numcovs%
&do i = 1 &to %.PES$numcovs%
  &type ------
  &type Cover: [value .PES$cover%i%]
  &type Error: [value .PES$error%i%]
  &type Items: [value .PES$ioilist%i%]
```

```
  &type Dangle: [value .PES$dangle%i%]
  &type Fuzzy: [value .PES$fuzzy%i%]
  &type Genweed: [value .PES$genweed%i%]
&end
&type
&type -------------------
&type Processing Options:
&type -------------------
&type Iterations: %.PES$iterations%
&type AML String: %.PES$aml%
&type -------------------

&return


/*------------------------
&routine CREATE_WORKSPACES
/*------------------------

/* set up some vars
&s .PES$inputws [joinfile %.PES$outdir% input -sub]
&s .PES$tempws [joinfile %.PES$outdir% temp -sub]
&s .PES$input+ws [joinfile %.PES$outdir% input+ -sub]
&s .PES$realizationsws [joinfile %.PES$outdir% realizations -sub]
&s .PES$realeachws [joinfile %.PES$realizationsws% each -sub]
&s .PES$realallws [joinfile %.PES$realizationsws% all -sub]
&s .PES$tablesws [joinfile %.PES$outdir% tables -sub]
&s .PES$fattabsws [joinfile %.PES$tablesws% fat -sub]
&s .PES$rrttabsws [joinfile %.PES$tablesws% rrt -sub]
&s .PES$sumtabsws [joinfile %.PES$tablesws% sum -sub]

/* Make output workspaces
cw %.PES$inputws%
cw %.PES$tempws%
cw %.PES$input+ws%
&sys mkdir %.PES$realizationsws%
&sys mkdir %.PES$realeachws%
cw %.PES$realallws%
&sys mkdir %.PES$tablesws%
cw %.PES$fattabsws%
cw %.PES$rrttabsws%
cw %.PES$sumtabsws%

/* create workspaces for each iteration
&do num = 0 &to %.PES$0iterations%
 cw [joinfile %.PES$realeachws% real%num% -sub]
```

```
&end

/* Copy coverages to input and temp workspaces
&do i = 1 &to %.PES$numcovs%
  &s ename = [entryname [value .PES$cover%i%]]
  copy [value .PES$cover%i%] [joinfile %.PES$inputws% %ename%]
  copy [value .PES$cover%i%] [joinfile %.PES$tempws% %ename%]
&end

/* redirect all cover vars to points to those in "tempws"
&do i = 1 &to %.PES$numcovs%
  &s ename = [entryname [value .PES$cover%i%]]
  &s .PES$cover%i% [joinfile %.PES$tempws% %ename% -sub]
&end

&return


/*---------------------
&routine DESCRIBECOVERS
/*---------------------
/* Determine if point, line, or poly
&do i = 1 &to %.PES$numcovs%
  &describe [value .PES$cover%i%]
  &s .PES$cover%i%type = poly
  &s .PES$cover%i%fat = pat
  &s .PES$cover%i%numpolys = %DSC$POLYGONS%
  &if %DSC$PAT_BYTES% le 0 &then &do
    &s .PES$cover%i%type = line
    &s .PES$cover%i%fat = aat
    &s .PES$cover%i%numarcs = %DSC$ARCS%
  &end
  &if not %DSC$QTOPOLOGY% and %DSC$XAT_BYTES% gt 0 &then &do /* no
polys.
    &if %DSC$AAT_BYTES% gt 0 &then &do /* arcs exist
      &s .PES$cover%i%type = line
      &s .PES$cover%i%fat = aat
      &s .PES$cover%i%numarcs = %DSC$ARCS%
    &end
    &else &do /* Only points.
      &s .PES$cover%i%type = point
      &s .PES$cover%i%fat = pat
      &s .PES$cover%i%numpoints = %DSC$POINTS%
    &end
  &end
  &type Coverage: [value .PES$cover%i%]
```

```
 &type   is type: [upcase [value .PES$cover%i%type]]

 /* check if poly coverage has arc attributes,keep them
 &s .PES$cover%i%pataat .FALSE.
 &if [value .PES$cover%i%type] = poly &then &do
   &if [exists [value .PES$cover%i%].aat -info] &then &do
     &s numitems = [listitem [value .PES$cover%i%].aat -info xxpestmp.txt]
     &if %numitems% gt 7 &then /* arc items exist
       &s .PES$cover%i%pataat .TRUE.
       &s .PES$cover%i%numarcs = %DSC$ARCS%
   &end
 &end



&end

&return

/*-------------------
&routine WRITE_AVFILE
/*-------------------

&s avfileunit = [open [joinfile %.PES$outdir% avfile.txt] openstat -write]
&s writestat = [write %avfileunit% %.PES$numcovs%]
&do i = 1 &to %.PES$numcovs%
 &s cov = [entryname [value .PES$cover%i%]]
 &s type = [value .PES$cover%i%type]
 &s writestat = [write %avfileunit% [quote %cov%,%type%]]
&end
&s writestat = [write %avfileunit% %.PES$iterations%]
&s closestat = [close %avfileunit%]

/* copy pes.apr project from PESCODE sysvar to output
&s copystat [copy %.PES$aprfile% [joinfile %.PES$outdir% pes.apr]]

&return


/*----------------------
&routine INTRODUCE_ERROR
/*----------------------
/* Pass each coverage with error (not 0) through randomization

&do i = 1 &to %.PES$numcovs%
```

```
/* Set some short variables for coverage name, path
&s type = [value .PES$cover%i%type]
&s fat = [value .PES$cover%i%fat]
&s covfull = [value .PES$cover%i%]
&s covname = [entryname [value .PES$cover%i%]]
&s error = [value .PES$error%i%]
&s dangle = [value .PES$dangle%i%]
&s fuzzy = [value .PES$fuzzy%i%]
&s genweed = [value .PES$genweed%i%]

/* clean and/or generlize coverage
&if %type% ne point &then &do
  &if %genweed% ne 0 &then &do
    &s tmpname = [scratchname -prefix xxpes -dir -full]
    copy %covfull% %tmpname%
    kill %covfull% all
    generalize %tmpname% %covfull% %genweed%
    build %covfull% %type%
    kill %tmpname% all
    kill [joinfile %.PES$inputws% %covname%] all
    copy %covfull% [joinfile %.PES$inputws% %covname%]
  &end
  &if %dangle% ne 0 &then &do
    &s tmpname = [scratchname -prefix xxpes -dir -full]
    copy %covfull% %tmpname%
    kill %covfull% all
    clean %tmpname% %covfull% %dangle% %fuzzy% %type%
    kill %tmpname% all
    kill [joinfile %.PES$inputws% %covname%] all
    copy %covfull% [joinfile %.PES$inputws% %covname%]
  &end

  /*update numarcs
  &describe %covfull%
  &s .PES$cover%i%numarcs = %DSC$ARCS%
  &s .PES$cover%i%numpolys = %DSC$polygons%

&end

/* build features
&select %type%
&when point
  build %covfull% point
&when line
  &do
    build %covfull% node
```

```
    build %covfull% line
  &end
&when poly
  &do
    createlabels %covfull%
    build %covfull% node
    build %covfull% line
    build %covfull% poly
  &end
&end

/* store original user-ids in new item
&workspace %.PES$tempws%
tables
&if not [iteminfo %covname%.%fat% -info pes_orig-id -exists] &then
  additem %covname%.%fat% pes_orig-id 4 5 b
sel %covname%.%fat%
cal pes_orig-id = %covname%-id
quit

/* Make cover-id unique (=$recno)
ae
ec %covfull%
&select %type%
&when point
  ef point
&when line,poly
  ef line
&end
sel all
cal %covname%-id = %covname%#
save all yes
quit

/* copy covers with unique user-ids to input+
copy %covfull% [joinfile %.PES$input+ws% %covname% -sub]

/* Store original polygon labels to build pat later
&if %type% = poly &then &do
  ae
  ec %covfull%
  ef label
  sel all
  &s polylabels [scratchname -prefix xxpes -dir -full]
  put %polylabels%
  save all yes
```

```
 quit
&end

/* Update describe elements after build.
&describe %covfull%

/* Store number of features (records
&select %type%
  &when point
    &s .PES$cover%i%recs = %DSC$POINTS%
  &when poly   /* Exclude universe poly
    &s .PES$cover%i%recs = [calc %DSC$POLYGONS% - 1]
  &when line
    &s .PES$cover%i%recs = %DSC$ARCS%
&end

/* Ungenerate coverage
&if %type% = poly &then
  ungenerate line %covfull% %covfull%.gen
&else
  ungenerate %type% %covfull% %covfull%.gen

/* copy randgen.c program from path defined by PESCODE sys var
&s cprogram = [entryname %.PES$randgen%]
&s copystat [copy %.PES$randgen% [joinfile %.PES$tempws% %cprogram%]]

&workspace %.PES$tempws%
&if %type% = line or %type% = poly &then &do
  tables
  sel %covname%.aat
  &s formatfile = [scratchname -prefix xxpes -file]
  unload %covname%.aat %covname%-id fnode# tnode# columnar %formatfile%
  quit
  &data %cprogram%
    line
    %covname%.aat
    %DSC$ARCS%
    %DSC$MAX_NODE%
    %covname%.gen
    %error%
    %.PES$iterations%
  &end
&end
&if %type% = point &then &do
  &data %cprogram%
    point
```

```
    %covname%.gen
    %DSC$POINTS%
    %error%
    %.PES$iterations%
  &end
&end
&workspace %.PES$homews%

/* Generate coverages
&do num = 0 &to %.PES$0iterations%
  generate [joinfile %.PES$realeachws% ~
   [joinfile real%num% %covname% -sub] -sub]
  input [joinfile %.PES$tempws% %num%x%covname%.gen]
  &if %type% = point &then
    points
  &else
    lines
  quit
  &s realcov = [joinfile %.PES$realeachws% ~
   [joinfile real%num% %covname% -sub] -sub]
  &if %type% = line or %type% = poly &then
    clean %realcov% # .00001 .00001 line
  &if %type% = point &then &do
    build %realcov% point
    addxy %realcov%   /* standard item now, like area,perimeter,length.
  &end
  &if %type% = poly &then
    build %realcov% poly

  /* Check same number of features after randomization, bailout if not
  &describe %realcov%
  &select %type%
    &when poly
      &do
        &if %DSC$POLYGONS% ne [value .PES$cover%i%numpolys] &then &do
          &type;&type [value .PES$cover%i%numpolys] polys became
%DSC$POLYGONS%
           &call BAILOUT
        &end
      &if [value .PES$cover%i%pataat] &then &do
          &if %DSC$ARCS% ne [value .PES$cover%i%numarcs] &then &do
            &type;&type [value .PES$cover%i%numarcs] arcs became %DSC$ARCS%
          &call BAILOUT
          &end
        &end
      &end
```

```
    &when arc
    &do
      &if %DSC$ARCS% ne [value .PES$cover%i%numarcs] &then &do
        &type;&type [value .PES$cover%i%numarcs] arcs became %DSC$ARCS%
      &call BAILOUT
      &end
     &end
  &end

&end



/* link original attributes to simulated data
&if %type% = point or %type% = line &then &do
  &do num = 0 &to %.PES$0iterations%
    &s realcov = [joinfile %.PES$realeachws% ~
     [joinfile real%num% %covname% -sub] -sub]
    joinitem %realcov%.%fat% %covfull%.%fat% ~
     %realcov%.%fat% %covname%-id %covname%-id
  &end
&end
&if %type% = poly &then &do
  &do num = 0 &to %.PES$0iterations%
    &s realcov = [joinfile %.PES$realeachws% ~
     [joinfile real%num% %covname% -sub] -sub]
    joinitem %realcov%.pat %covfull%.pat %realcov%.pat %covname%-id ~
     %covname%-id  /*values scrambled but gets items
    ae
    ec %realcov%
    ef label
    get %polylabels%
    save all yes
    quit
    build %realcov% poly

    /* get arc attributes too.
    &if [variable .pes$cover%i%pataat] &then &do
      &if [value .PES$cover%i%pataat] &then
        joinitem %realcov%.aat %covfull%.aat %realcov%.aat %covname%-id ~
         %covname%-id
    &end
  &end
  kill %polylabels% all
&end
```

```
&end /* end of cover loop

&return


/*-------------------
&routine RUN_USER_AML
/*-------------------
/* Run user AML on simulated data
&if not [null %.PES$aml%] &then &do

  /* first run on source data in input directory
  &workspace %.PES$inputws%
  &run [unquote %.PES$aml%]

  /* run on input+ data
  &workspace %.PES$input+ws%
  &run [unquote %.PES$aml%]

  /* run on simulated data
  &do i = 0 &to %.PES$0iterations%
   &workspace [joinfile %.PES$realeachws% real%i% -sub]
   &run [unquote %.PES$aml%]
  &end

&end

&return


/*------------------------
&routine PERSIM_TO_ALLSIMS
/*------------------------
/* copy all simulated data into single workspace
&do w = 0 &to %.PES$0iterations%
  &do i = 1 &to %.PES$numcovs%
    &s ename = [entryname [value .PES$cover%i%]]
    &s inwork = [joinfile %.PES$realeachws% real%w% -sub]
    copy [joinfile %inwork% %ename% -sub] ~
     [joinfile %.PES$realallws% %ename%%w% -sub]
  &end
&end
&return


/*--------------------
```

```
&routine UNLOAD_TABLES
/*--------------------fat
&workspace %.PES$realallws%
tables
&do w = 0 &to %.PES$0iterations%
  &do i = 1 &to %.PES$numcovs%
    &s ename = [entryname [value .PES$cover%i%]]
    &s table = %ename%%w%.[value .PES$cover%i%fat]
    &if [exists %table% -info] &then &do
      &s file [joinfile %.PES$fattabsws% %table%.txt]
      &s amlunit = [open %file% openstat -write]
      &s spaceslist [listitem %table% -info]
      &s commalist [translate [quote %spaceslist%] , ' ']
      &s outstat [write %amlunit% [quote %commalist%]]
      &s closestat = [close %amlunit%]
      sel %table%
      &if [value .PES$cover%i%type] = poly &then &do
        nsel
        asel $recno = 1 /* skip the universe poly
        nsel
      &end
      sort %ename%%w%-id
      unload [joinfile %.PES$fattabsws% %table%.txt]
      &if [value .PES$cover%i%type] = poly &then &do
        asel $recno = 1 /* get it back
      &end
      sort %ename%%w%#
    &end
  &end
&end

quit
&workspace %.PES$homews%
&return


/*--------------
&routine FAT2RRT
/*--------------

/* copy randgen.c program from wherever it is to tables dir
&s cprogram = [entryname %.PES$fat2rrt%]
&s copystat [copy %.PES$fat2rrt% [joinfile %.PES$fattabsws% %cprogram%]]

/* Re sort files
&workspace %.PES$fattabsws%
```

```
&do i = 1 &to %.PES$numcovs%
  &s fat = [value .PES$cover%i%fat]
  &s ename = [entryname [value .PES$cover%i%]]
  &s table = %ename%0.%fat%.txt
  &if [exists %table% -file] &then &do
    &data fat2rrt.exe
      %ename%
      %fat%
      [value .PES$cover%i%recs]
      %.PES$iterations%
    &end
    &do r = 1 &to [value .PES$cover%i%recs]
      &s file = %ename%%r%.%fat%.rrt
      &s file2 = [joinfile %.PES$rrttabsws% %ename%%r%.%fat%.rrt]
      &s copystat = [copy %file% %file2%]
      &s delstat = [delete %file%]
    &end
  &end
&end
&workspace %.PES$homews%

&return

/*---------------
&routine RRT2INFO
/*---------------
/* convert RRTs to INFO

&do i = 1 &to %.PES$numcovs%
  &s fat = [value .PES$cover%i%fat]
  &s ename = [entryname [value .PES$cover%i%]]

  /* use first (0) for template
  &s table = [joinfile %.PES$realallws% %ename%0.%fat%]
  &if [exists %table% -info] &then &do
    copyinfo %table% [joinfile %.PES$rrttabsws% %ename%.%fat%.rrt]
  &end
&end

&workspace %.PES$rrttabsws%
tables
&do i = 1 &to %.PES$numcovs%
  &s fat = [value .PES$cover%i%fat]
  &s ename = [entryname [value .PES$cover%i%]]
  &s table = %ename%.%fat%.rrt
  &if [exists %table% -info] &then &do
```

```
    additem %table% real# 4 5 b # $recno
    &do r = 1 &to [value .PES$cover%i%recs]
      copy %table% %ename%%r%.%fat%.rrt nodata
      sel %ename%%r%.%fat%.rrt
      add from %ename%%r%.%fat%.rrt
    &end
  sel
  kill %table%
  &end
&end
quit
&workspace %.PES$homews%

&return

/*-------------
&routine STATS
/*-------------
&do i = 1 &to %.PES$numcovs%
  &s fat = [value .PES$cover%i%fat]
  &s ename = [entryname [value .PES$cover%i%]]
  &do r = 1 &to [value .PES$cover%i%recs]
    &s table = [joinfile %.PES$rrttabsws% %ename%%r%.%fat%.rrt]
    &if not [null [value .PES$ioilist%i%]] &then &do
      &if [exists %table% -info] &then &do
        &do item &list [unquote [value .PES$ioilist%i%]]
          &if [iteminfo %table% -info %item% -exists] &then &do
            &s outstattable = [joinfile %.PES$sumtabsws% ~
             [entryname %table%].sum.%item%]
            statistics %table% %outstattable%
            mean %item%
            standarddeviation %item%
            end
          &end
          &else
            &type Item %item% not found in %table%
        &end

        /* recombine item tables
        &s realoutputsumtable = [joinfile %.PES$sumtabsws% ~
            [entryname %table%].sum]
        &do item &list [unquote [value .PES$ioilist%i%]]
          &s outstattable = [joinfile %.PES$sumtabsws% ~
            [entryname %table%].sum.%item%]
          &if not [exists %realoutputsumtable% -info] &then
            copyinfo %outstattable% %realoutputsumtable%
```

```
          &else
           joinitem %realoutputsumtable% %outstattable% %realoutputsumtable%
frequency
          &end
      &end
    &end
  &end
&end

&return


/*-----------------
&routine ADD_REC-ID
/*-----------------
&workspace %.PES$sumtabsws%
tables
&do i = 1 &to %.PES$numcovs%
  &s fat = [value .PES$cover%i%fat]
  &s ename = [entryname [value .PES$cover%i%]]
  &do r = 1 &to [value .PES$cover%i%recs]
    &s table = %ename%%r%.%fat%.rrt.sum
    &if [exists %table% -info] &then &do
      additem %table% %ename%-id 4 5 b # $recno
      sel %table%
      cal %ename%-id = %r%
      save %ename%[after %table% %r%].bdf /*binary disk file
      sel
    &end
  &end
&end

&do i = 1 &to %.PES$numcovs%
  &s fat = [value .PES$cover%i%fat]
  &s ename = [entryname [value .PES$cover%i%]]

  /* use first rec table as template
  &s table = %ename%1.%fat%.rrt.sum
  &if [exists %table% -info] &then &do
    copy %table% %ename%.%fat%.rrt.sum.all nodata
  &end
&end

&do i = 1 &to %.PES$numcovs%
  &s fat = [value .PES$cover%i%fat]
  &s ename = [entryname [value .PES$cover%i%]]
```

```
  &s table = %ename%.%fat%.rrt.sum.all
  &if [exists %table% -info] &then &do
    sel %table%
    get %ename%.%fat%.rrt.sum.bdf
  &end
&end

quit
&workspace %.PES$homews%

&return

/*-----------------
&routine JOIN_STATS
/*-----------------
&do i = 1 &to %.PES$numcovs%
  &s fat = [value .PES$cover%i%fat]
  &s ename = [entryname [value .PES$cover%i%]]

  /* use first rec table as template
  &s table = [joinfile %.PES$input+ws% %ename%.%fat%]
  &s sumtable = [joinfile %.PES$sumtabsws% %ename%.%fat%.rrt.sum.all]
  &if [exists %table% -info] and [exists %sumtable% -info] &then
    joinitem %table% %sumtable% %table% %ename%-id

&end


&return



/*--------------
&routine REBUILD
/*--------------

/* Set some short variables for coverage name, path
&do i = 1 &to %.PES$numcovs%
  &s cov+ = [joinfile %.PES$input+ws% ~
   [entryname [value .PES$cover%i%]] -sub]
  build %cov+% [value .PES$cover%i%type]
&end

&return

/*-----------
```

```
&routine EXIT
/*-----------
/* Clean up and exit menu

/* restore original workspace
&if [variable .PES$homews] &then
  &workspace %.PES$homews%

&if [variable curdisplay] and [show program] = ARC &then
  display %curdisplay%

/* finalize log file
&if [variable logfile] and [variable .PES$outdir] &then &do
  &type;&type Stop time: [date -full]
  &watch &off
  &s copystat = [copy %logfile% [joinfile %.PES$outdir% pes.log]]
  &s delstat = [delete %logfile%]
&end

/* delete the main menu list file
&if [variable .pes$mainfile] &then
&s delstat = [delete %.pes$mainfile% -file]

/* Delete global vars
&dv .pes*

&ty

&return


/*--------------
&routine BAILOUT
/*--------------
&severity &error &ignore
&call exit
&return PES.AML is bailing out-;&type
```

<u>Pes_main.aml</u>

```
/*-----------------------------------------------------------------------
/*          Positional Error Simulator
/*-----------------------------------------------------------------------
/*   Program: PES_MAIN.AML
/*   Purpose: Get input coverages, errors, items, iterations, and output
```

```
/*          directory to run simulations.
/*------------------------------------------------------------------------
/*   History: Sam Henderson - May 2000
/*=============================================================
============
/*
&args routine arglist:rest
&severity &error &routine bailout
/*
&s .pes_main$cancel .FALSE.
/* Check arguments
&if ^ [null %routine%] &then
  &call %routine%
&else
  &call usage
&return

/*-----------
&routine INIT   /*   {'position'} {'stripe'} {MODELESS | MODAL}
/*-----------
&dv .pes_main$*

/* Initialize tool interface
/*
&set position = [extract 1 [unquote %arglist%]]
&set stripe   = [extract 2 [unquote %arglist%]]
&set modality = [extract 3 [unquote %arglist%]]
&if [null %position%] or %position%_ = #_ &then
  &set position = &cc &screen &cc
&if [null %stripe%] or %stripe%_ = #_ &then
  &set stripe = Positional Error Simulator Version 1.0
&if [null %modality%] or %modality%_ = #_ &then
  &set mode =
&else
  &if [translate %modality%] = MODAL &then
    &set mode = &modal
  &else
    &set mode =
/*
/* Issue thread delete self if thread depth = 2 and input is tty
&if [show &thread &depth] = 2 and ~
 [extract 1 [show &thread &stack]] = tty &then
  &set launch = &thread &delete &self
&else
  &set launch
/*
```

```
&if [show &thread &exists tool$pes_main] &then
   &thread &delete tool$pes_main

&thread &create tool$pes_main %mode% ~
  &menu pes_main ~
  &position [unquote %position%] ~
  &stripe [quote [unquote %stripe%]] ~
  &pinaction '&run pes_main exit'
%launch%
/*
&return

/*----------
&routine ADD
/*----------
/* Open the add form to return a list of coverages.
&run pes_add init # # modal

/* append current cover and options to file
&if not %.pes_add$cancel% &then &do
  &s fileunit = [open %.pes$mainfile% openstat -append]
  &s writestat = [write %fileunit% [quote [quote %.pes_add$covops%]]]
  &s closestat = [close %fileunit%]
&end

&return


/*--------------------
&routine OUTPUT_BROWSE
/*--------------------
/* get output directory - must exist already
&r getdirectory init .pes_main$outdir * # [quote OUTPUT DIRECTORY]
&return


/*---------
&routine OK
/*---------
&call exit
&return

/*------------
&routine USAGE
/*------------
/* Display usage for this tool
```

```
&type Usage: PES_MAIN INIT {'"position"'} {'"stripe"'} {MODELESS | MODAL}
&return &inform

/*-------------
&routine CANCEL
/*-------------
&s .pes_main$cancel .TRUE.
&call EXIT
&return

/*-----------
&routine EXIT
/*-----------
/* Clean up and exit menu
&if [show &thread &exists tool$pes_main] &then
  &thread &delete tool$pes_main
&return

/*--------------
&routine BAILOUT
/*--------------
&severity &error &ignore
/* &call exit
&return &warning An error has occurred in routine: %routine% (PES_MAIN.AML).
```

Pes_main.menu

```
7
/*----------------------------------------------------------------------
/*          Positional Error Simulator
/*----------------------------------------------------------------------
/*   Program: PES_MAIN.MENU
/*   Purpose: Menu for PES_MAIN.AML
/*----------------------------------------------------------------------
/*   History: Sam Henderson - May 2000
/*=================================================================
============
/*%FORMINIT &run pes_main ADD

   Input Data: (cover, error, items...)
   %datalist1
```

```
   %butto0


    Iterations:          %edi2


    AML and Arguments (optional):
    %edit1


    Output Directory:
    %edi0                  %button1


   %but2         %button3
%datalist1 INPUT .pes_main$selected 40 KEEP SCROLL YES TYPEIN NO ROWS 6
CHOICE -FILE %.pes$mainfile% -NOSORT
%butto0 BUTTON KEEP 'Add...' &r pes_main add
%edi2 INPUT .pes_main$iterations 5 KEEP TYPEIN YES SCROLL NO SIZE 1024
INTEGER
%edit1 INPUT .pes_main$aml 27 KEEP TYPEIN YES SCROLL NO SIZE 1024
CHARACTER
%edi0 INPUT .pes_main$outdir 27 KEEP TYPEIN YES SCROLL NO SIZE 1024
CHARACTER
%button1 BUTTON KEEP 'Browse...' &run pes_main OUTPUT_BROWSE
%but2 BUTTON KEEP 'OK' &r pes_main ok
%button3 BUTTON KEEP 'Cancel' &r pes_main cancel
%FORMOPT NEXTFIELD ADVANCE SETVARIABLES IMMEDIATE
```

Pes_add.aml

```
/*------------------------------------------------------------------------
/*          Positional Error Simulator Version
/*------------------------------------------------------------------------
/*   Program: PES_ADD.AML
/*   Purpose: Add coverage, error, and items of interest to main input list.
/*------------------------------------------------------------------------
/*   History: Sam Henderson - May 2000
/*========================================================================
============
/*
&args routine arglist:rest
&severity &error &routine bailout
/*
&s .pes_add$cancel .FALSE.
/* Check arguments
```

```
&if ^ [null %routine%] &then
  &call %routine%
&else
  &call usage
&return

/*-----------
&routine INIT   /*  {'position'} {'stripe'} {MODELESS | MODAL}
/*-----------
&dv .pes_add$*
/* Initialize tool interface
/*
&set position = [extract 1 [unquote %arglist%]]
&set stripe   = [extract 2 [unquote %arglist%]]
&set modality = [extract 3 [unquote %arglist%]]
&if [null %position%] or %position%_ = #_ &then
  &set position = &cc &screen &cc
&if [null %stripe%] or %stripe%_ = #_ &then
  &set stripe = Positional Error Simulator Version 1.0
&if [null %modality%] or %modality%_ = #_ &then
  &set mode =
&else
  &if [translate %modality%] = MODAL &then
    &set mode = &modal
  &else
    &set mode =
/*
/* Issue thread delete self if thread depth = 2 and input is tty
&if [show &thread &depth] = 2 and ~
 [extract 1 [show &thread &stack]] = tty &then
  &set launch = &thread &delete &self
&else
  &set launch
/*
&if [show &thread &exists tool$pes_add] &then
   &thread &delete tool$pes_add
&thread &create tool$pes_add %mode% ~
  &menu pes_add ~
  &position [unquote %position%] ~
  &stripe [quote [unquote %stripe%]] ~
  &pinaction '&run pes_add exit'
%launch%
/*
&return
```

```
/*-------------
&routine BROWSE
/*-------------
/* Open the add form to return a list of coverages.
&r getcover init .PES_ADD$cover
&return


/*---------
&routine OK
/*---------

/* Check error field.
&s noproblems .TRUE.


&if %noproblems% &then &do
  &s .pes_add$covops = ~

%.pes_add$cover%,%.pes_add$error%,%.pes_add$dangle%,%.pes_add$fuzzy%,%.pes_
add$genweed%,%.pes_add$items%
  &call EXIT
&end

&return

/*------------
&routine USAGE
/*------------
/* Display usage for this tool
&type Usage: PES_ADD INIT {'''position'''} {'''stripe'''} {MODELESS | MODAL}
&return &inform

/*-------------
&routine CANCEL
/*-------------
&s .pes_add$cancel .TRUE.
&call EXIT
&return

/*-----------
&routine EXIT
/*-----------
/* Clean up and exit menu
&if [show &thread &exists tool$pes_add] &then
  &thread &delete tool$pes_add
```

```
&return

/*--------------
&routine BAILOUT
/*--------------
&severity &error &ignore
/* &call exit
&return &warning An error has occurred in routine: %routine% (PES_ADD.AML).
```

<u>Pes_add.menu</u>

```
7
/*------------------------------------------------------------------------
/*          Positional Error Simulator
/*------------------------------------------------------------------------
/*   Program: PES_ADD.MENU
/*   Purpose: Menu for PES_ADD.AML
/*------------------------------------------------------------------------
/*   History: Sam Henderson - May 2000
/*========================================================================
============
    Coverage:
  %display0
  %button0


    1-Sigma error radius:              %edi0


    Clean dangle and fuzzy:   %edi2      %edi3


    Generalize weed tolerance:          %edi4



    Items of interest (item1,item2...):
   %edi1


      %but1       %button2
%display0 DISPLAY .pes_add$cover 45 VALUE
%button0 BUTTON KEEP 'Browse...' &r pes_add browse
%edi0 INPUT .pes_add$error 5 KEEP TYPEIN YES SCROLL NO SIZE 1024
INTEGER
```

```
%edi2 INPUT .pes_add$dangle 5 KEEP TYPEIN YES SCROLL NO SIZE 1024
INTEGER
%edi3 INPUT .pes_add$fuzzy 5 KEEP TYPEIN YES SCROLL NO SIZE 1024
INTEGER
%edi4 INPUT .pes_add$genweed 5 KEEP TYPEIN YES SCROLL NO SIZE 1024
INTEGER
%edi1 INPUT .pes_add$items 45 KEEP TYPEIN YES SCROLL NO SIZE 1024
CHARACTER
%but1 BUTTON KEEP 'OK' &r pes_add ok
%button2 BUTTON KEEP 'Cancel' &r pes_add cancel
%FORMINIT &run pes_add BROWSE
%FORMOPT NEXTFIELD ADVANCE SETVARIABLES IMMEDIATE
```

Randgen.c

```
/*----------------------------------------------------------------------
/*        Positional Error Simulator
/*----------------------------------------------------------------------
/*
/*   PROGRAM: randgen.c (for PES.AML)
/*        This program reads an ARC/INFO generate file and creates
/*        error perturbed generate files.
/*        This program is a product of research conducted by
/*        Sam Henderson, University of Florida Geomatics Program -
/*        Read the online documents & tutorials before using:
/*        http://www.surv.ufl.edu/~sam
/*
/*    INPUT: Line or point ARC/INFO generate file - will be prompted:
/*             line
/*                 aat file
/*                 number of arcs
/*                 number of nodes
/*                 generate file name
/*                 1 sigma error
/*                 number of iterations (N)
/*             point
/*                 generate file name
/*                 number of points
/*                 1 sigma error
/*                 number of iterations (N)
/*
/*    OUTPUT: N line or point generate files with error introduced
/*
/*    AUTHOR: Sam Henderson
/*
```

```
/*      DATE: May 2000
/*
/*      CALLS:
/*
/*      NOTES: See notes for PES.AML
/*
/*
/*-----------------------------------------------------------------------*/

#include    <stdio.h>
#include    <stdlib.h>
#include    <string.h>
#include    <math.h>
#include    <time.h>
#include        <io.h>


// define limits, constants
#define     MAXFILENAME     100              // input file name length
#define     MAXPTLENGTH     10               // point name length
#define     MAXFEATS        5000             // max number of features
#define     PI              3.14159265358979    // pi


// define structures
typedef struct {
   int id, fnode, tnode, verts;
} ATab;

typedef struct {
       double x, y;
} Ntab;

typedef struct {
       int id, fnode, tnode, vertpos;
       double x, y;
} Vtab;

typedef struct {
       int id;
       double x, y;
} Ptab;

typedef struct {
       double x, y;
} TmpNtab;
```

```
typedef struct {
       int fnode, tnode, vertpos;
       double x, y;
} TmpVtab;

typedef struct {
       int id;
       double x, y;
} TmpPtab;


// declare functions
void layout(double xo, double yo, double az, double dist, double *Px, double *Py);
double randnorm(void);
void d2e(char inputfilename[MAXFILENAME], char feature[10], int numfeats);


//-------------
void main(void)
//-------------
{
  char    aat_file[MAXFILENAME];
  char    point_gen_file[MAXFILENAME];
  char    line_gen_file[MAXFILENAME];
  char    outfilename[MAXFILENAME];
  ATab    arcs[MAXFEATS];
  Ptab    points[MAXFEATS];
  Ntab    nodes[MAXFEATS];
  Vtab    vertices[MAXFEATS];
  TmpNtab tmpnodes[MAXFEATS];
  TmpVtab tmpvertices[MAXFEATS];
  int     numarcs;              // number of arcs in coverage
  int     numpoints;            // number of points in coverage
  int     numnodes;             // number of nodes in coverage
  int     numverts;             // number of vertices in coverage
  FILE    *infile;
  FILE    *outfile;
  int     a,p,n,v;              // feature counters
  double  tmpx,tmpy;
  char    tmpstring[100];
  int     vertpos;              // vertex position along arc (1,2..)
  int     i,f;                  // record, file counters
  int     vcounter, vnum;       // vertex counters..
  char    feature[10];          // point, line, or poly
  float   radius = 0;           // standard error
```

```
int     numfiles;                // number of random output files
double  randaz, randdist;         // random azimuth and distance
double  randx, randy;             // output random x y coordinates
int     curid, curfnode, curtnode; // for arc loops
char    tmpcurid[10];



  // get feature class (point, line, poly)
  printf("Enter feature class: <POINT | LINE> --> ");
  scanf("%s", feature);

  // seed the rand function
  srand( (unsigned)time( NULL ) );

  // handle point coverages -----------------------------------------------------------
  if (strcmp(feature, "point") == 0) {

     // get name of point generate file
     printf("Enter name of space-delimited point GEN file --> ");
     scanf("%s", point_gen_file);

     // get number of points (records) in file
     printf("Number of points --> ");
     scanf("%d", &numpoints);

     // get 1 sigma error radius
     printf("\nEnter radius of 1 sigma error radius for POINTs --> ");
     scanf("%f", &radius);

     // get number of iterations
     printf("Number of random files to create --> ");
     scanf("%d", &numfiles);

     // Replace "D+" with "E+" in generate file and update filename
     d2e(point_gen_file, feature, numpoints);
     sprintf(tmpstring, "%s",point_gen_file);
     sprintf(point_gen_file, "%s%s","x", tmpstring);


           // open point gen file
           if ( (infile=fopen(point_gen_file,"r")) == NULL) {
                 printf("File: %s could not be opened.\n", point_gen_file);
                 exit(1);
           }
```

```
            // read in point coordinates to populate Ptab
            p=0;
            do {
               fscanf(infile, "%d %lf %lf", &points[p].id, &points[p].x, &points[p].y);
               p++;
            } while (p<numpoints);

            // close point generate file
            fclose(infile);

            /* write Ptab structure to screen
            printf("\nOriginal point data:\n");
            for (p=0; p<numpoints; p++) {
                  printf("ID = %d, X = %lf, Y = %lf
\n",points[p].id,points[p].x,points[p].y);
            }*/

            // For each iteration, randomize points and write gen file
            for (f=0; f<numfiles; f++) {

                  // create output generate file
                  sprintf(outfilename, "%d%s", f, point_gen_file);
                  if ( (outfile=fopen(outfilename,"w")) == NULL) {
               printf("File: %s could not be created.\n", outfilename);
                        exit(1);
                  }

                  // create generate file with random points
                  for (i=0; i<numpoints; i++) {

                        // find coordinates at az and dist from input point
                        randdist = radius*randnorm();
                        randaz = rand() % 361;
                        layout(points[i].x, points[i].y,randaz,randdist,&randx,&randy);
                        //printf("DIST: %lf,  AZ: %lf\n", randdist, randaz);
                        fprintf( outfile, "%d %lf %lf\n", i + 1, randx, randy );

                  }

                  // write eof info
                  fprintf( outfile, "END\n");
                  fclose(outfile);
            }
   }

      // handle line coverages----------------------------------------------------------
```

```
if (strcmp(feature, "line") == 0) {

        // get name of AAT file
        printf("Enter name of space-delimited AAT file --> ");
        scanf("%s", aat_file);

        // get number of arcs (records) in file
        printf("Number of arcs --> ");
        scanf("%d", &numarcs);

        // get number of nodes for Ntab
        printf("Number of nodes --> ");
        scanf("%d", &numnodes);

        // get name of line generate file
        printf("Enter name of space-delimited line GEN file --> ");
        scanf("%s", line_gen_file);

        // get 1 sigma error radius
        printf("Enter radius of 1 sigma error radius for LINEs --> ");
        scanf("%f", &radius);

        // get number of iterations
        printf("Number of random files to create --> ");
        scanf("%d", &numfiles);

        // Replace "D+" with "E+" in generate file and update filename
        d2e(line_gen_file, feature, numarcs);
        sprintf(tmpstring, "%s",line_gen_file);
        sprintf(line_gen_file, "%s%s","x", tmpstring);


        // open AAT file
        if ( (infile=fopen(aat_file,"r")) == NULL) {
                printf("File: %s could not be opened.\n", aat_file);
                exit(1);
        }

        // read through AAT file and store in ATab
        a=0;
        do {
                fscanf(infile, "%d %d %d", &arcs[a].id, &arcs[a].fnode, &arcs[a].tnode);
                arcs[a].verts = 0;
                a++;
        } while (a<numarcs);
```

```
// close AAT file
fclose(infile);

// open line gen file
if ( (infile=fopen(line_gen_file,"r")) == NULL) {
	printf("File: %s could not be opened.\n", line_gen_file);
	exit(1);
}

// start reading and populating Ntab and Vtab
curid = 0;
v = 0;

do {
	fscanf(infile, "%s", tmpcurid);
	if (strcmp(tmpcurid, "END") != 0)
		sscanf(tmpcurid, "%d", &curid);

	curfnode = 0;curtnode = 0;vertpos=0;

	// look up fnode# in Atab based on arc id
	a=0;
	while (arcs[a].id != curid) {
		a++;
	}
	curfnode = arcs[a].fnode;  // gets fnode at this arc
	curtnode = arcs[a].tnode;

	// store fnode x y in appropriate Ntab record
	fscanf(infile, "%lf %lf", &nodes[curfnode].x, &nodes[curfnode].y);

	// determine if next line is the tnode or a vertex - look for end
	fscanf(infile, "%lf %lf", &tmpx, &tmpy);
	fscanf(infile, "%s", tmpstring);

	if (strcmp(tmpstring, "END") != 0) { // tmpxy must be a vertex

		vertpos = 0;

		do {

			// store tmpx and tmpy in Vtab (sequential order)
			vertices[v].x = tmpx;
			vertices[v].y = tmpy;
			vertices[v].fnode = curfnode;
			vertices[v].tnode = curtnode;
```

```
                    vertices[v].id = v;
                    vertices[v].vertpos = ++vertpos;
                    arcs[a].verts = vertices[v].vertpos;
                    v++;

                    // current tmpstring isnt END, it is next x coord
                    sscanf( tmpstring, "%lf", &tmpx );

                    fscanf(infile, "%lf", &tmpy);
                    fscanf(infile, "%s", tmpstring);

                    if (strcmp(tmpstring, "END") == 0) {
                         nodes[curtnode].x = tmpx;
                         nodes[curtnode].y = tmpy;
                    }

               } while (strcmp(tmpstring, "END") != 0);

          }
          else {  // tmpxy isn't a vertex, its a tnode, store it
               nodes[curtnode].x = tmpx;
               nodes[curtnode].y = tmpy;

          }
     } while (strcmp(tmpcurid, "END") != 0);

     // store total number of v's from loop
     numverts = v;


     // close line gen file
     fclose(infile);

     // write structures to screen
     /*
     printf("\nNODE TABLE:\n");
     for (n=1; n<numnodes + 1; n++) {
          printf("n: %d, x: %lf, y: %lf\n", n,nodes[n].x, nodes[n].y);
     }
     printf("\nVERTEX TABLE:\n");
     for (v=0; v<numverts; v++) {
          printf("v: %d,FNODE = %d, TNODE = %d, VERTPOS = %d, x %lf, y is
%lf\n",
               v,vertices[v].fnode,vertices[v].tnode,vertices[v].vertpos,
                    vertices[v].x, vertices[v].y);
     }
```

```
        printf("\nARC TABLE:\n");
        for (a=0; a<numarcs; a++) {
            printf("ID = %d, FNODE = %d, TNODE = %d, NUMVERT = %d\n",
            arcs[a].id,arcs[a].fnode,arcs[a].tnode,arcs[a].verts);
        }*/


        // For each iteration, randomize nodes and vertices to
        // populate ith tmp structures Ntab and Vtab
        for (f=0; f<numfiles; f++) {


            //printf(" \n\n---------------  ITERATION %d  ---------------\n", f);
            // randomize nodes
            for (n=1; n<numnodes + 1;n++) {
                randdist = radius*randnorm();
                randaz = rand() % 361;
                layout(nodes[n].x, nodes[n].y,randaz,randdist,&randx,&randy);
                tmpnodes[n].x = randx;
                tmpnodes[n].y = randy;
                //printf("DIST: %lf,  AZ: %lf\n", randdist, randaz);
            }

            // randomize vertices
            for (v=0; v<numverts;v++) {
                randdist = radius*randnorm();
                randaz = rand() % 361;
                layout(vertices[v].x, vertices[v].y,randaz,randdist,&randx,&randy);
                tmpvertices[v].x = randx;
                tmpvertices[v].y = randy;
                //printf("DIST: %lf,  AZ: %lf\n", randdist, randaz);
            }


            // write random structures to screen
            /*
            printf("\n RANDOM NODE TABLE:\n");
            for (n=1; n<numnodes + 1; n++) {
                printf("n: %d, x: %lf, y: %lf\n", n,tmpnodes[n].x, tmpnodes[n].y);
            }
            printf("\nRANDOM VERTEX TABLE:\n");
            for (v=0; v<numverts; v++) {
                printf("v: %d,FNODE = %d, TNODE = %d, VERTPOS = %d, x %lf,
y is %lf\n",
                        v,vertices[v].fnode,vertices[v].tnode,vertices[v].vertpos,
                            tmpvertices[v].x, tmpvertices[v].y);
```

```
		}*/

		// create output generate file
		sprintf(outfilename, "%d%s", f, line_gen_file);
		if ( (outfile=fopen(outfilename,"w")) == NULL) {
		  printf("File: %s could not be created.\n", outfilename);
			exit(1);
		}

		vcounter = 0;
		for (a=0;a<numarcs;a++) {
			fprintf(outfile, "     %d\n", arcs[a].id);
			fprintf(outfile, "     %lf   %lf\n", tmpnodes[arcs[a].fnode].x, 
tmpnodes[arcs[a].fnode].y);

			if (arcs[a].verts == 0) {
				fprintf(outfile, "     %lf   %lf\n", tmpnodes[arcs[a].tnode].x, 
tmpnodes[arcs[a].tnode].y);
				fprintf(outfile, "END\n");
			}
			else {
				for (vnum=0;vnum<arcs[a].verts;vnum++) {
					fprintf(outfile, "     %lf   %lf\n", tmpvertices[vcounter].x, 
tmpvertices[vcounter].y);
					vcounter++;
				}
				fprintf(outfile, "     %lf   %lf\n", tmpnodes[arcs[a].tnode].x, 
tmpnodes[arcs[a].tnode].y);
				fprintf(outfile, "END\n");
			}

		}

		// write eof info
		fprintf(outfile, "END\n");
		fclose(outfile);

	}

   }

}  // end of main



//------------------
```

```
double randnorm(void)
//-------------------
// returns normally distributed data

{
   static int      firstnorm = 1;
   static double   normvar, normvar2;
   double   rmaxp2 = RAND_MAX + 2.0, rval1, rval2, mult, angle;

   if (firstnorm) {
      rval1 = (rand()+1.0) / rmaxp2;
      rval2 = (rand()+1.0) / rmaxp2;
      mult = sqrt( -2.0*log(rval1));
      angle = 2*PI*rval2;
      normvar = mult*sin(angle);
      normvar2 = mult * cos(angle);
   }
   else normvar = normvar2;
   firstnorm = 1 - firstnorm;
   return normvar;
}


//------------------------------------------------------------------
void layout(double xo, double yo, double az, double dist, double *Px,
         double *Py)
//------------------------------------------------------------------
{
   *Px = xo + (fabs(dist) * sin((PI/180) * az));
   *Py = yo + (fabs(dist) * cos((PI/180) * az));
}


//----------------------------------------------------------------------
void d2e(char inputfilename[MAXFILENAME], char feature[10], int numfeats)
//----------------------------------------------------------------------
{
      char    outfilename[MAXFILENAME];
      char    string[1000];
      FILE    *infile;
      FILE    *outfile;
      int            result;
      int            p,l,i;

      // open input gen file
      if ( (infile=fopen(inputfilename,"r")) == NULL) {
```

```
        printf("File: %s could not be opened.\n", inputfilename);
        exit(1);
    }

    // create output generate file
    sprintf(outfilename, "%s%s","x", inputfilename);
    if ( (outfile=fopen(outfilename,"w")) == NULL) {
          printf("File: %s could not be created.\n", outfilename);
        exit(1);
    }

if (strcmp(feature, "point") == 0) {
        p=0;
        i=0;
        do {
        fscanf(infile, "%s", string); // this is the id
            fprintf(outfile, "      %s      ", string);
            i=0;
            do {
                fscanf(infile, "%s", string);
                result = strchr( string, 'D' ) - string + 1;
                if (result > 0)
                    strncpy( string + result - 1, "E", 1 );
                fprintf(outfile, "%s    ", string);
                if (i==1)
                    fprintf (outfile, "\n");
                i++;
            } while (i<2);
            p++;
        } while (p<numfeats);
    }

    if (strcmp(feature, "line") == 0) {
        l=0;
        do {
        fscanf(infile, "%s", string);
            fprintf(outfile, "      %s\n", string);
            while (strcmp(string, "END") != 0) {
                fscanf(infile, "%s", string);
                if (strcmp(string, "END") == 0)  { // end of that arc section
                    l++;
                    fprintf(outfile, "END\n");
                }
                else {
                    result = strchr( string, 'D' ) - string + 1;
                    if (result > 0)
```

```
                        strncpy( string + result - 1, "E", 1 );
                    fprintf(outfile, "     %s     ", string);

                    fscanf(infile, "%s", string); // ycoord
                    result = strchr( string, 'D' ) - string + 1;
                    if (result > 0)
                        strncpy( string + result - 1, "E", 1 );
                    fprintf(outfile, "%s\n", string);
                }
            }
        } while (l<numfeats);
    }

    fprintf(outfile, "END\n");
    fclose(outfile);
    fclose(infile);

}
```

<u>Fat2rrt.c</u>

```
/*-----------------------------------------------------------------------
/*          Positional Error Simulator
/*-----------------------------------------------------------------------
/*
/*   PROGRAM: fat2rrt.c (for PES.AML)
/*          This program will re-sort input data by combining records
/*          into files
/*          This program is a product of research conducted by
/*          Sam Henderson, University of Florida Geomatics Program -
/*          Read the online documents & tutorials before using:
/*          http://www.surv.ufl.edu/~sam
/*
/*    INPUT: rootname of comma-delimited input files (from root0 - rootN)
/*           (.dat assumed)
/*          number of records per file (R)
/*          number of files (N)
/*
/*   OUTPUT: R files will be returned with N records each.
/*
/*   AUTHOR: Sam Henderson
/*
/*     DATE: May 2000
/*
/*    CALLS:
/*
```

```
/*      NOTES: See notes for PES.AML
/*
/*
/*---------------------------------------------------------------------*/
#include    <stdio.h>
#include    <stdlib.h>
#include    <string.h>

#define     MAXRECORDS      1000
#define     MAXRECLENGTH    1000

typedef struct {
   char record[MAXRECORDS];
} recordstruct;


//-------------
void main(void)
//-------------
{
   char             infilename[100];
   char             outfilename[100];
   FILE              *infile;
   FILE              *outfile;
   char             rootname[100];
   char             tabext[100];
   recordstruct        records[MAXRECORDS];
   char             record[MAXRECLENGTH];    // holds entire record
   char             header[MAXRECLENGTH];    // holds item names (row1)
   char             newheader[MAXRECLENGTH]; // holds item names (row1)
   int             rpf=0;                 // points per file
   int             nof=0;                 // number of files
   int             i;                    // counter up to rpf
   int             f;                    // counter up to nof
   int             n;                    // loop counter

   printf("Expecting files in the form:  root0.ext.txt\n");
   printf("Enter root name of file (.txt assumed) --> ");
   scanf("%s", rootname);

   printf("Enter table extension (aat) --> ");
   scanf("%s", tabext);

   printf("Enter number of records per file (not counting header row) --> ");
   scanf("%d", &rpf);
      rpf++;  // to include header row?
```

```
    printf("Enter total number of files (iterations) --> ");
    scanf("%d", &nof);

    for (i=0; i<rpf; i++) {

        for (f=0; f<nof; f++) {

                // create input file name and open
                sprintf(infilename, "%s%d%s%s%s", rootname, f, ".", tabext, ".txt");
                if ( (infile=fopen(infilename,"r")) == NULL) {
                        printf("File: %s could not be opened.\n", infilename);
                        exit(1);
                }

                // skip to the ith line and read into ith structure
                for (n=0; n<(i+1); n++) {
                        fscanf(infile, "%s",record);
                        if ((i==0) && (f==0))
                                strcpy(header, record);
                        if (n == i) {
                                strcpy(records[f].record, record);
                        }
                }
                fclose(infile);

        }

                // create output file
                if (i != 0) {
                        sprintf(outfilename, "%s%d%s%s%s", rootname, i, ".", tabext, ".rrt");
                        if ( (outfile=fopen(outfilename,"w")) == NULL) {
                                printf("File: %s could not be created.\n", outfilename);
                                exit(1);
                        }
                         // send current contents of structure to file.
                        sprintf(newheader, "%s%s", "SIM,", header);
                        //fprintf(outfile, "%s\n", newheader);
                        for (n=0;n<f;n++) {
                                fprintf(outfile, "%d%s%s\n", n, ",", records[n].record);
                        }
                        fclose(outfile);
                }
        }
}
```

LIST OF REFERENCES

3001, Inc., 1998a, "Selected Structures in FEMA 'Evaluation of Erosion Hazards' Study Area, Dare County, NC," Metadata document.

3001, Inc., 1998b, " Current Erosion Reference Feature, Dare County, NC," Metadata document.

3001, Inc., 1998c, "Current FIRM Flood Zones, Dare County, NC," Metadata document.

American Society for Photogrammetry and Remote Sensing (ASPRS), 1990, “ASPRS Accuracy Standards for Large-Scale Maps,” Photogrammetric Engineering & Remote Sensing, vol. 56, no. 7, pp. 1068-1070.

Anderson, J.M. and Mikhail, E.M, 1998, Surveying Theory and Practice, McGraw-Hill, New York.

Arbia, G., Griffith, D. and Haining, R., 1998, “Error Propagation Modelling in Raster GIS: Overlay Operations,” International Journal of Geographical Information Science, vol. 11, no. 5, pp. 409-434.

Blakemore, M., 1984, “Generalization and Error in Spatial Databases,” Cartographica, no. 21, pp. 131-139.

Bolstad, P.V., Gessler, P. and Lillesand, T.M., 1990a, "Positional Uncertainty in Manually Digitized Map Data," International Journal of Geographical Information Systems, vol. 12, no. 2, pp. 145-167.

Bolstad, P.V., Gessler, P. and Lillesand, T.M., 1990b, "A Variance Components Analysis of Manually Digitized Map Data," Surveying and Land Information Systems, vol. 50, no. 3, pp. 201-207.

Canters, F., 1997, “Evaluating the Uncertainty of Area Estimates Derived from Fuzzy Land-Cover Classification,” Photogrammetric Engineering & Remote Sensing, vol. 63, no. 4, pp. 403-414.

Chou, Y., 1969, Statistical Analysis, Holt, Rinehart and Winston, New York, pp. 798-799.

Davis, T.J. and Keller, C.P., 1997 "Modelling Uncertainty in Natural Resource Analysis using Fuzzy Sets and Monte Carlo Simulation: Slope Stability Prediction," International Journal of Geographical Information Science, vol. 11, no. 5, pp. 409-434.

Drummond, J., 1995: "Positional Accuracy," Elements of Spatial Data Quality, Elsevier Sciences Ltd, New York, pp. 31-57.

Dunn, R., Harrison, A.R. and White, J.C., 1990, "Positional Accuracy and Measurement Error in Digital Databases of Land Use: An Empirical Study," International Journal of Geographical Information Systems, vol. 4, no. 4, pp. 385-398.

Ehlers, M. and Shi, W., 1996, “Error Modelling for Integrated GIS,” Cartographica, vol. 33, no. 1, pp. 11-20.

Ehlschlaeger, C.R. and Goodchild, M.F., 1994, "Uncertainty in Spatial Data: Defining, Visualizing, and Managing Data Errors," GIS/LIS 1994 Proceedings, pp. 246-253.

Emmi, P.C. and Horton, C.A., 1995, "A Monte Carlo Simulation of Error Propagation in a GIS-Based Assessment of Seismic Risk," International Journal of Geographical Information Systems, vol. 9, no. 4, pp. 447-461.

Federal Emergency Management Agency (FEMA), 1998, "Flood Hazard Mapping," Map Modernization Work in Progress, Online: http://www.fema.gov/mit/tsd/MM_WIP1c.htm (03/08/00).

Federal Geographic Data Committee (FGDC), FGDC-STD-001-1998a, "Content Standard for Digital Geospatial Metadata (revised June 1998)," Federal Geographic Data Committee, Washington, D.C.

Federal Geographic Data Committee (FGDC), FGDC-STD-007.3-1998b, "Geospatial Positioning Accuracy Standards Part 3: National Standard for Spatial Data Accuracy," Federal Geographic Data Committee, Washington, D.C.

Fisher, P.F., 1991a, "Modelling Soil Map-Unit Inclusions By Monte Carlo Simulation," International Journal of Geographical Information Systems, vol. 5, no. 2, pp. 193-208.

Fisher, P.F., 1991b, “First Experiments in Viewshed Uncertainty: The Accuracy of the Viewshed Area,” Photogrammetric Engineering & Remote Sensing, vol. 57, no. 10, pp. 1321-1327.

Goodchild, M.F., 1989, "Modeling Error in Objects and Fields," Accuracy of Spatial Databases, Taylor & Francis Ltd, New York, pp. 107-113.

Goodchild, M.F., 1993, “Data Models and Data Quality: Problems and Prospects,” Environmental Modeling with GIS, Oxford University Press, New York, pp. 94-103.

Goodchild, M.F., 1995, "Attribute Accuracy," Elements of Spatial Data Quality, Elsevier Sciences Ltd, New York, pp. 59-79.

Goodchild, M.F. and Gopal, S., 1989, Accuracy of Spatial Databases, Taylor & Francis Ltd., New York.

Goodchild, M.F., Guoqing, S. and Shiren, Y., 1992, "Development and Test of an Error Model for Categorical data," International Journal of Geographical Information Systems, vol. 6, no. 2, pp. 87-104.

Guptill, S.C., 1989, "Inclusion of Accuracy data in Feature Based, Object-Oriented Data Model," Accuracy of Spatial Databases, Taylor & Francis Ltd., New York, pp. 91-97.

Heuvelink, G.B.M., Burrough, P.A. and Stein, A., 1989, "Propagation of Errors in Spatial Modelling with GIS," International Journal of Geographical Information Systems, vol. 3, no. 4, pp. 303-322.

Hunter, G.J., (7/31/2000), “Re: paper request.” E-mail to the author.

Hunter, G.J. and Goodchild, M.F., 1994, "Design and Application of a Methodology for Reporting Uncertainty in Spatial Databases," Journal of the Urban and Regional Information Systems Association, pp. 771-786.

Hunter, G.J. and Goodchild, M.F., 1995a, "Modeling the Uncertainty of Vector Data in Geographic Information Systems," ESRI 1995 User Conference Proceedings.

Hunter, G.J. and Goodchild, M.F., 1995b, "Dealing with Error in Spatial Databases: A Simple Case Study," Photogrammetric Engineering & Remote Sensing, vol. 61, no. 5, pp. 529-537.

Hunter, G.J. and Goodchild, M.F., 1996, “A New Model for Handling Vector Data Uncertainty in GIS,” Journal of the Urban and Regional Information Systems Association, 8, 1, pp. 51-57.

Hunter, G.J. and Goodchild, M.F., 1997 “Modeling the Uncertainty of Slope and Aspect Estimates Derived from Spatial Databases,” Geographical Analysis, vol. 29, no. 1, pp. 35-49.

Hunter, G.J. and Goodchild, M.F., 1999a, "New Tools for Handling Spatial Data Quality: Moving from Academic Concepts to Practical Reality," Journal of the Urban and Regional Information Systems Association, vol. 11, no. 2, pp. 25-34.

Hunter, G.J., Qiu, J. and Goodchild, M.F., 1999b, “Application of a New Model of Vector data Uncertainty,” Spatial Accuracy Assessment: Land Information Uncertainty in Natural Resources, Ann Arbor Press, Michigan, pp. 203-208.

Kiiveri, H.T., 1997, "Assessing, Representing and Transmitting Positional Uncertainty in Maps," International Journal of Geographical Information Science, vol. 11, no. 1, pp. 33-52.

Lanter, D.P. and Veregin, H., 1992, "A Research Paradigm for Propagating Error in Layer-Based GIS," Photogrammetric Engineering & Remote Sensing, vol. 58, no. 6, pp. 825-833.

Lee, J., Snyder, P.K. and Fisher, P.F., 1992, “Modeling the Effect of Data Errors on Feature Extraction from Digital Elevation Models,” Photogrammetric Engineering & Remote Sensing, vol. 58, no. 10, pp. 1461-1467.

Leung, Y. and Yan, J., 1998, "A Locational Error Model for Spatial Features," International Journal of Geographical Information Science, vol. 12, no. 6, pp. 607-620.

Maffini, G., Arno, M. and Bitterlich, W., 1989, "Observations and Comments on the Generation and Treatment of Error in Digital GIS Data," Accuracy of Spatial Databases, Taylor & Francis Ltd, New York, pp. 55-66.

Mikhail, E.M. and Ackerman, F., 1976, Observations and Least Squares,” Thomas Y. Crowell Company, Inc., University Press of America, Inc., Washington D.C., pp. 33.

Openshaw, S., 1989, "Learning to Live with Errors in Spatial Databases," Accuracy of Spatial Databases, Taylor & Francis Ltd, New York, pp. 263-276.

Shi, W., 1998, "A Generic Statistical Approach for Modelling Error of Geometric Features in GIS," International Journal of Geographic Information Science, vol. 12, no. 2, pp. 131-143.

Stanislawski, V., Dewitt, B.A., and Shrestha, R.L., 1996, “Estimating Positional Accuracy of Data Layers within a GIS through Error Propagation,” Photogrammetric Engineering & Remote Sensing, vol. 62, no. 4, pp. 429-433.

Thapa, K. and Burtch, R.C., 1991, "Primary and Secondary Methods of Data Collection in GIS/LIS," Surveying and Land Information Systems, vol. 51, no. 3, pp. 126-170.

Thapa, K. and Bossler, J., 1992, "Accuracy of Spatial Data Used in Geographic Information Systems," Photogrammetric Engineering & Remote Sensing, vol. 58, no. 6, pp. 835-841.

United States Geological Survey (USGS), 1999, National Map Accuracy Standards Fact Sheet FS-171-99, Online: http://rockyweb.cr.usgs.gov/nmpstds/nmas.html (09/15/00).

United States Geological Survey (USGS), National Mapping Division, Spatial Data Transfer Standard (SDTS), DRAFT for Review, November 20, 1997, American National Standards Institute, Inc.

Veregin, H., 1989a, "Error Modeling for the Map Overlay Operation," Accuracy of Spatial Databases, Taylor & Francis Ltd., New York, pp. 3-18.

Veregin, H., 1994, "Integration of Simulation Modeling and Error Propagation for the Buffer Operation in GIS," Photogrammetric Engineering & Remote Sensing, vol. 60, no. 4, pp. 427-435.

Veregin, H., 1995, "Developing and Testing of an Error Propagation Model for GIS Overlay Operations," International Journal of Geographical Information Systems, vol. 9, no. 6, pp. 595-619.

Veregin, H., 1996, "Error Propagation through the Buffer Operation for Probability Surfaces," Photogrammetric Engineering & Remote Sensing, vol. 62, no. 4, pp. 419-428.

Wechsler, S.P. “A Methodology for Digital Elevation Model (DEM) Uncertainty Evaluation: The Effect DEM Uncertainty on Topographic Parameters,” Journal of the Urban and Regional Information Systems Association, Online: http://campus.esri.com/campus/library/techpapers/urisa/1999/1999-21.pdf (09/15/00).

Wolf, P. and Ghilani, C., 1997, Adjustment Computations, John Wiley & Sons, Inc., New York.

Yuan, D., 1997, “A Simulation Comparison of Three Marginal Area Estimators for Image Classification,” Photogrammetric Engineering & Remote Sensing, vol. 63, no. 4, pp. 385-392.

## BIOGRAPHICAL SKETCH

Sam Henderson was born in Florida in 1976. He spent his elementary education in Gainesville and attended middle and high school in West Palm Beach. From 1990 to 1994 he studied classical piano at the Palm Beach County School of the Arts high school. In 1995 he entered the University of Florida and began his study of geomatics. He spent the warm summer of 1997 on I-95 as a rod-man in a surveying crew. He worked the next two years at 3001, Inc, a GIS surveying and mapping firm, where he developed an inclination for programming, frisbee, and ARC/INFO. He received his Bachelor of Science in 1999 and stayed at UF to continue studies in GIS. He received his Master of Science in December of 2000 and accepted a position at the National Imagery and Mapping Agency in Washington, D.C.