This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

A Spreadsheet Approach to Facilitate Visualization of Uncertainty in Information

Alexander Streit, Binh Pham, Ross Brown

***Abstract*—Information uncertainty is inherent in many problems and is often subtle and complicated to understand. While visualization is a powerful means for exploring and understanding information, information uncertainty visualization is ad hoc and not widespread. This paper identifies two main barriers to the uptake of information uncertainty visualization: firstly, the difficulty of modeling and propagating the uncertainty information; and secondly, the difficulty of mapping uncertainty to visual elements.**

To overcome these barriers, we extend the spreadsheet paradigm to encapsulate uncertainty details within cells. This creates an inherent awareness of the uncertainty associated with each variable. The spreadsheet can hide the uncertainty details, enabling the user to think simply in terms of variables. Furthermore, the system can aid with automated propagation of uncertainty information, since it is intrinsically aware of the uncertainty. The system also enables mapping the encapsulated uncertainty to visual elements via the formula language and a *visualization sheet*. Support for such low-level visual mapping provides flexibility to explore new techniques for information uncertainty visualization.

***Index Terms*—Uncertainty Visualization, Information Uncertainty, Fuzzy Visualization, Visualization Process, Visualization Framework, Information Modeling**

I. Introduction

INFORMATION UNCERTAINTY is present in many fields and numerous modeling techniques exist to manage this uncertainty. Using information uncertainty modeling techniques not only provides greater confidence in results, but can also give an indication of *how much* confidence to place in the result. Visualization is a powerful tool for exploring and understanding information. However, when it comes to visualizing information with uncertainty, too often the information is treated as though it were entirely certain, thus discarding valuable knowledge.

The term *uncertainty* has broad meaning and can relate to numerous aspects of uncertainty, for some examples see [16]. This paper deals with uncertainty about the true value of information, which we refer to as *information uncertainty*. Such uncertainties arise due to predictions, errors or imprecision in measurement, linguistic ambiguity or vagueness, lacking or insufficient information, and similar sources. The common theme is that the uncertainty can be characterized for a particular unit of information.

Despite ongoing research into information uncertainty visualization methods (e.g. [5], [6], [11], [19]), information uncertainty visualization has failed to gain widespread acceptance. Barriers to the uptake of information uncertainty visualization include an artificial separation between modeling of information and its uncertainty, and the need for non-standard display techniques. This combination often results in ad hoc visualizations. From a practical point of view, the tools that users employ are not conducive to uncertainty modeling and visualization, which results in a tendency by users to ignore uncertainty for visualization.

To overcome these barriers, we present a visualization system based on a spreadsheet paradigm that inherently supports modeling, propagation, and visualization of information uncertainty. Facilitating this system requires the integration of uncertainty modeling techniques into a hierarchical order, encapsulation of uncertainty parameters in highly cohesive polymorphic data types, and access to a sufficiently general form of uncertainty when mapping to visual elements. Spreadsheets are ubiquitous, intuitive, and offer several other advantages, such as immediate feedback when changes are made.

The significance of our spreadsheet approach is fourfold. Firstly, our spreadsheet does not require additional cells to hold uncertainty details when they are added, which prevents the complexity of the spreadsheet layout from spiraling out of control. Secondly, adding uncertainty information does not change spreadsheet layout, hence the user's model is represented identically to a model without uncertainty details. Thirdly, the user's formulae remain unchanged, which makes them easier to read and understand, and not specific to any particular uncertainty modeling type. Fourthly, abstracting the mapping of uncertainty information to visual elements

The authors are with the Faculty of Information Technology at the Queensland University of Technology, Australia. Email: a.streit@qut.edu.au, b.pham@qut.edu.au, r.brown@qut.edu.au

Digital Object Indentifier 10.1109/TVCG.2007.70426

allows the user to change uncertainty information in the model without destroying the visualizations that depend on the model.

This paper is organized as follows. Section 2 reviews related work on spreadsheets and visualization. Section 3 discusses issues facing users when they seek to visualize information uncertainty, drawing out desired characteristics of solutions motivating our work. Section 4 describes encapsulating uncertainty within information, such that the semantics of the parameters and propagation of the uncertainty is given structural support, as well as proposing a general uncertainty form for use with visualization. Section 5 details the motivation and design for a spreadsheet-based integrated modeling and visualization system and presents capabilities of such an approach. Section 6 uses a case study in financial modeling to demonstrate the new framework's improvements in accessibility, power, and visualization sophistication over traditional approaches.

II. Related Work

The spreadsheet paradigm widely understood for managing numerical information, prompting researchers to explore other uses. An early proposal to generalize spreadsheets is the Analytical Spreadsheet Package [21] (ASP), which allowed any Smalltalk-80 object to be placed inside a cell and used Smalltalk messages as formulae. While this provides flexibility, it is too general and complicated for non-expert users to understand. However, ASP did anticipate many of the uses explored by works that followed, such as widgets, which are available in Spreadsheets for Images [15] (SI). SI extends traditional spreadsheets to include graphical objects, including several different widgets and images. Further, SI takes the unusual step of allowing formulae to write their results to a different cell. While this offers flow control, it can complicate the user's interpretation of the spreadsheet.

FINESSE [27] specifically targeted real-time financial information, adding images, heat maps, and graphs to the regular cell types. FINESSE introduced "presentation relationships", where groups of cells have access to common presentation attributes. This provides for shared memory that is not shown in a cell. However, other systems (including SI) achieve a similar effect by storing presentation attributes in cells.

The Spreadsheet for Information Visualization[7], [8] (SIV) explored more general visualization, building on the Visualization Toolkit [24] (VTK). Each cell in SIV can contain a visualization, including the data sets used to drive the visualization. Visualization related operators are available and can operate on multiple cells, such as a whole column. SIV is motivated by the ability to compare visualizations side-by-side, particularly to see incremental changes, which is often referred to as "small multiples" after [25, pp. 67]. Further, a key advantage of spreadsheets is to use templates for analysis and experimentation. While suited to visualization tasks, SIV is not particularly suited to modeling as it is optimized for fewer cells containing larger datasets and has dispensed with traditional text and numerical cells.

VisTrails [1], [3] specifically uses a spreadsheet for displaying multiple visualizations for side-by-side exploration. In this sense the term spreadsheet refers to the tabular appearance rather than any ability to create formula driven relationships. Similarly, tabular visualization methods, such as Hyperslice [26] and TableLens [22], share some similarity to spreadsheets. However, traditional spreadsheets are sparse, allow a mix of cells, and offer inter-cell dependencies through formulae. These properties lend to a *paper-likeness* that separates spreadsheets from tabular displays.

Prior work on uncertainty visualization frameworks includes the multi-agent framework for supporting visualization of fuzzy systems [20]. This framework consists of multiple agents that coordinate to deliver appropriate options to the user, with the aim of being context-sensitive and relevant. Our framework aims to provide an environment conducive to modeling and visualization of information uncertainty and provides users with flexibility when it comes to visual mappings. However, the two are not mutually exclusive and the multi-agent approach could be incorporated in the visualization system component of our framework. The profile agent of the multi-agent framework learns user preferences to tune the options that are displayed to the user. This is similar to [10], which details a formal notation and calculus for visualization exploration. Both of these are intended to improve visualization workflow and could be integrated into the framework described here, however this is beyond the scope of this paper.

III. Issues in Information Uncertainty Visualization

This section examines the issues that confront users when they seek to visualize information uncertainty. At present, information uncertainty visualization is non-trivial and requires the user to have both a comprehensive understanding of uncertainty as well as sophistication with visualization tools.

A. Sensemaking/Visualization

Visualization is "the bringing out of meaning in information" [12]. It is performed iteratively and usually as part of the *sensemaking cycle* [4], [23]. The iterative looping is not exclusive to mapping data into visual form; instead, users sometimes return to the data model to gather or transform data. This is particularly true for information uncertainty. For example: uncertainty details can be deemed to be more important later, once the basic model is in place; or the uncertainty details may change as more becomes known about the variables. Therefore, frameworks for information uncertainty visualization should ideally allow the user to go back to make changes with minimal effort.

B. Visualizing Information Uncertainty

Flexibility: Visualization of information uncertainty is different to visualizing other forms of information for two main reasons. Firstly, information uncertainty is associated with a particular unit of information. This means that the uncertainty cannot be freely visualized without regard to its interpretation relative to the information to which it belongs. Secondly, information uncertainty is usually mapped differently to visual elements. For example, uncertainty is commonly mapped to intrinsic properties, such as transparency or color; or by adding a dimension to geometry, such as using a surface where there would otherwise be a line. Therefore, a visualization system for information uncertainty requires the flexibility in mapping uncertainty to visual elements, including intrinsic properties and changed geometry.

Figure 1 demonstrates how information uncertainty is associated with information, but typically mapped differently to visual elements. Four graph visualizations of historical and predicted employment rates in California are shown. The first graph (a) assumes that growth will continue at the average growth rate of the past 15 years and is therefore visualized using traditional means. While the information in graph (a) is modeled as *not* being subject to uncertainty, it requires the unreliable assumption about employment rates to be made. The graph in (b) estimates that the growth will continue at the average rate. The fact that the predictions are estimates is indicated by the line stippling, an intrinsic property of the line. The graph in (c) shows the possible range within the maximum and minimum growth rates experienced in the past 15 years. The uncertainty is indicated by extending the one dimensional line into a two dimensional polygon. The graph in (d) uses a normal distribution centered on the average growth rate. The uncertainty is indicated by both extending the dimensionality of the line as well mapping to the intrinsic property of opacity.

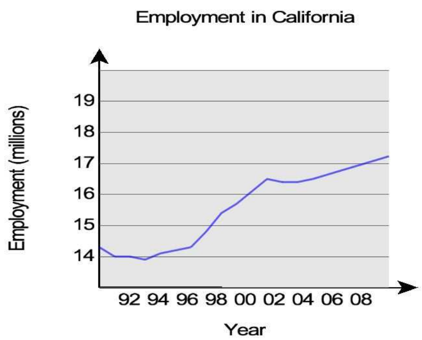
Homogeneous Access: To enable the visual mappings that expose the uncertainty in variables, it is necessary to have access to the associated uncertainty details. However, there are numerous uncertainty modeling techniques that use different methods for encoding the uncertainty. This creates a barrier to visualizing uncertain information because visual mappings that work with one uncertainty modeling technique may not work for another. Such inconsistency creates a strong dependency between visualizations and the data types used in the model, limiting the user's ability to update the model. Therefore, a generalized means for accessing information uncertainty information should be sought to enable a consistent environment information uncertainty visualization.

C. Declaring and Managing Information Uncertainty

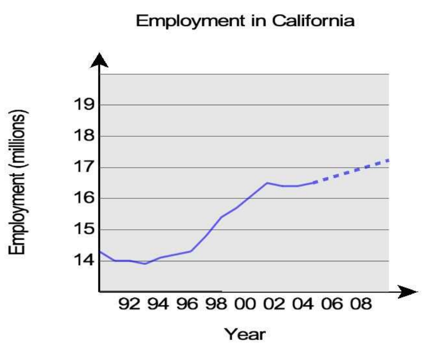
Model Rigidity: The declaration of the information uncertainty should be co-located with the information to which it *relates*, since the two are fundamentally connected. However, this relationship is neglected in most environments, which instead require the user to declare the parameters of the uncertainty separately from the variable. This results in an added layer of complexity and the user is faced with an increasingly intricate data model.

Adding uncertainty information to a data model allows the user to specify a greater level of detail about the model. However, changing the uncertainty modeling technique typically requires the user to reconstruct the affected portion data model, often involving a fundamental change in form. This makes the data model rigid and, as a consequence, the user will typically need to anticipate their use of uncertainty and build their model accordingly. This poses a limitation for uncertainty visualization, limiting the workflow loop where the user updates the model in response to visualization.

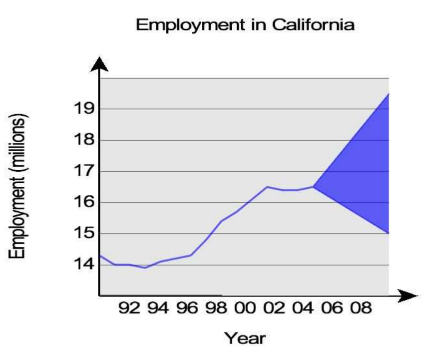
Separation of Parameters: Furthermore, the strongly related parameters of the uncertainty modeling technique are often treated as separate variables. For example, rather than declaring a variable as being modeled using a probability, many environments require separate variables for mean and variance. This is akin to use of the "go-to" directive before the advent of structured programming, because the burden is upon the user to treat these variables as being connected. This separation has two significant ramifications: firstly, it is easier to introduce errors since the environment does not enforce any semantic properties of the uncertainty parameters;



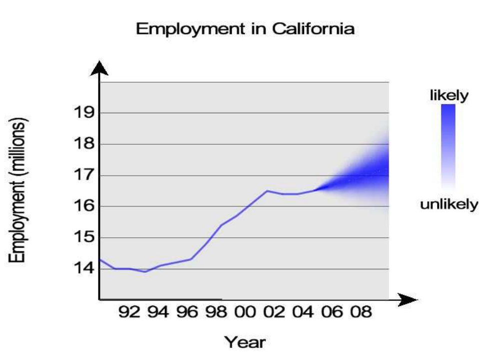
(a)



(b)



(c)



(d)

Figure 1. Visualizations of Employment Numbers in California. Years 2005-2010 are predicted. (a) Assuming Average Growth (b) Indicating Growth is Estimated (c) Possible Growth (d) Likely Growth. (Data Source: California Employment Development Department)

and secondly, the introduced complexity discourages users from using uncertainty modeling techniques, limiting the information available for visualization. Therefore, the uncertainty parameters should be treated as part of a unit and the system should enforce the semantics of the parameters.

Propagation of Uncertainty: Since it is usually up to the user to manage and interpret uncertainty parameters, the use of information uncertainty therefore requires the user to have a mathematical understanding of modeling techniques, not only when declaring the uncertainty, but also for the subsequent propagation of that uncertainty. It would therefore be ideal if the system can track the uncertainty with the variable to which it applies and manage propagation of the uncertainty whenever that variable is used.

IV. Encapsulating Uncertainty in Information

The previous section described a number of uncertainty modeling and management issues that users face when they seek to visualize information uncertainty. This section seeks to overcome these issues through the use of a highly cohesive polymorphic[1] data model that encapsulates the uncertainty within the variable. This approach has three significant advantages: firstly, the uncertainty models become polymorphic, allowing the user to think in terms of variables and not their modeling techniques, thus reducing model rigidity; secondly, it provides structural support for dealing with parameters of the uncertainty as a unit, thereby allowing automatic propagation of uncertainty and avoiding common errors that arise when related parameters are separated; and thirdly, it integrates information uncertainty modeling techniques into a consistent framework, allowing homogeneous access for mapping uncertainty into visual form.

A. Levels of Uncertainty Detail

There are numerous modeling techniques for describing information uncertainty. Interested readers are directed to [14] for mathematical foundations that aim toward a general theory of information. We place these modeling techniques into one of five general categories: *estimate*, where the value is not guaranteed to be the true value; *non-specificity*, where the true value is known to be one of a set of values; *probability*, where the likelihood of the true value is known; *membership*, where

[1] *highly cohesive* refers to inseparability of uncertainty parameters; *polymorphic* refers to the definition covering different data types

Graph	Growth Rate	Known / Assumed
(a)	0.147	it is certainly 0.147
(b)	$0.147, estimated = true$	it is not necessarily 0.147
(c)	$[-0.3, 0.6]$	it is between -0.3 and 0.6
(d)	$\mu = 0.147, \sigma = 0.1$	it is probably 0.147

Table I
PREDICTED GROWTH RATES USED IN FIGURE 1

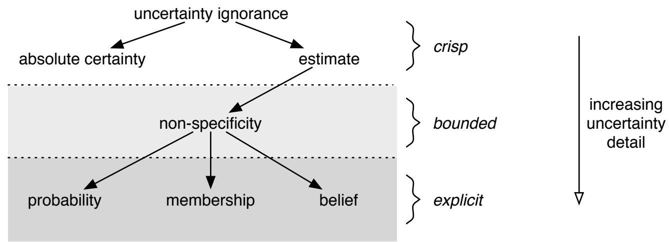


Figure 2. The information uncertainty modeling techniques sorted into three strata

the degree of membership[2] within a group or label is known; and *belief*, where the believability of values is known. Two additional categories are required for completeness: *absolute certainty*, which covers values where there is no uncertainty; and *uncertainty ignorance*, which is the degenerate case where uncertainty is unknown or ignored.

Information uncertainty modeling is a way of *improving fidelity* of the model. By adding information about the uncertainty of a variable, the user is increasing the level of knowledge about that variable. Consider the future employment growth rates used in Figure 1, which are shown in Table I. With each graph, more information is shown about the future employment rates. The value used in graph (a) is ignorant of potential for variance in the predicted growth, implying its value to be certain. In graph (b), it is known that the value is only an estimate, which is additional knowledge that was not available in graph (a). Graph (c) adds further information: it is certain that the value will be within the bounds. Graph (d) adds even more information, the degree of certainty about what the actual growth value will be.

We further group the modeling techniques according to their level of uncertainty detail (see Figure 2): the top tier being the *crisp* strata; the middle tier being the *bounded* strata; and the lower tier being the *explicit* strata. Crisp modeling techniques describe a single value, to the exclusion of alternatives. Estimates belong in the crisp strata, because estimates represent values that are treated as though they were the true value: no information about the uncertainty is known other than its existence. In the bounded strata, the modeling techniques encode the boundaries between what is possible and what is not. For example, the accuracy of a measurement device can be specified as ±10 units. This means that it is certain that the measurement is within ±10 units (assuming the device is working correctly), however, nothing is stated about the degree of certainty of the possible values: it is not stated that +1 is more or less certain than +2. The explicit strata contains modeling techniques that explicitly state the degree of certainty of all candidate values. For example, it is more certain that the predicted employment growth rate in California will be 0.147 than -0.3.

[2]Although standard sets use the term "membership", they do not employ a notion of *partial* membership and are thus part of the *non-specificity* category.

Each strata provides increasing detail about the uncertainty over the previous one. As the user refines their model, they can progress downward along the strata with the addition of more information. Thus, as the model is refined, a variable may begin as an estimate, then be refined to an interval once the extents are known, then be refined further into a probability distribution as the full likelihood becomes known. This refinement is implied by the arrows in Figure 2. The reverse operation is also possible where, by removing information, the variable can be modeled using a technique that is further back up the level of detail tree.

B. Information Uncertainty Conversion and Propagation

To support the concept of refinement, *conversion operators* are necessary to convert from one modeling technique to another. Conversion operators are normally subject to the uncertainty invariance principle [14], which requires that the level of uncertainty remain unchanged after conversion. We use the term *promotion* to refer to conversions that increase the level of uncertainty detail and *demotion* for decreasing the level of detail. The specific implementation of conversion operators will depend on the user's mathematical requirements and there is no single correct choice. For example, a common interpretation when promoting a non-specific uncertainty value to a probability is to assign each possible value an equal probability [13]. However, the user may wish to take an alternative approach and instead create a normal distribution.

For every variable there exists a demotion that produces a crisp value. We term this crisp value the *representative value* for that variable. For example, in the case of probabilities, the representative value can be one of the most likely values, the mean, or any other operator that suits the purpose of the user. The intention of the representative value is to provide a value that the user considers a reasonable estimate. The user will associate this value with the variable when details of the uncertainty are hidden from view.

Once a variable has associated uncertainty information, the uncertainty is propagated through subsequent operations upon that variable. In order for variables to correctly interact, either the user has chosen a mathematical model that defines the semantics of the operation for the combination of their associated uncertainty models (e.g. [14], [18], [28]), or the uncertainties are first converted into compatible forms. Which conversions are performed depend upon the needs of the user. For example, consider the following operation $C = A + B$, where A is an interval and B is a probability. The user might prefer to make minimal assumptions, where B would be demoted to an interval; or the user may wish to maximize uncertainty detail, where A would be promoted to a probability using default rules.

C. A Visualizable Form of Information Uncertainty

There are numerous ways to model information uncertainty, some of which are discrete and others that are continuous. Such a heterogeneous environment can make visualization tools excessively complicated. It would therefore be beneficial to have a generalized form for describing uncertainty that can be used when mapping uncertainty to visual features.

Our approach is inspired by the description of fuzzy sets, using the membership function μ, as a generalized form of crisp sets (e.g. [18]). A fuzzy set is defined using membership function μ: $\delta = \mu(v)$ where δ is the level of membership ranging from 0 (definitely not a member) to 1 (definitely a member) and v is the candidate value. Thus the candidate value 28 is half in the fuzzy set $long$ if $long(28) = 0.5$. This method of definition can be applied to crisp sets, for example the set A can be defined by:

$$\delta = \begin{cases} 1 & v \in A \\ 0 & otherwise \end{cases}$$

We expand this reasoning to other uncertainty modeling types. However, we choose to replace the symbol for membership to avoid confusion. Thus the visualization accessible form of information uncertainty is a function:

$$\delta = f(v)$$

where δ is the *degree of certainty* ranging from 0 to 1, v is the candidate value, and $f()$ is the degree of certainty function. Traditional numbers can be considered to be a special type of uncertainty modeling technique: the technique specifying total certainty. The constant c is described by the following uncertainty function:

$$\delta = \begin{cases} 1 & v = 2 \\ 0 & otherwise \end{cases}$$

We choose $\delta = 0$ to indicate impossibility. The true meaning of δ varies with the type of uncertainty being modeled. For non-specificity types, δ is either 0 (not possible) or 1 (possible). For membership methods, δ ranges from 0 (definitely not a member) to 1 (definitely a member). For probabilities and belief, δ ranges from 0 (impossible) to 1 (certainly).

Some visualizations are intended to compare the values of $\delta$with each other. Depending on the uncertainty of the variable in question, the range of δ can vary. In these circumstances it is desirable to *normalize* δ such that $min(\delta) = 0$ and $max(\delta) = 1$.

The visualization system can focus on providing means for mapping a range of $[0..1]$ to visual elements, rather than providing methods specific to each type of modeling technique. The next section describes such an information uncertainty visualization system based on a spreadsheet paradigm. $f(v)$ is provided as a special operator in the spreadsheet formula language, giving read only access to the uncertainty.

V. Spreadsheet System for Information Uncertainty Visualization

This section describes an integrated visualization and modeling system design that uses a spreadsheet paradigm. This integrates the modeling and visualization tasks, allowing a tight feedback loop between visual inspection and data model building.

The relationship between spreadsheets and other approaches can be illustrated using a formal definition for spreadsheets that consists of four components [9]: the *schema*, a definition of the spreadsheet logic; the *data*, which are the instance values for this spreadsheet; the *editorial*, which consists of headings, borders, etc.; and *binding*, which is the mapping of the content to the tabular structure of cells. It is the *binding* property that is responsible for the tabular layout of a spreadsheet. The *binding* could be replaced by a mapping of data to member variables and the *editorial* could be converted to comments, thereby allowing implementation of the same functionality using a traditional programming language.

The spreadsheet paradigm allows a great amount of freedom for users to organize their information. The freedom to quickly perform experimental calculations that do not have to be integrated with the rest of the model is of benefit to users. However, the drawback of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

this freedom is that spreadsheets can be error prone. The fact that spreadsheets are so widespread and yet capable of errors has motivated much research into spreadsheet testing methods [2], [17], particularly where spreadsheets are used for financial decisions. Further discussion of error reduction strategies is beyond the scope of this paper.

The terminology used in this section is as follows. The *spreadsheet matrix* is made up of *sheets*. A *sheet* is a heterogeneous sparse two-dimensional grid of *cells*. Sheets are also theoretically infinite, but practically constrained due to resource limitations. Their heterogeneity refers to the ability to have cells of different types within the same sheet. A *cell* is an addressable location that contains a unit of information. We use the term *uncertainty spreadsheet* to refer to any spreadsheet that includes information uncertainty.

A. Motivation for Our Approach

Our approach integrates both the visualization and modeling tools into a single system. Spreadsheets are ideal for this because they are interruptible, widely understood, and in a constantly running state. The interruptible characteristic allows the user to move to another location in the spreadsheet to experiment, without interfering with their main task. They are widely understood by users because spreadsheet use is ubiquitous, especially in the financial modeling field. Finally, unlike scripts that must be run before they produce results, a spreadsheet is constantly in an up-to-date state, allowing it to be easily interrogated and refined.

To support the needs of visualizing information uncertainty, the system must provide flexibility for mapping uncertainty information to visual elements. The formula construct of spreadsheets is both expressive and powerful, and is easily extended to enable access to uncertainty information. Thus, the spreadsheet formula presents an ideal mapping method.

There exist numerous information uncertainty modeling techniques and multiple mathematical models for the propagation of these uncertainties. Therefore we use a plug-in based architecture to allow new uncertainty data types and propagation models to be added to the system. Additionally, new visual elements can be provided through the plug-in system, since new display techniques continue to be developed.

B. Design of the Spreadsheet Software

The goal of this software is to directly support uncertainty information within spreadsheet cells, in a managed and extensible way. The process for construction of a spreadsheet using the software is described in later in Section V-D.

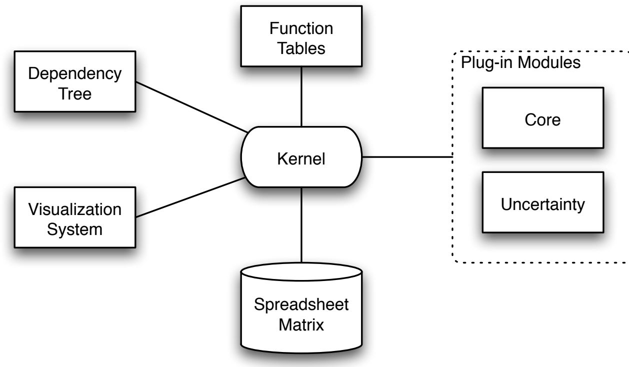


Figure 3. Diagram of the System Components

Figure 3 shows the main components of the spreadsheet software, which is designed to be modular. The *kernel* provides the interface between the components. The *spreadsheet matrix* is a collection of sheets and cells that the user interacts with, containing all the information that is required for persistent storage. The *dependency tree* is used to update appropriate cells when changes are made. The *function tables* provide an index to functions that are invoked in formulae, and are necessary to enable uncertainty propagation and conversion. The *visualization system* is a scenegraph-based graphics display system that maps data stored in the spreadsheet matrix to visual elements. The *core* plug-in adds basic spreadsheet functionality, including string, number, and formula cell types; the *uncertainty* plug-in extends the traditional types to include estimates, intervals, and normal probabilities.

The user navigates the spreadsheet matrix and enters input to be placed in the cells. The input takes the form of a string, for which there is an edit field at the top of the screen. Once the input has been submitted by the user, each plug-in is queried to determine which should handle the string. If none match then it is a string type. The string is parsed by the appropriate plug-in, which generates and returns a cell object. For example, the string "10+-2" will be processed by the interval component of the uncertainty plug-in to produce an interval of 10 ± 2. The resulting cell object is inserted into the current sheet at the current cursor location.

Formulae are identified by a leading equals symbol. The remainder of the string is parsed and converted into a sequence of function calls, with infix operators (e.g. "+") being converted into function names (e.g. "add"). The function tables are used to invoke the appropriate function handler for the parameter types. Plug-ins register function handlers with a {*function name, parameter types*} signature. Propagation of uncertainty is managed by using multiple handlers for the same function name, but with different parameter type combinations. For example, interval addition can be used when both parameters to the add function are intervals. The mathematical model for managing the propagation of uncertainty is user selectable, as different domains may have different propagation requirements: when more than one function handler is defined for the same signature, the user can choose which one is active.

To handle operations between different types of uncertainty, an appropriate function handler must be defined for the parameter types. Our prototype system uses function handlers that promote parameters to the higher level of detail parameter type. For example, addition of estimates and intervals result in intervals.

The use of formulae creates functional relations between cells and it is from these relations that the dependency tree is built. The dependency tree lists the cells that directly depend upon a particular cell. There cannot be any circular references as this would create race conditions. When a user completes updating an existing cell, the system recalculates any affected cells. Affected cells are determined by walking the dependency tree, starting with the current node. If the current node is not a member of the dependency tree then no other cells need updating.

Typical spreadsheet languages contains basic operators (such as addition, subtraction, multiplication, and division) and usually a wealth of commonly used functions (such as statistical and financial functions). To support information uncertainty, the language needs to be extended to allow access to the underlying uncertainty information. Two categories of new functions are added. The first category contains the *conversion operators*, which convert from one type of uncertainty model to another. The second category involves interrogation of the uncertainty details, which is intended to be used for mapping to visual elements. This second category makes use of the visualizable uncertainty form given in Section IV-C. The prototype system provides five such interrogation functions, listed in Table II.

The user will sometimes think of the cell contents as the representative value. Therefore, to avoid visual clutter, the system has an option that hides the uncertainty details and displays the representative value in the cell. Figure 4 shows a screenshot of the prototype system where uncertainty hiding has been enabled. The currently highlighted cell shows the value 171.51, whereas the cell contents is actually “normpdf(175.51,5)”. This behavior parallels formulae, which display the result of the calculation in the cell rather than the formula itself. The prototype automatically shades cells to help identify those that are uncertain. The color scheme was chosen arbitrarily.

Function	Returns
isCertain(x)	*True* if the cell x is completely certain.
hasUncertainty(x)	*True* if the cell x has associated uncertainty information.
Lower(x)	The lower bounds of the cell x.
Upper(x)	The upper bounds of the cell x.
Certainty(x, y)	The degree of certainty that the cell x is going to be y, where y is a crisp value. y is a literal constant or the address of a reference to another cell. In the unusual case that y is uncertain, the representative value is used.

Table II
PROTOTYPE UNCERTAINTY INTERROGATION FUNCTIONS

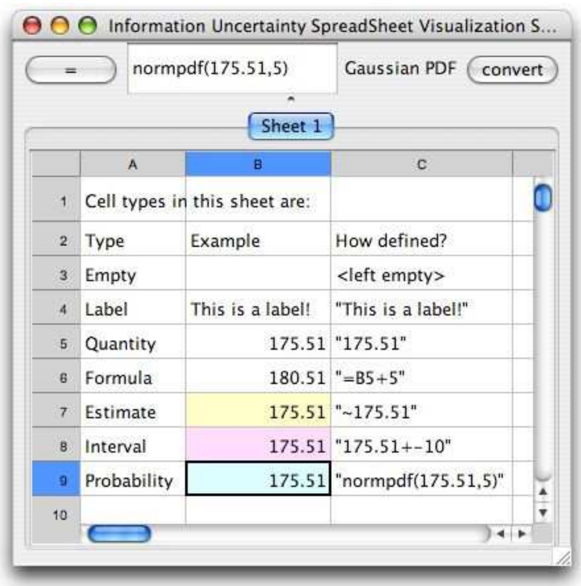


Figure 4. Screenshot of the Prototype with Uncertainty Hidden

The visualization system is implemented as a specialized sheet, called a *visualization sheet*. The layout of the visualization sheet matches the scene graph structure, with every non-empty row of the visualization sheet representing a node in the graph. The starting column indicates the position of the node in the hierarchy, where a row that begins in column n will be a child node of its nearest preceding row that begins in columns $1..n-1$, or the root node if none can be found. Thus, all rows beginning with the column “A” will be children of the root node, while rows beginning in column “B” will be children of a preceding row beginning in column “A”, etc. The first non-empty column contains a string key that determines the node type. The subsequent columns

	A	B	C	D	E	F
1	Title	Effect of Interest Rate Movements on NPV				
2	2Daxes					
3		Yaxis	Net Present Value	0.25	-100k	0k
4		Xaxis	Year of Sale	0		2
5	Scale	0.1	0.003	1		
6		Translate	0	100	0	
7			Color	0	0	255
8				AreaLine	0	0

Figure 5. Visualization Sheet for the Graph in Figure 8

contain the parameters for that node. The nodes themselves are implemented by classes derived from the scene graph *group_node*. It is the responsibility of the node object to perform type checking on parameters.

Figure 5 shows a visualization sheet with three children of the root node: a Title node, which displays title text; a 2Daxes node, which generates a rectangular grid; and a Scale node, which adds a scaling to the transformation matrix of its children. The 2Daxes node has two children, which specify the labels for each axis. The Scale node has a Translate child, and together they position the data correctly over the 2Daxes object. The Color node specifies that its child should be drawn in blue. The AreaLine node takes a sequence of lower and upper y-values and produces the polygon representing the data. All parameter fields can either contain an immediate value or a formula. For example, the cell E8 in Figure 5 contains the formula "=Lower(Sheet 1!F8)", indicating that this cell should contain the lower bounds of the cell "F8" in the sheet "Sheet 1".

Uncertainty is mapped to visual elements using the functions in Table II. For example, the Certainty() function can be used to map the degree of certainty to opacity for nodes supporting an opacity parameter, as was done in Figure 1 (d).

C. Capabilities of the Spreadsheet Software

The default uncertainty plug-in implements several information uncertainty modeling techniques. This allows users to construct a model using information uncertainty models as native types. The parameters of the uncertainty are inherent in the cell, providing structural and semantic support for the uncertainty modeling technique, thereby avoiding potential errors that can arise when parameters are separated.

The system provides an ability to choose the propagation model from options supplied by plug-ins. The user is now free to iteratively build uncertainty into a model: first, a rough crisp data model is produced as a proof-of-concept; then, the user refines the data model by promoting variables to add uncertainty detail. Each variable with uncertainty is treated as a unit and propagation of the uncertainty is handled automatically using the chosen propagation model.

The modular and extensible plug-in based architecture allows new cell types, propagation models, and scene graph nodes to be defined by plug-ins. New cell types can be used to cover additional uncertainty modeling techniques; new propagation models are able to handle propagation of uncertainty and combinations of various modeling techniques; and new scene graph nodes allow future display techniques to be supported.

The use of a visualization sheet keeps the interface consistent and brings the power of formulae to the visual mapping process. Through the combinations of several visual elements, sophisticated visualizations can be constructed by the user. This provides the flexibility to perform traditional visualization tasks as well as supporting the sometimes unusual needs of information uncertainty visualization.

D. Process for Constructing an Uncertainty Spreadsheet

We use an incremental process to building uncertainty spreadsheets. It is typically most convenient to begin construction of a model at the high level, where the focus is on logic, before proceeding to add details. The use of uncertainty modeling increases detail and is therefore typically most conveniently added later, once the basic structure of the model is in place. This is particularly true of information uncertainty.

Figure 6 shows the process for building an uncertainty spreadsheet. In the first step an initial spreadsheet is built, typically using *uncertainty ignorance* datatypes. This step is similar to building a traditional spreadsheet, and includes the spreadsheet structure, variables, and formulae. Next, the spreadsheet is iteratively refined in three ways: firstly, uncertainty detail is added to (or removed from) variables; secondly, visualizations in the model are added, altered, or removed; and thirdly, the model can be refined in the traditional sense, such as changing formulae or adding variables.

The task labeled *refine uncertainty details* consists of two types of activities: adding / removing uncertainty detail, and changing the mathematical model for propagation. There are three main steps to add or remove uncertainty details: firstly, the appropriate variable is identified, e.g. a variable whose uncertainty is currently ignored; secondly, its details are changed (promoted / demoted); and thirdly those changes are evaluated, returning to the second step if found to be inadequate.

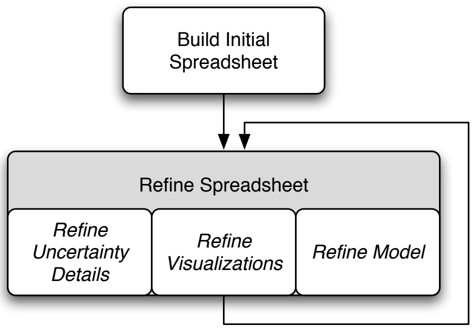


Figure 6. Process for Constructing an Uncertainty Spreadsheet

E. Comparison to Traditional Spreadsheets

To incorporate uncertainty modeling in a traditional spreadsheet requires three major changes. Firstly, the uncertainty details must be recorded in the spreadsheet somewhere, resulting in additional cells being used. The addition of new cells changes the layout of the spreadsheet and increases the amount of information that the user faces. Furthermore, the number of cells that are added depend on the number of parameters required by the uncertainty data type. Secondly, formulae need to be changed to incorporate the propagation of uncertainty details. These formulae become harder to understand, because the uncertainty information handling obscures the fundamental operation. The uncertainty information propagation must also be carried forward to all downstream formulae, which can be many. Thirdly, any graphs or visualizations should be updated to include uncertainty information as appropriate.

There are four limitations to traditional spreadsheets for incorporating uncertainty. Firstly, the user is required to be intimately aware of the uncertainty modeling *technique*, including rules for its propagation, before they can incorporate it in their model. Secondly, it is prohibitive to change the level of uncertainty information after the initial design. Adding uncertainty information after the model is already in place becomes an arduous task that is error prone. Should more information come to light, for which another uncertainty modeling technique is more appropriate, then all affected parts of the model have to be manually rebuilt. Thirdly, it is also prohibitive to change the mathematical propagation model, since changing propagation rules requires all formulae to be rewritten. Fourthly, there are currently few built-in visualization techniques for information uncertainty. The visualization techniques that are supplied target specific uncertainty modeling types (e.g. intervals). To create sophisticated visualizations requires the user to export their data to a more advanced visualization system.

Our system overcomes these four limitations. In contrast to adding new cells to the spreadsheet, our approach is to store the uncertainty information in the same cell. The immediate advantages of this approach are that the spreadsheet does not change in layout and the number of cells do not increase, irrespective of the type or volume of uncertainty information it contains. Furthermore, the system is aware of this uncertainty information and has a mechanism for resolving appropriate propagation operations in formulae, meaning that formulae do not change either. Thus, it is a local change to a *single* cell to add, change, or remove uncertainty for a variable. Exceptions only occur where the user's chosen mathematical model prohibits particular operations or combinations, which is no different to any traditional approach. The system resolves operations using a table of operations that the user can control at a global level. Therefore, should an alternative mathematical model be needed, no change to the actual spreadsheet contents is required. Our system uses a flexible visualization sheet that allows sophisticated visualizations to be explored. The advantage is that any changes to the spreadsheet are immediately reflected in the visualizations.

We interviewed two financial analysis experts, three regular spreadsheet users, and one general computer user. Participants were shown the prototype system applied to a sensitivity analysis scenario. All respondents found the approach to be clear and intuitive. Of the five that used spreadsheets regularly, all indicated that they would like to use uncertainty spreadsheets as a tool. The most common reasons for wanting to use an uncertainty spreadsheet were time savings and immediate visual feedback.

VI. Case Study

This section illustrates the advantages of using our architecture over a traditional spreadsheet when used in a case study. The problem to be explored is understanding and visualizing the profitability of a prospective investment property. Acquiring property for investment and rental income is a common prospect for many who may not have a background in finance. However, there are many estimations and subtle interactions between variables that can have significant effects on the profit outcomes. Furthermore, many of these interactions are poorly understood or difficult to define, even for experts.

The decision to acquire an investment property is based on profitability of the investment. Therefore, the output of the model is a Net Present Value (NPV) calculation that gives a comparison of the profitability of buying a property using a deposit against investing that same deposit into a fixed interest vehicle. A positive

NPV indicates that the property investment is more attractive.

The NPV calculation is as follows:

$$NPV = \sum_{n=1}^{t} \frac{CashFlow(n)}{(1+i_n)^n}$$

where t is the number of years the property is held; i_n is the after tax interest rate in year n; and $CashFlow$ is given by

$$CashFlow = r - p - o + x - C^I + C^O - u$$

where r is rental income; p is the loan payment for the current year; o is the ongoing expenses; x is the tax refund due to investment; C^I is the deposit paid on purchase; C^O is the deposit + net profit on sale; and u are upfront costs.

Building the Initial Spreadsheet: If uncertainty is ignored, then the common approach to this problem is to create a tabular spreadsheet: each column contains a variable and each iteration of n adds another row. A summary page is created where input variables, such as increases in salary, can be placed in an accessible location. The user is able to change input details and observe their effects over a number of years, usually with the aid of graphs.

Most of the variables in this model are subject to uncertainty. For example: rental income becomes progressively less certain the farther into the future it is predicted; loan repayments are similarly uncertain since they are dependent on a variable interest rate; the tax refund is uncertain because it depends upon taxation law, employment status, and promotions, all of which can change unexpectedly; and the net profit on sale is always subject to uncertainty.

Spreadsheet Layout is Unchanged: Adding uncertainty details using our software does not add new cells or change the spreadsheet layout. Figure 7 illustrates this using the annual salary increase projected over 20 years. There are four variables: *salary growth*, which is given by the user; *salary*; *tax*; and *net income*. Uncertainty information propagates from *salary increase* to *salary* to *tax*. The user wishes to model the *salary increase* as an interval of 71 (6 to 8). Figure 7 (a) shows the original spreadsheet model prior to modeling an interval. Figure 7 (b) presents a solution using a traditional spreadsheet, which requires six columns to represent three variables. Each column had to be manually added and the formula for *tax* and *salary* had to be updated to reflect this change. Figure 7 (c) shows our prototype system with uncertainty hiding switched on. The *salary growth* field was promoted to an interval (71) and no other change was made. In this view the updated model closely reflects the original[3]. Figure 7 (d) is the same as Figure 7 (c) with uncertainty hiding switched off.

Formulae are Unchanged: The shaded cells indicate that they contain an interval, thus it can be seen from Figure 7 (b) that the uncertainty is propagated automatically to both *salary* and *tax*. The formulae for these cells, however, are unchanged. It is noteworthy that while the figure shows the representative value in each cell, the user can always toggle the viewing option to show the uncertainty details instead of the representative value. The traditional approach not only changes layout, but the formulae had to be repeated to calculate both the low and high rates.

To achieve the same effect using a traditional spreadsheet requires more effort. Firstly, *each* affected variable must be expanded to two cells, namely the upper and lower bounds. This typically involves adding an additional column for each variable that is calculated over multiple years. Secondly, the propagation of the uncertainty information must be manually managed by adding the appropriate formulae.

Visualization can be Abstracted from Uncertainty Type: Using the traditional spreadsheet limited the graphs to those that the program provided, of which two could be used to indicate the intervals. The first was a graph that used error bars, while the second was to overlay the maximum and minimum lines on the same graph as two different data points. However, these traditional graphs only work with interval data. In contrast, the graph in Figure 8 will work with the other datatypes.

The graph in Figure 8 was generated using three elements in a visualization sheet: a title text object, a 2D axes object, and a polygon. The 2D axes object takes as parameters the label and range for the vertical and horizontal axes. The polygon requires a color specified in the first four cells, followed by a series of alternating x and y coordinates. The y coordinate is given by firstly the lower bounds of the variable, then the upper bounds, using formulae of the form "=Lower(*cellref*)", where *cellref* is a reference to a cell containing NPV for the appropriate year. These functions are defined for all numerical types. For example, the *Upper()* and *Lower()* functions return the same value when that value is certain, resulting in a line graph.

Changing Uncertainty Models is Easy: The user can choose to use the modeling technique that is appropriate for the variable, with little regard for how the rest of the

[3]Note that the number shown represents the halfway value between the upper and lower limits. The upper limit grows more rapidly than the lower limit, thus the mean value for $[6, 8]\%$growth will not match 7% growth.

salary growth	7%		
year	salary	tax	net income
1	60000	10200	49800
2	64200	10914	53286
3	68694	11677.98	57016.02
4	73502.58	12495.44	61007.14

(a)

salary rate lo	6%	salary rate hi	8%			
year	salary (lo)	salary (hi)	tax (lo)	tax (hi)	net (lo)	net (hi)
1	60000	60000	10200	10200	49800	49800
2	63600	64800	10812	11016	52788	53784
3	67416	69984	11460.72	11897.28	55955.28	58086.72
4	71460.96	75582.72	12148.36	12849.06	59312.60	62733.66

(b)

salary growth	7%		
year	salary	tax	net income
1	60000	10200	49800
2	64200	10914	53286
3	68700	11679	57021
4	73521.84	12498.71	61023.13

(c)

salary growth	[6, 8]%		
year	salary	tax	net income
1	60000	10200	49800
2	[63600, 64800]	[10812, 11016]	[52584, 53988]
3	[67416, 69984]	[11460.72, 11897.28]	[55518.72, 58523.28]
4	[71460.96, 75582.72]	[12148.36, 12849.06]	[58611.90, 63434.36]

(d)

Figure 7. Interval Modeling Example: (a) Original Model (b) Traditional Spreadsheet (c) Prototype System Uncertainty Hidden (d) Prototype System Uncertainty Shown

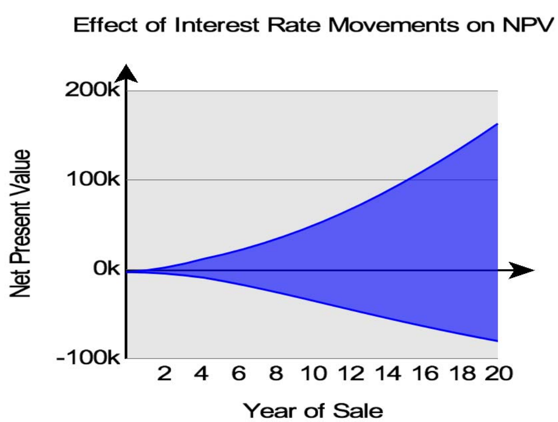


Figure 8. Using An Interval (0.5) for *Annual Change in Interest Rates* Propagates the Uncertainty to NPV

data is modeled. The interest rates are unlikely to change maximally and more likely to stay even. Therefore, it is desirable to model the changes in interest rates as a probability distribution. To model this we choose a Gaussian distribution centered on no change, which is a reasonable assumption in an unknown economic climate. Using our system, the user simply promotes the *annual change in interest rates* to a Gaussian probability distribution. As with intervals, the uncertainty will be automatically propagated through to NPV. If multiple uncertain variables interact, our system automatically manages their combined uncertainty information.

In contrast, the traditional spreadsheet requires more work to achieve the same effect. Each variable now requires two cells of a different sort: the first cell to contain the mean, and the second cell for the variance. Every formula that was previously written to handle the intervals must now be changed to handle normal distributions, which requires both mathematical competency as well as care to avoid introducing errors. If multiple uncertain variables interact, then the formulae must be painstakingly integrated.

Flexibility for Sophisticated Visualization: The ability to use multiple visual elements, and map data to those elements using formulae, gives the user the flexibility to create sophisticated visualizations such as Figure 9. This figure shows the most likely NPV against the *year of sale* and the *property value appreciation*. The volume is actually composed by layering several surfaces, with the certainty of NPV mapped to opacity. The color of the surface is red if the NPV is negative, green otherwise. A wire-frame outline of the extents of the thresholded NPV volume was added to provide context.

The information shown in Figure 9 could not be produced using current spreadsheets. Firstly, flexibility of visualization was required to stack multiple surfaces with varying color and translucency together with a wireframe outline into a single 3D space. Secondly, the calculations that underpin the uncertainty propagation are complicated enough to be prohibitive.

Reaction of Domain Experts: The case study was presented to two financial analysis experts for feedback. Their reactions were positive: results were intuitive to follow, the methods would save them time, and the visualizations were easy enough to follow and provided useful information.

VII. Conclusion and Future Work

The spreadsheet software presented here allows the user to easily add (and remove) uncertainty details to

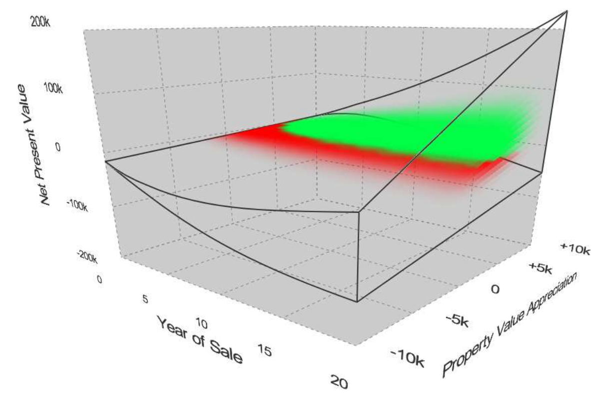


Figure 9. Volumetric Representation of the Most Likely Effect Interest Rate Changes Will Have on NPV.

variables as needed. This makes it easy for the user to find the appropriate uncertainty modeling technique for the problem at hand. The uncertainty information is automatically propagated to related parts in the model and is readily available for mapping to visual elements. This quickly fills the model with accessible uncertainty information, making it easier to visualize. The visualizations are built on a low level, using formulae and cell references. This provides the expressive power of spreadsheet formulae as well as the benefits of online visualization. The user can create unusual and sophisticated visualizations with immediate feedback whenever a change is made to the spreadsheet. All these factors overcome barriers to the use of information uncertainty visualization by making it easier, less error prone, and more accessible.

The extensible architecture allows new cell types, operations, and visual elements to be defined by plugins. In this paper we have considered several modeling techniques, but further work can easily add more. For example, classic sets are non-specific types that could be evaluated in an expert systems context. While new data types can be added, parsing formulae to recognize arbitrary constants in an extensible way is non-trivial (consider "= 5 +- 2 - 2", which should evaluate to 32) . The prototype system does not support unusual constants to be declared inside a formula. Future work could explore use of a dynamic parser. One way around the problem is to force all constants to be defined by functions, for example "*interval*(5, 2) - 2".

The visualization sheet keeps the interface consistent and brings the power of formulae to the visual mapping process. The user is given flexibility to create the visualizations that they desire. While this is ideal for information uncertainty visualization research, it is perhaps too low level for typical users of spreadsheets. It would be worth investigating the incorporation of a higher level interface, such as the multi-agent framework.

Cells are able to display two states: either the representative value, or the full description. Spreadsheets appeal to users because they can see across multiple cells, therefore it is worth exploring further ways of representing the uncertainty inside cells. For example, other views might show various levels of detail. Another option would be to show graphical representations. There has been extensive work on spreadsheets that incorporate graphical cells and these ideas can be explored in an uncertainty context.

The color scheme used to shade uncertain cells in the prototype was chosen arbitrarily, but this could be explored further. For example, the color might be chosen to represent the degree of certainty in the cell and assign a more saturated color to those that are more uncertain. Such a scheme could aid the user to quickly distinguish areas of high uncertainty.

While the system presented here uses a spreadsheet paradigm, many of the concepts can be applied to other paradigms, such as data flow networks or script based languages. Typical mathematical modeling languages, such as Matlab, currently require separate variables to hold the mean and variance of a Gaussian distribution. Consequently, it is not a simple task to promote a variable to a different uncertainty type. Future work could investigate a similar approach for these environments.

References

[1] L. Bavoil, S. P. Callahan, P. J. Crossno, J. Freire, C. E. Scheidegger, C. T. Silva, and H. T. Vo. Vistrails: enabling interactive multiple-view visualizations. In *Visualization, 2005. VIS 05. IEEE*, pages 135–142, 2005.

[2] Polly S. Brown and John D. Gould. An experimental study of people creating spreadsheets. *ACM Transactions on Office Information Systems*, 5(3):258–272, July 1987.

[3] Steven P. Callahan, Juliana Freire, Emanuele Santos, Carlos E. Scheidegger, Claudio T. Silva, and Huy T. Vo. Vistrails: visualization meets data management. In *SIGMOD '06: Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, pages 745–747, New York, NY, USA, 2006. ACM Press.

[4] Ed H. Chi and Stuart K. Card. Sensemaking of evolving web sites using visualization spreadsheets. In *INFOVIS '99: Proceedings of the 1999 IEEE Symposium on Information Visualization*, page 18, Washington, DC, USA, 1999. IEEE Computer Society.

[5] S. Djurcilov, K. Kim, P. Lermusiaux, and A. Pang. Visualizing scalar volumetric data with uncertainty. *Elsevier Computers & Graphics*, 26:239–248, 2002.

[6] L. Hall and M. Berthold. Fuzzy parallel coordinates. In *IEEE Fuzzy Information Processing Society*, pages 74–78, 2000.

[7] Ed Huai hsin Chi, Joseph Konstan, Phillip Barry, and John Riedl. A spreadsheet approach to information visualization. In *UIST '97: Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*, pages 79–80, New York, NY, USA, 1997. ACM Press.

[8] Ed Huai hsin Chi, John Riedl, Phillip Barry, and Joseph Konstan. Principles for information visualization spreadsheets. *IEEE Computer Graphics and Applications*, 18(4):30–38, 1998.

[9] Tomas Isakowitz, Shimon Schocken, and Jr. Henry C. Lucas. Toward a logical/physical theory of spreadsheet modeling. *ACM Trans. Inf. Syst.*, 13(1):1–37, 1995.

[10] T. J. Jankun-Kelly, K.-L. Ma, and M. Gertz. A model for the visualization exploration process. In *IEEE Visualization*, pages 323–330, 2002.

[11] C.R. Johnson and A.R. Sanderson. A next step: Visualizing errors and uncertainty. *IEEE Computer Graphics and Applications*, 23(5):6–10, September/October 2003.

[12] P. Keller and M. Keller. *Visual Cues*. IEEE Press, 1992.

[13] George J. Klir. The many faces of uncertainty. In B.M. Ayyub and M.M. Gupta, editors, *Uncertainty Modelling and Analysis: Theory and Applications*, pages 3–19. Elsevier Science B.V., 1994.

[14] George J. Klir. *Uncertainty and Information: Foundations of Generalized Information Theory*. Wiley-Interscience, 2005.

[15] Marc Levoy. Spreadsheets for images. In *SIGGRAPH '94: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, pages 139–146, New York, NY, USA, 1994. ACM Press.

[16] Alan M. MacEachren, Anthony Robinson, Susan Hopper, Steven Gardner, Robert Murray, Mark Gahegan, and Elisabeth Hetzle. Visualizing geospatial information uncertainty: What we know and what we need to know. *Cartography and Geographic Information Science*, 32(3):139–160, July 2005.

[17] II Marc Fisher, Gregg Rothermel, Darren Brown, Mingming Cao, Curtis Cook, and Margaret Burnett. Integrating automated test generation into the WYSIWYT spreadsheet testing methodology. *ACM Trans. Softw. Eng. Methodol.*, 15(2):150–194, 2006.

[18] J. M. Mendel. *Uncertain Rule-Based Fuzzy Logic Systems*. Prentice Hall PTR, 2001.

[19] Alex T. Pang, Craig M. Wittenbrink, and Suresh K. Lodha. Approaches to uncertainty visualization. *The Visual Computer*, 13:370–390, 1997.

[20] Binh Pham and Ross Brown. Multi-agent approach for visualisation of fuzzy systems. In *International Conference on Computational Science*. Lecture Notes in Computer Science, June 2-4 2003.

[21] Kurt W. Piersol. Object-oriented spreadsheets: the analytic spreadsheet package. In *OOPLSA '86: Conference Proceedings on Object-oriented Programming Systems, Languages and Applications*, pages 385–390, New York, NY, USA, 1986. ACM Press.

[22] Ramana Rao and Stuart K. Card. The table lens: merging graphical and symbolic representations in an interactive focus + context visualization for tabular information. In *CHI '94: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 318–322, New York, NY, USA, 1994. ACM Press.

[23] Daniel M. Russell, Mark J. Stefik, Peter Pirolli, and Stuart K. Card. The cost structure of sensemaking. In *CHI '93: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 269–276, New York, NY, USA, 1993. ACM Press.

[24] Will Schroeder, Ken Martin, and Bill Lorensen. *The Visualization Toolkit, Third Edition*. Kitware Inc., 2004.

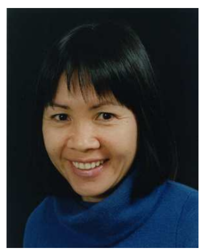
[25] Edward R. Tufte. *Envisioning Information*. Graphics Press, May 1990.

[26] Jarke J. van Wijk and Robert van Liere. Hyperslice: visualization of scalar functions of many variables. In *VIS '93: Proceedings of the 4th Conference on Visualization '93*, pages 119–125, 1993.

[27] A. Varshney and A. Kaufman. Finesse: a financial information spreadsheet. In *INFOVIS '96: Proceedings of the 1996 IEEE Symposium on Information Visualization (INFOVIS '96)*, page 70, Washington, DC, USA, 1996. IEEE Computer Society.

[28] Y.Y. Yao. Interval based uncertain reasoning. In *IEEE Fuzzy Information Processing Society*, pages 363–367, 2000.

Alexander Streit is a PhD student with the Faculty of Information Technology at the Queensland University of Technology in Brisbane, Australia. His research focus is information uncertainty modeling and visualization. His prior research work includes computer graphics algorithms, information visualization, virtual reality, and advanced display systems. He expects to receive his PhD in late 2007.



Binh Pham is currently a Professor in the Faculty of Information Technology at the Queensland University of Technology, Brisbane, Australia, after holding the position of Director of Research for 7 years from 2000. Prior to this, she held the IBM Foundation Chair in Information Technology at the University of Ballarat from 1995-99, and was an Associate Professor in the School of Computing & Information Technology at Griffith University from 1993-1995. She was the Founding Director of the Victorian Centre for Image Processing & Analysis (CIPAG) at Monash University from 1991-1993. Her research interests include computer graphics, multimedia, image analysis, intelligent systems, spatial and temporal data mining, and their application in diverse domains.

Ross Brown works as a Senior Lecturer in the School of Software Engineering and Data Communications at Queensland University of Technology, Brisbane. He also chairs the Visual and Media Computing Research Group within the Faculty. His teaching and research interests are in computer games, computer graphics, visualisation and related disciplines. In particular, he has recently focussed on research into intelligent manufacturing tools that provide support for the automation and augmentation of workflow practices for the video game industry. He also teaches shader rendering and games production as part of the new QUT Computer Games and Interactive Entertainment course.