A Data Model for Scientific Visualization with Provisions for Regular and Irregular Grids

R. B. Haber
Department of Theoretical & Applied Mechanics
Center for Supercomputing Research & Development
University of Illinois at Urbana-Champaign
Urbana, IL 61801

B. Lucas and N. Collins
Thomas J. Watson Research Center
I.B.M. Corporation
Hawthorne, NY 10532

Abstract

This paper presents a mathematical data model for scientific visualization based on the mathematics of fiber bundles. The findings of previous works are extended to the case of piecewise field representations (associated with grid-based data representations), and a general mathematical model for piecewise representations of fields on irregular grids is presented. A discussion of the various types of regularity that can be found in computational grids and techniques for compact field representation based on each form of regularity are presented. These techniques can be combined to obtain efficient methods for representing fields on grids with various regular or partially regular structures.

1 Introduction

An effective data model is essential to the design and development of scientific software systems. For example, object-oriented programming techniques depend on a hierarchical abstraction (mathematical model) of the data and the functions performed on that data. A shared abstraction of the scientific information is essential when a software system is developed by a team of programmers or when a single system is intended for interdisciplinary use. Thus, the data model is a primary consideration in the design of any scientific software system, whether it be an interdisciplinary tool or a special-purpose application.

There has been considerable interest of late in developing data formats to allow scientists to share and transport data sets freely. This was the topic of a recent ACM SIGGRAPH workshop [1]. CDF[2], NetCDF[3] and the NCSA HDF[4] are examples of data formats designed for scientific applications that are currently in use. These formats and their associated software systems allow scientists to port data from one host system to another and from one application to another. In addition, these software systems include data models and associated data structures that allow scientists to describe certain objects, such as regular grids with data. While the underlying data formats (conventions for handling the tasks of storing and communicating arrays, strings, etc. between applications, hosts and mass storage systems) are reasonably robust, the data models in previous systems lack the generality needed to support many scientific and engineering applications.

In this paper, we focus on the problem of representing scientific data on irregular, regular and partially regular grids, and indicate the several distinct forms of regularity that can appear in scientific data sets. The emphasis is on the effective representation of field data - as might arise in a variety of observational, experimental and computational science and engineering applications. Here we present only the conceptual, mathematical framework of the data model; details of the data structures and the data formats needed to implement the data model will be covered in later papers.

Butler and Pendley observed that the mathematical notion of *fiber bundles* provides a useful abstraction for an object-oriented scientific data model of considerable generality [5,6]. This paper specializes and extends this idea to incorporate localized, piecewise field descriptions. In addition, we describe compact data representations that exploit various forms of regularity in the data.

We have observed that non-mathematicians tend to be intimidated by some of the mathematical jargon used to describe fiber bundles. For this reason, we deviate somewhat from the standard mathematical terminology in this paper. The interested reader can find a treatment of the topic of fiber bundles in [7].

The following section presents an overview of the data model. Section III introduces the *field element* as the basis of piecewise field representations. Section IV introduces methods for compact representations on regular grids and section V presents an example application.

2 Overview of the Field Data Model

This section presents a scientific data model that is suitable for representing continuum fields. However, it can also represent discontinuous geometries (such as random sample points) and wireline structures (such as ball-and-stick molecular models). A *field* is an object comprised of a *base* and *dependent data*. Roughly

CH3046-0/91/0000/0298/$01.00 © 1991 IEEE

speaking, the base describes the range of the independent variables and the dependent data contains the corresponding dependent-variable information. More precisely, a base is a manifold whose coordinates are the independent variables for the field. The base might correspond to a geometric object, but this need not be so. The dependent data prescribes a value of the dependent variable for every position on the base.

A field is obtained by combining a base with dependent data. Figure 1 shows a simple field corresponding to a scalar function of a single variable. The base, the interval $[a, b]$ on the real number line, corresponds to the domain of the function. The range of the function forms a subset of the dependent variable space - the set of real numbers. The function is a mapping that assigns a member of the dependent variable space to every location on the base. In general, a field description includes specifications of the space that the base resides in, the base domain and the dependent variable space plus an assignment of a member of the dependent variable space to every base location.

Next, we discuss the fundamental entities in the data model - bases, dependent data and fields. These are the basis of the piecewise data model and the compact representation methods presented in subsequent sections.

2.1 Base Model

Figure 2 illustrates the base data model. A base is a manifold, $\Omega \subset B$; where B is a *base-coordinate space*. Every position on Ω is associated with a *base position vector*, $\Xi \in B$. We introduce a *manifold-coordinate space*, M, and a *manifold domain*, $\omega \subset M$, to describe Ω. The dimensions of M and B must satisfy the condition, $dim(M) \leq dim(B)$. A position on the manifold is designated by a unique *manifold position vector*, $\xi \in \omega$. The relation between Ω and ω is established by an invertible, differentiable mapping Φ: $\omega \rightarrow B$; such that Ξ and Ω are the images of ξ and ω under Φ.

$$\Xi = \Phi(\xi); \quad \Omega = \Phi(\omega) \qquad (1)$$

In Figure 2, the example base domain Ω is a curved surface in three-dimensional Cartesian space, $B = \Re^3$,

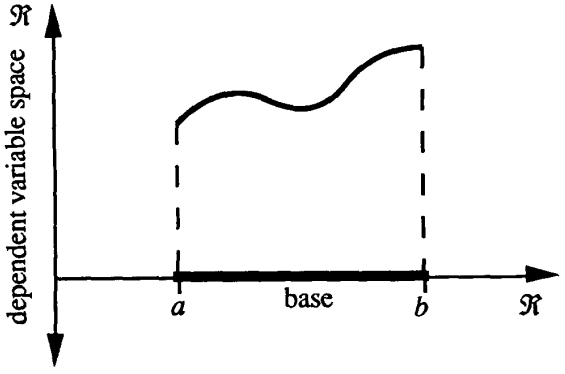

Figure 1 A simple field

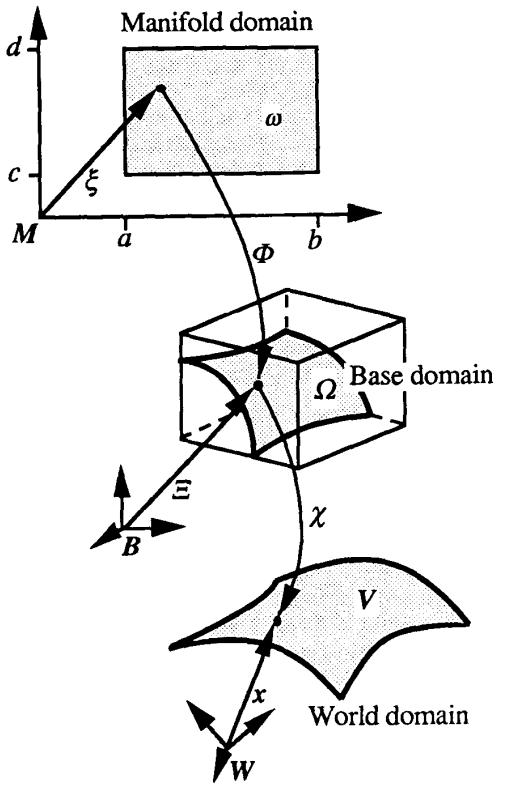

Figure 2 The base data model

and the example manifold domain is a rectangular region, $\omega = [a, b] \times [c, d]$ in two-dimensional Cartesian space, $M = \Re^2$. In this case, $dim(M) < dim(B)$. Note that $A \times B$ denotes the Cartesian product of A and B.

It is often useful to introduce a *world-coordinate space* W, as shown in Figure 2. For example, many computer graphics software systems use a three-dimensional Cartesian world coordinate system to describe displayable objects. Thus, if the base domain is described in a special coordinate system (cylindrical, spherical, toroidal, etc.), the base coordinates Ξ must be transformed to the Cartesian world-coordinate space prior to display. In other cases, the base-coordinate space might be defined locally. Then it is necessary to map the local base coordinates into the global world-coordinate space. In general, we define a mapping $\chi : \Omega \rightarrow W$, such that the *world position vector*, $x \in W$, and the *world domain*, $V \subset W$, are the images of Ξ and Ω under χ.

$$x = \chi(\Xi); \quad V = \chi(\Omega) \qquad (2)$$

The mapping χ to any given world-coordinate space W is a property of the base-coordinate space B. For the example in Figure 2, the mapping χ represents a simple rotation and translation, where $B = W = \Re^3$.

The base can be viewed alternatively as a manifold

embedded in either B or W. Note that the coordinates of the manifold position vector ξ are the independent variables and that the base coordinates Ξ and the world coordinates x are instances of dependent variables (although Φ and χ can be identity maps, so that the distinction between ξ, Ξ and χ might be blurred).

Topology describes the concept of a neighborhood within the base - given some position Ξ on the base Ω, what points are nearby? The answer to this question is important in a variety of visualization algorithms, such as trajectory integration, isosurface construction, smoothing algorithms and certain rendering algorithms. The differentiable mapping Φ provides a means to find the answer. The neighborhood of Ξ on Ω of radius $R > 0$ is defined as the set of points,

$$N(\Xi, R) \equiv \{v \mid v = \Xi + u;\ v \in \Omega;\ |u|_{\Omega} \leq R\} \quad (3)$$

2.2 Dependent-data Model

The dependent variable for a given field is designated by the symbol $y \in Y$, where Y is the dependent variable space. The dependent-data model consists of a description of Y and a definition of the value of y at every point on the base manifold. In other words, we seek a mapping Ψ: $\omega \rightarrow Y$, such that

$$y = \Psi(\xi) \quad \text{in } \omega \quad (4)$$

Sometimes it is more natural to define y as a function of the base coordinates, $y = \Psi'(\Xi)$, or the world coordinates, $y = \Psi''(x)$. In these cases, we can still construct explicit functions of ξ via the mappings Φ and χ.

$$y = \Psi(\xi) \equiv \Psi'(\Phi(\xi)) \quad \text{in } \omega \quad (5a)$$
$$y = \Psi(\xi) \equiv \Psi''(\chi(\Phi(\xi))) \quad \text{in } \omega \quad (5b)$$

Equations *(5)* emphasize that the manifold coordinates are the independent variables in the field data model.

2.3 Field Model

A *field space*, S, is the Cartesian product of a base domain, $\Omega \subset B$, and a dependent-variable space Y;

$$S = \Omega \times Y \quad (6)$$

and is defined by supplying a mapping Φ: $\omega \rightarrow \Omega$ and a space description for Y. The space descriptions for M and B and the definition of the manifold domain, $\omega \subset M$, are implicit in the mapping Φ.

A *field*, F, is obtained by assigning a specific value of the dependent variable, $y \in Y$, to every position on the base. This assignment is a mapping Ψ: $\omega \rightarrow Y$; where $y \in Y$ is the image of $\xi \in \omega$ under Ψ. Thus, a field F is a pair of mappings,

$$F = (\Phi, \Psi) \quad (7)$$

that share the same manifold domain ω. The space description for the dependent-variable space Y is implicit in the mapping Ψ. [In the standard mathematical nomenclature, a field space is called a *fiber bundle* and a field is a *fiber bundle section*.]

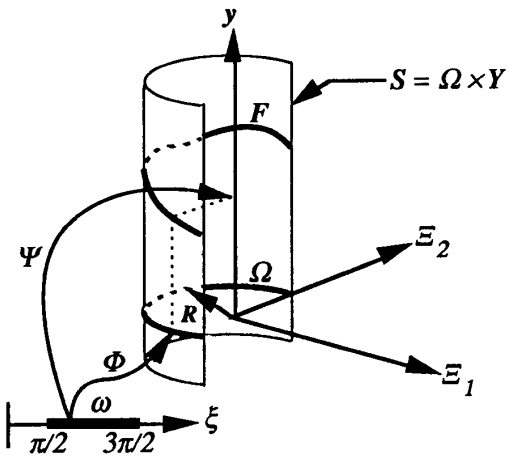

Figure 3 A field space S and a member field F

A field space is simply the union of all fields that share the same base Ω (described by the mapping Φ) and the same dependent-variable space Y. Figure 3 illustrates simple examples of a field and a field space. Here, $M = Y = \Re$ and $B = \Re^2$. The manifold domain ω is the interval $[\pi/2, 3\pi/2]$ and the mapping Φ is specified by $\Xi_1 = R\ cos\ \xi$; $\Xi_2 = R\ sin\ \xi$. The base Ω is the semi-circle of radius R in the second and third quadrants of B. The graph of the field space S is the half cylinder projected from Ω in the y direction. The addition of the mapping Ψ defines the field F, whose graph appears as a curve inscribed in S.

3 Piecewise Field Representations

This section introduces a component of the data model called a *field element* that supports piecewise field representations. Each field element describes the base and the dependent variable over a local region using a standard method of representation for the given element type. In other words, each field element describes a segment of the overall field, and the complete field is described by a collection of field elements that comprise a composite field group. Clearly, a field element is itself an instance of a field; but it is the standard manner of representation used to describe the base and the dependent variable that differentiates a field element from other field representations.

Field elements are especially useful when analytic representations of the base or dependent variable are not practical. For example, the entire base domain Ω might be too complicated to be described by a single mapping Φ and a simple range of the manifold coordinates ξ, or the variation of the dependent variable y might be too complicated to be conveniently described as a single analytic function over Ω. In such cases, it is often useful

to subdivide the base into a number of segments so that simple, local descriptions of the manifold ω and the mappings Φ and Ψ will suffice within each segment. Each segment corresponds to a field element and the base Ω is the union of the field element bases,

$$\Omega = \Omega_1 \cup \ldots \cup \Omega_{nel} = \Phi_1(\omega_1) \cup \ldots \cup \Phi_{nel}(\omega_{nel}) \quad (8)$$

where ω_α and Φ_α are the local manifold domain and base mapping for field element α and nel is the number of field elements used to describe the complete base Ω. This piece-by-piece, local representation of a field is consistent with many computational methods, such as finite element and finite difference methods.

A field element consists of a base representation, a dependent variable representation and a global topology representation. These are specified in a standard format that is unique to each *element type*. Each element type is defined by a unique combination of a manifold space, a range of the manifold coordinates to define the manifold domain ω, and parametric models for specifying the mappings Φ and Ψ. Due to the piecewise representation, the base topology must be defined both locally (within each element) and globally (via information linking each element to its neighbors). The local topology representation follows automatically from the local manifold coordinates. The global topology representation consists of a directory of elements adjacent to each face or vertex of each field element. A field element description includes a specification of the field element type and the data needed to define the global topology and to complete the parametric definitions of Φ and Ψ. Methods for describing these components of a field element are discussed next.

In a parametric representation, the base coordinates in field element α are expressed as functions of the manifold coordinates and a set of parameters, $c_{\alpha\beta}$; $\beta = 1, p_\alpha$.

$$\Xi = \Phi_\alpha(c_{\alpha 1}, \ldots, c_{\alpha p_\alpha}, \xi) \ \text{in} \ \omega_\alpha \quad (9)$$

It is often useful to introduce an interpolation form of *(9)*, such that the base coordinates are expressed as a linear combination of basis functions defined over ω_α.

$$\Xi = \sum_{\beta=1}^{p_\alpha} \Xi_{\alpha\beta}\phi_{\alpha\beta}(\xi) \ \text{in} \ \omega_\alpha \quad (10)$$

The basis functions are selected so that $\phi_{\alpha\beta}(\xi_\gamma) = \delta_{\beta\gamma}$; where $\delta_{\beta\gamma}$ is the Kronecker delta and ξ_γ is the manifold coordinate at the γ-th interpolation position in field element α. The combination coefficient $\Xi_{\alpha\beta}$ represents the value of Ξ at the β-th interpolation point in field element α.

The dependent variable is also represented in a parametric format,

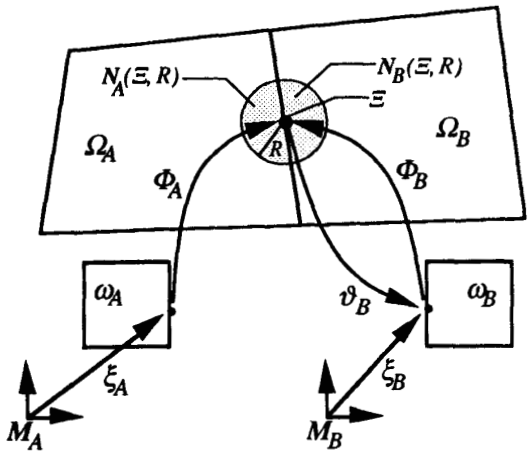

Figure 4 Continuation of the topology description across an interior boundary between two field elements

$$y = \Psi_\alpha(d_{\alpha 1}, \ldots, d_{\alpha q_\alpha}, \xi) \ \text{in} \ \omega_\alpha \quad (11)$$

that can be specialized to an interpolation form.

$$y = \sum_{\beta=1}^{q_\alpha} y_{\alpha\beta}\psi_{\alpha\beta}(\xi) \ \text{in} \ \omega_\alpha \quad (12)$$

The basis functions are selected so that $\psi_{\alpha\beta}(\xi_\gamma) = \delta_{\beta\gamma}$ and $d_{\alpha\beta} = y_{\alpha\beta}$, where $y_{\alpha\beta}$ is the value of y at the β-th interpolation point in field element α.

Note that the number of basis functions, the definitions of the basis functions and the interpolation positions need not be the same for the base mapping and the dependent variable mapping. However, it is sometimes convenient to choose $p_\alpha = q_\alpha$ and $\phi_{\alpha\beta} = \psi_{\alpha\beta}$. This choice corresponds to the *isoparametric* finite element model [8].

The differentiable coordinates ξ within each field element provide an implicit definition of topology for all interior positions. However, more information is needed to fully define the topology along the boundary of an element. Figure 4 illustrates the situation for an interior boundary between a pair of two-dimensional field elements. A common base position Ξ is identified by the manifold coordinates ξ_A and ξ_B in elements A and B, respectively. The neighborhood of Ξ of radius R is the union of two semi-circular regions, $N(\Xi, R) = N_A(\Xi, R) \cup N_B(\Xi, R)$; where the sets $N_A(\Xi, R)$ and $N_B(\Xi, R)$ are defined as,

$$N_A(\Xi, R) \equiv \{v \,/\, v = \Xi + u;\ v \in \Omega_A;\ |u| \le R\} \quad (13)$$

$$N_B(\Xi, R) \equiv \{v \,/\, v = \Xi + u;\ v \in \Omega_B;\ |u| \le R\} \quad (14)$$

Thus, to complete the topology definition for the points along the boundary of a field element, one needs to identity the adjacent element(s) and provide a means to

find the manifold coordinate ξ in the adjacent element that corresponds to a given position Ξ. We can store an ordered list of pointers to the neighbor elements of field element α, $n_{\alpha\beta}$; $\beta = 1, m_\alpha$, in which m_α is the number of adjacent field elements for element α and position in the list indicates the face or vertex of element α corresponding to each neighbor element. The neighbor list must include all adjacent elements, including those that reside in different fields in a composite field group. In order to support the field product operation described in section IV, each element α must be included in its own neighbor list.

Suppose we are given ξ_A on the boundary of element A and we wish to find the corresponding manifold coordinate ξ_B in element B. If we can find an inverse mapping (explicit or numerical), $\vartheta_B: \Omega_B \rightarrow \omega_B$ or $\vartheta_B=(\Phi_B)^{-1}$, then,

$$\xi_B = \vartheta_B(\Phi_A(\xi_A)) \qquad (15)$$

Practitioners of computational science and engineering should find the notions of fields and field elements familiar. For example, there is a direct correspondence between a finite element and a field element. Splines, surface patches and other constructs from computational geometry also fit the model well. Finite difference operators imply underlying basis functions that can be used to define consistent field elements. Significantly, computer-graphics-oriented data (whether it be a volume, surface or raster image description) can be subsumed within this model. Therefore, a separate data model is not needed to support rendering operations in a visualization system.

We are now prepared to summarize the information needed to describe a field as a collection of field elements. To begin, we need to know the space types B and Y for the field and the number of elements *nel* used to define the field. If the world and base spaces are not identical, we need a description of the world space W and the transformation χ (note that χ is a property of the pair (B, W) and need only be specified once for an entire field). Assuming that we are using a linear combination of basis functions to define Φ and Ψ in each field element, we need to list the following information (the interpolation forms, *(10)* and *(12)*, are assumed from here on):

For each field element $\alpha = 1, nel$:

- *Element type description:* $\{M_\alpha, \omega_\alpha \subset M_\alpha, p_\alpha, (\phi_{\alpha\beta};\ \beta = 1, p_\alpha), q_\alpha, (\psi_{\alpha\beta};\ \beta = 1, q_\alpha), m_\alpha\}$
- *Base parameters:* $\{\Xi_{\alpha\beta};\ \beta = 1, p_\alpha\}$
- *Dependent variable parameters:* $\{y_{\alpha\beta};\ \beta = 1, q_\alpha\}$
- *A list of neighboring elements* $\{n_{\alpha\beta};\ \beta = 1, m_\alpha\}$

If *nel* is large, then the amount of data needed to specify the field in the *verbose* format described above can be quite large. In some applications the subdivision of the field into elements does not exhibit any form of regularity. These representations are termed *irregular*, and it is difficult to improve on the verbose representation. Fortunately, in many applications the subdivision exhibits some form of regularity, so that a compact representation is possible, as described in the following section.

4 Compact Field Representations for Regular Grids

This section is concerned with exploiting regularity to achieve compact representations of the field model developed in the preceding section. The term regular, as applied to grid data in the visualization community, has been used to refer to a number of quite different concepts. For the purposes of this paper, we define regularity as any special structure in a field data model that can be exploited to achieve a compact representation. It turns out that there are a number of distinct forms of regularity that can occur in a field representation. In practice, these can occur singly or in combination. Therefore, it is useful to develop a group of complementary compact representation methods, corresponding to each class of regularity, that can be combined to achieve the maximum degree of data compaction. A classification of different forms of regularity and the corresponding compact representation methods are presented next.

4.1 Uniform Fields

A *uniform field* is a field that contains only one type of field element. In this case, the element type description can be made a property of the field, so that it does not need to be stored redundantly for every element in the field.

4.2 Shared Parameters

The base coordinates Ξ and the dependent variable y are often continuous across the boundaries between adjacent field elements. In these cases, it is typical to use the interpolation forms, *(13)* and *(16)*, with shared values of the parameters $\Xi_{\alpha\beta}$ and $y_{\alpha\beta}$ for base positions on the shared boundary. In the verbose representation the shared parameters must be repeated in every element in which they occur. Indirection can be used to achieve a compact representation when the number of shared parameters is large. The lists $\{\Xi_{\alpha\beta};\ \beta = 1, p_\alpha\}$ and $\{y_{\alpha\beta};\ \beta = 1, q_\alpha\}$ in the representation of each field element α are replaced by pointer lists $\{i_{\alpha\beta};\ \beta = 1, p_\alpha\}$ and $\{j_{\alpha\beta};\ \beta = 1, q_\alpha\}$ that refer to lists of shared-parameters, $\{\Xi_\beta;\ \beta = 1, p\}$ and $\{y_\beta;\ \beta = 1, q\}$, where p and q represent, respectively, the number of distinct base parameters and the number of distinct dependent-variable parameters in the field. Of course, the decision of whether to use indirection is an independent choice for the base and the dependent variable parameters. The use of indirection is effective when the number of shared parameters is large.

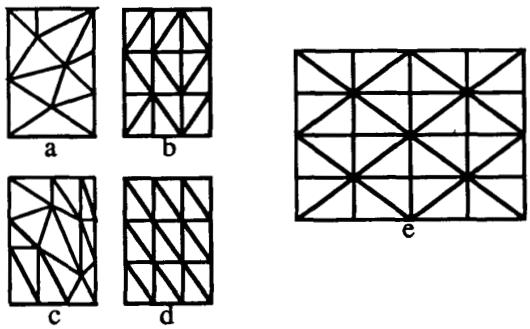

Figure 5 (a) irregular positions, irregular topology; (b) regular positions, irregular topology; (c) irregular positions, regular topology; (d) regular positions, regular topology; (e) a non-trivial, regular topology

4.3 Regular Base Positions

Sometimes it is possible to specify the base parameters $\Xi_{\alpha\beta}$ (or Ξ_β if indirection is used) implicitly. If the grid positions $\Xi_{\alpha\beta}$ or Ξ_β are evenly spaced or follow any regular pattern, then their values can be computed as functions of the indices α and β.

$$\Xi_{\alpha\beta} = \Xi^*(\alpha,\beta) \text{ or } \Xi_\beta = \Xi^*(\beta) \qquad (16)$$

This case of *regular positions* enables a very compact representation in which the function Ξ^* replaces the explicit base parameter list, $\{\Xi_{\alpha\beta};\ \beta = 1, p_\alpha;\ \alpha = 1, nel\}$ or $\{\Xi_\beta;\ \beta = 1, p\}$.

4.4 Regular Topology

A collection of field elements can also exhibit a *regular topology*. That is, the lists of pointers to the shared parameter lists, $\{i_{\alpha\beta};\ \beta = 1, p_\alpha;\ \alpha = 1, nel\}$ and $\{j_{\alpha\beta};\ \beta = 1, q_\alpha;\ \alpha = 1, nel\}$, and the lists of adjacent field elements, $\{n_{\alpha\beta};\ \beta = 1, m_\alpha;\ \alpha = 1, nel\}$, follow regular patterns. Each of these lists can be replaced by a single function that applies to the entire field.

$$i_{\alpha\beta} = i^*(\alpha,\beta) \qquad (17a)$$
$$j_{\alpha\beta} = j^*(\alpha,\beta) \qquad (17b)$$
$$n_{\alpha\beta} = n^*(\alpha,\beta) \qquad (17c)$$

Arbitrary permutations of regular and irregular positions and topologies are possible, as illustrated in Figure 5. The pattern underlying a regular topology can involve a patch of field elements, as shown in Figure 5(e).

4.5 Product Fields

The Cartesian product operation is a powerful tool for generating compact representations of higher-dimensional fields using lower-dimensional fields. The product field representation supports very compact descriptions and is ideally suited for describing fields that exhibit partially regular positions and/or topology.

A *product field* is the Cartesian product of two lower-dimensional fields. Suppose we have defined a pair of field spaces that share the same dependent-variable space, $S_A = \Omega_A \times Y$ and $S_B = \Omega_B \times Y$. The product field space $S = S_A \times S_B \equiv \Omega \times Y$, where

$$\Omega = \Omega_A \times \Omega_B \qquad (18)$$

and is formed from $nel = (nel)_A (nel)_B$ elements, each designated by an index pair $\alpha\beta$, where $\alpha = 1, (nel)_A$ and $\beta = 1, (nel)_B$. For element $\alpha\beta$ in the product field we have,

$$M_{\alpha\beta} = (M_\alpha)_A \times (M_\beta)_B; \quad \xi_{\alpha\beta} = ((\xi_\alpha)_A, (\xi_\beta)_B) \qquad (19a)$$

$$\omega_{\alpha\beta} = (\omega_\alpha)_A \times (\omega_\beta)_B \qquad (19b)$$

$$p_{\alpha\beta} = (p_\alpha)_A (p_\beta)_B \qquad (19c)$$

$$\Xi(\xi_{\alpha\beta}) = \sum_{\gamma=1}^{(p_\alpha)_A} \sum_{\delta=1}^{(p_\beta)_B} \Xi_{\alpha\beta\gamma\delta}\, \phi_{\alpha\beta\gamma\delta}(\xi_{\alpha\beta}) \qquad (19d)$$

$$\Xi_{\alpha\beta\gamma\delta} = ((\Xi_{\alpha\gamma})_A, (\Xi_{\beta\delta})_B) \qquad (19e)$$

$$\phi_{\alpha\beta\gamma\delta}(\xi_{\alpha\beta}) = (\phi_{\alpha\gamma}(\xi_\alpha))_A (\phi_{\beta\delta}(\xi_\beta))_B \qquad (19f)$$

$$q_{\alpha\beta} = (q_\alpha)_A (q_\beta)_B \qquad (19g)$$

$$y(\xi_{\alpha\beta}) = \sum_{\gamma=1}^{(q_\alpha)_A} \sum_{\delta=1}^{(q_\beta)_B} y_{\alpha\beta\gamma\delta}\, \psi_{\alpha\beta\gamma\delta}(\xi_{\alpha\beta}) \qquad (19h)$$

$$\psi_{\alpha\beta\gamma\delta}(\xi_{\alpha\beta}) = (\psi_{\alpha\gamma}(\xi_\alpha))_A (\psi_{\beta\delta}(\xi_\beta))_B \qquad (19i)$$

$$m_{\alpha\beta} = (m_\alpha)_A (m_\beta)_B \qquad (19j)$$

$$n_{\alpha\beta\gamma\delta} = ((n_{\alpha\gamma})_A, (n_{\beta\delta})_B) \qquad (19k)$$

where each parameter and neighbor pointer in the product field element is denoted by the index pair $\gamma\delta$. The base parameters of the product field, $\Xi_{\alpha\beta\gamma\delta}$, and the corresponding basis functions, $\phi_{\alpha\beta\gamma\delta}(\xi_{\alpha\beta})$, are generated automatically by the product operation, as seen in *(19e,f)*. By definition, the dependent-variable space Y is the same as in the fields A and B (it is not derived from a product operation). Although the basis functions for the dependent variable in the product field, $\psi_{\alpha\beta\gamma\delta}(\xi_{\alpha\beta})$, are generated by the product operation (see equation *(19i)*, the dependent-variable parameters, $y_{\alpha\beta\gamma\delta}$, must be specified separately.

The product operation generates implicit element type

descriptions (equations *(19a,b,c,d,f,g,h,i,j)*) and implicit topology descriptions (equation *(19k)*). Equation *(19e)* provides the basis of a compact base representation. Only $(p_\alpha)_A + (p_\beta)_B$ base parameters need be stored for each product element $\alpha\beta$, instead of $(p_\alpha)_A(p_\beta)_B$ parameters, as in the verbose form of *(19d)*. If either of the component fields feature shared parameters, regular positions or regular topology, then the corresponding compact representations can be combined with *(19e)* and *(19k)* to obtain an extremely compact storage scheme (see the example in the following section).

Figure 6 shows how the product operation supports compact representations of fields that are only partially regular. The component field shown in (a) has non-uniform element types, irregular topology and irregular positions, while the component field in (b) has a uniform element type, a regular topology and regular positions. A compact representation of the second field is used, based on a single element type description, a field-element count of *4*, a grid-point count of *5* and a uniform spacing, Δ. The shared list of base parameters is generated by the implicit formula, $\Xi_\alpha = (\alpha - 1)\Delta;\ \alpha = 1, 5$. The pair of pointers into this list for field element α, $(i^*(\alpha,1), i^*(\alpha,2))$, are given by $(\alpha, \alpha + 1)$ for $\alpha = 1,4$. The implicit pair of neighbors for field element α, $(n^*(\alpha,1), n^*(\alpha,2))$, is given as $(\alpha - 1, \alpha + 1)$ for $\alpha = 1,4$ (with the obvious modifications for the first and last elements). A verbose representation is used for the first component field, except that a shared list of base parameters is used. The two component representations, combined with equations *(19)* completely define the product field shown in (c), except for the specification of the dependent-variable parameters, $y_{\alpha\beta\gamma\delta}$. The product operation can also be applied to field element types, as shown in figure 7, to generate an extensive library of compound element types from various permutations of simple field element types.

The verbose format described in section III provides the flexibility needed to describe any field representation, but it does not necessarily provide for efficient storage. The various compact representation techniques presented in this section are complementary, and can be used in combination to take advantage of all of the available regularity in a given field description. Nonetheless, these representations are best understood as "shorthand" versions of the verbose field representation.

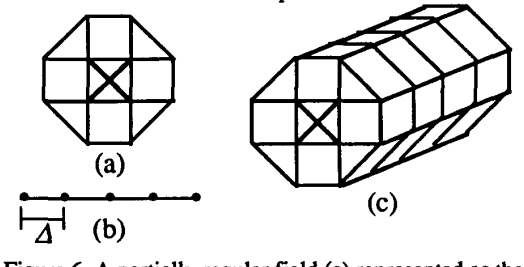

Figure 6 A partially regular field (c) represented as the product of an irregular field (a) and a regular field (b)

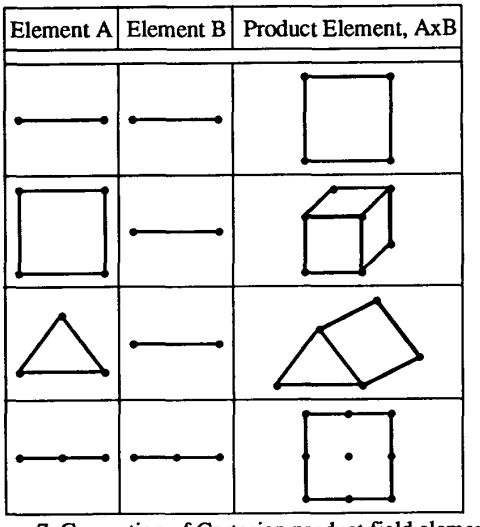

Figure 7 Generation of Cartesian product field elements

5 Example

This example illustrates the application of several of the compact representation techniques to the description of the toroidal solid shown in figure 9. The solid is generated by the Cartesian product of three component fields, designated *A*, *B* and *C*, shown in Figure 8. Each of the component fields is a uniform field, so the field element type description only needs to be stated once (in fact, all three fields share the same field element type). We have, $M_\alpha = \Re$; $\omega_\alpha = [0, 1]$; $p_\alpha = 2$; $\phi_{\alpha 1}(\xi) = 1 - \xi$; $\phi_{\alpha 2}(\xi) = \xi$; and $m_\alpha = 3$, for all values of $\alpha = 1, nel$ and for all fields F_A, F_B and F_C. That is, all of the component fields are based on one-dimensional elements with linear interpolation functions, each (except the first and last) having two neighbor field elements. Furthermore, we specify that the same interpolation functions will be used for the dependent-variable model, so $q_\alpha = p_\alpha$ and $\psi_{\alpha\beta} = \phi_{\alpha\beta}$.

Next we describe the bases for the three component fields. The base spaces B_A, B_B and B_C correspond to the coordinates Θ, θ and r, respectively, of the toroidal coordinate system with major radius R shown in Figure 8. Each of the component fields has regular positions (uniform spacing) and regular topology, so we only need to specify *nel* (8, 7, 10), a starting value $\Xi_0(0, -9\pi/10, 0)$ and an increment Δ $(\pi/6, \pi/5, \overline{r}/10)$ for each component field. Then the shared list of base parameters and the pointer lists for each field are given by,

$$\Xi_\beta = \Xi^*(\beta) = (\beta - 1)\Delta + \Xi_0 \qquad (20a)$$

$$i_{\alpha\beta} = i^*(\alpha,\beta) = (\alpha - 1) + \beta;\ \alpha = 1, nel;\ \beta = 1, p_\alpha \qquad (20b)$$

$$j_{\alpha\beta} = j^*(\alpha,\beta) = (\alpha - 1) + \beta;\ \alpha = 1, nel;\ \beta = 1, q_\alpha \qquad (20c)$$

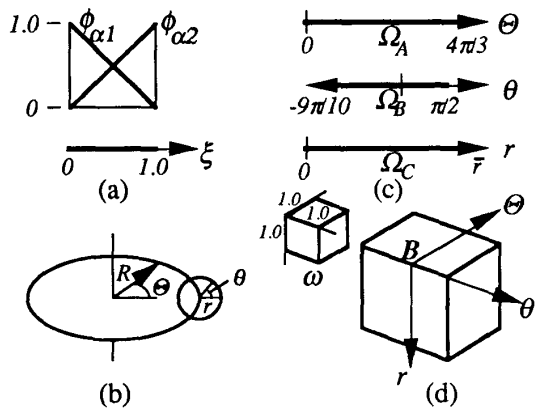

Figure 8 Compact, piecewise model of a toroidal solid (a) typical manifold domain and basis functions for a field element; (b) toroidal coordinate system; (c) base domains of the component fields; (d) manifold domain of product field primitive and product base domain

$$n_{\alpha\beta} = n^*(\alpha,\beta) = (\alpha - 2) + \beta;\ \alpha = 1, nel;\ \beta = 1, m_\alpha \quad (20d)$$

where we have taken care to include each element in its own neighbor list. We exclude elements *0* and *nel + 1*, respectively, from the neighbor lists of the first and last element in each field.

The desired field is specified by the product $F_A \times F_B \times F_C$ and the shared parameter list, $y_{\alpha\beta\gamma\kappa\lambda\rho}$. The first three subscripts in the dependent-variable parameter list identify the field primitive and the last three identify the interpolation point. A typical field primitive and the base of the product field are shown in Figure 8d. The product field geometry is displayed in a three-dimensional, Cartesian world coordinate space in Figure 9, where the mapping χ is given by,

$$x = (R + r\cos\theta)\cos\Theta;\ y = (R + r\cos\theta)\sin\Theta;\ z = \sin\theta \quad (21)$$

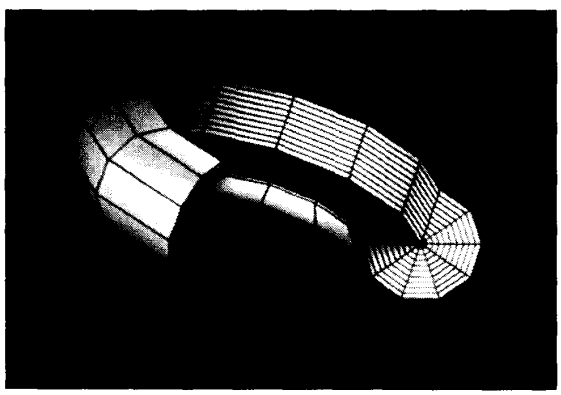

Figure 9 Segment of a toroidal solid obtained by mapping the base of the product field in figure 8d to Cartesian world coordinates. The gridlines indicate the division of the base into field elements.

6 Conclusions

We have presented a general mathematical model for piecewise representations of scientific field data. The model clarifies the meaning of the term "regular grid," and demonstrates that there are actually several forms of regularity. Each of these can be exploited to obtain a compact representation. Data structures based on this mathematical model appear to be attractive for implementation in general scientific computing applications as well as in visualization systems.

In this paper we have concentrated on the problem of scientific data representation, as opposed to data manipulation and data processing. In fact, the data model presented here also provides a suitable basis for object-oriented visualization and scientific computing systems. The object class hierarchies and object functions of such a system will be described in a later paper.

References

1. Treinish, L. A., "Data Structures and Access Software for Scientific Visualization," (Summary of a workshop held at ACM SIGGRAPH '90), *Computer Graphics*, 1991, (to appear).

2. Treinish, L. A. and M. L. Gough, "A Software Package for the Data Independent Management of Multi-Dimensional Data," *EOS Transactions*, American Geophysical Union, Vol. 68, 633 - 635, 1987.

3. Rew, R. K. and Davis, G. P., "NetCDF: An Interface for Scientific Data Access," *Computer Graphics and Applications*, IEEE, July, 1990.

4. National Center for Supercomputing Applications, "NCSA HDF Calling Interfaces and Utilities," Version 3.0, 1989.

5. Butler, D. M. and Pendley, M. H., "The visualization management system approach to visualization in scientific computing," *Computers in Physics*, Vol. 3, 40-44, 1989.

6. Butler, D. M. and Pendley, M. H., "A Visualization Model Based on the Mathematics of Fiber Bundles," *Computers in Physics*, Vol. 3, 45-51, 1989.

7. Burke, W. L., *Applied Differential Geometry*, Cambridge University Press, Cambridge, U.K., 1985.

8. Zienkiewicz, O. C. and Taylor, R. L., *The Finite Element Method, Volume 1*, 4th Ed., McGraw Hill, London, 1989.