A Perspective on Requirements for iField

Mark C. Miller, LLNL

iMesh Supports Interoperable Meshes

iField should support Interoperable Fields

What is a *field*?

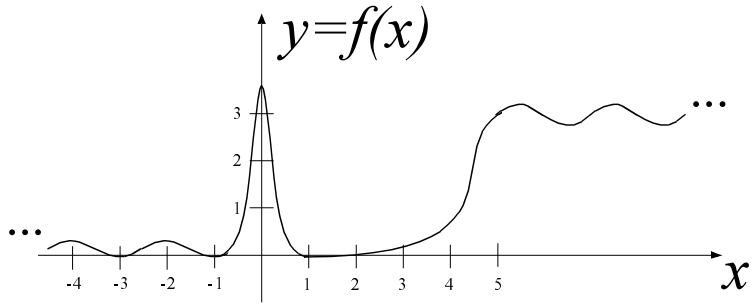
What does it mean for a field to be *interoperable*?

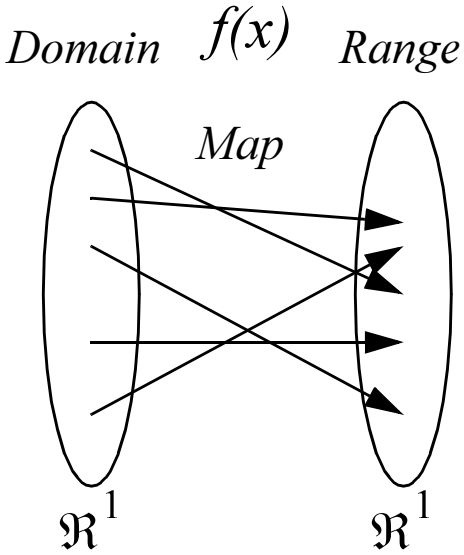
What is a Field?

The concept of a field serves as the standard for real (or imagined) observables in the real, continuous world

In terms of scientific computing, fields are the dependent (independent too) variables in a simulation

I think of a Field kinda-sorta like a Function



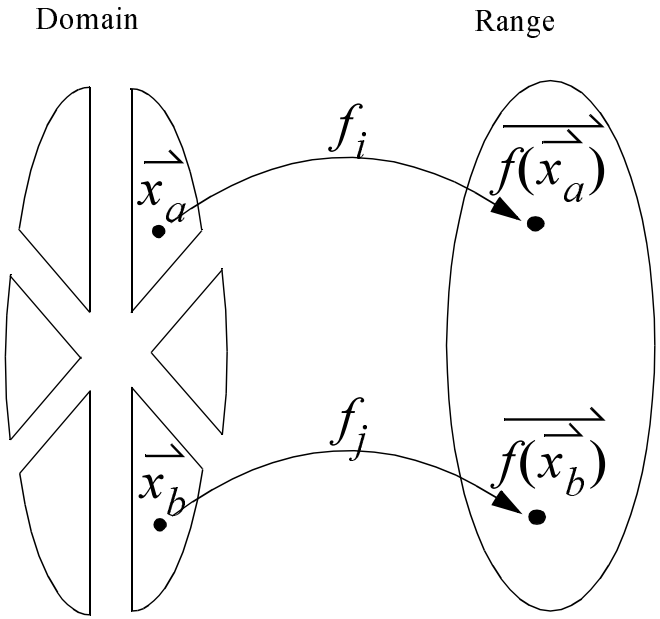


Three Logical Parts: Domain, Range, Map

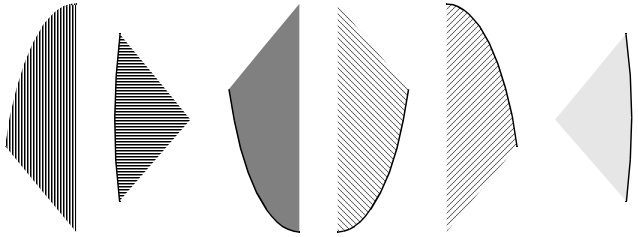
$$\Re^n \overset{f}{\Rightarrow} \Re^m$$

- *where n and m are the "dimensions" of the domain and range*

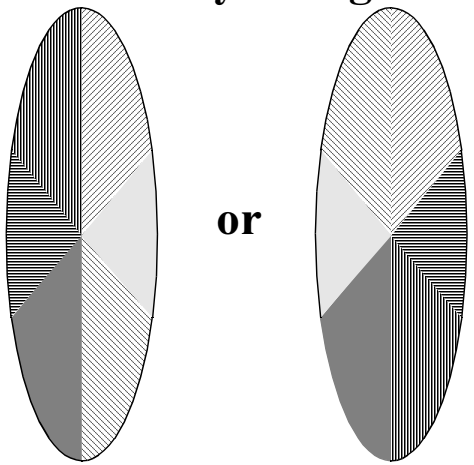
Often, the domain is decomposed into a bunch of similarly treated pieces



Given these pieces...



How do they fit together...

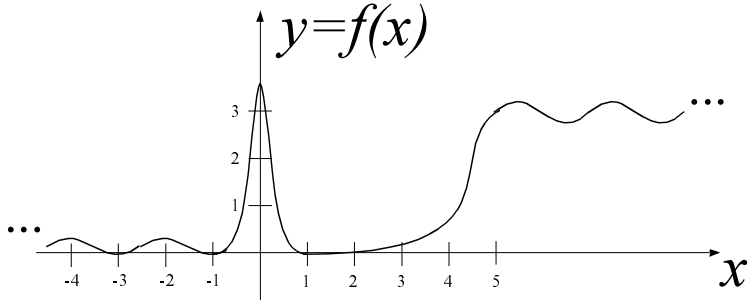


Topology: how pieces of domain (sets) relate to each other

- *Purely set theoretic in nature (no fields, not even coordinate field)*

An "Ideal" Field vs. A "Numerical" Field

Ideal: Infinitely many values with infinite precision



How do we represent such a thing on a computer?

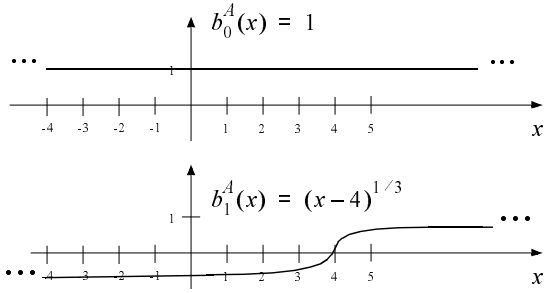
$$F(x) \cong \tilde{F}(x) = \sum_{i=0}^{N-1} f_i b_i(x)$$

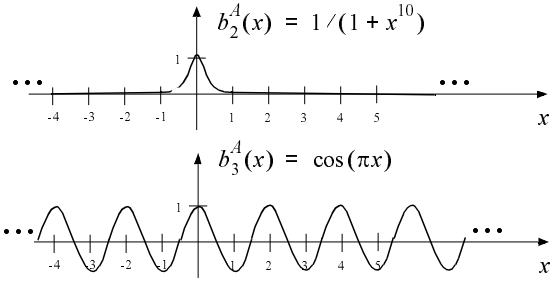
A finite number of coefficients, f_i, and a set of basis functions

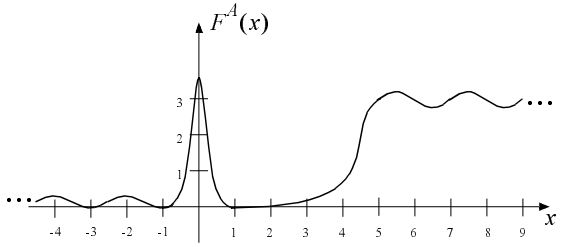
- *Still need to know the basis functions with infinite precision.*
- *Coefficients also called "dofs" (not to be confused with dofs of simulation)*

Four Different Representations Example #1:

$$\left\{ b_i^A(x) \right\} = \left\{ 1, (x-4)^{1/3}, 1/(1+x^{10}), \cos(\pi x) \right\}$$





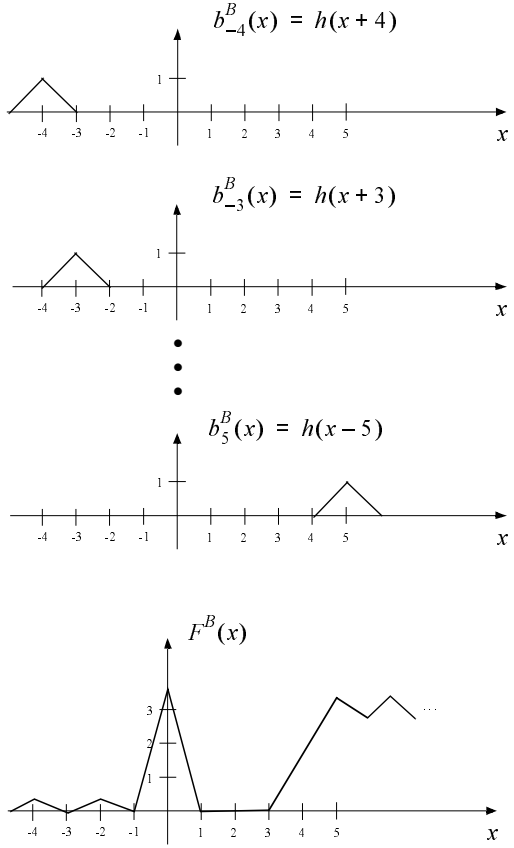


$$F^A(x) = \sum_{i \in [0,3]} f_i^A b_i^A(x)$$

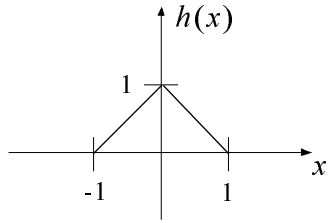
$$\left\{ b_i^A(x) \right\} = \left\{ 1, (x-4)^{1/3}, 1/(1+x^{10}), \cos(\pi x) \right\}$$

$$\left\{ f_i^A \right\} = \{ 1.5, 1.5, 3, 0.25 \}$$

Four Different Representations Example #2:



$$h(x) = \begin{cases} 1+x & -1 \le x \le 0 \\ 1-x & \text{for } 0 \le x \le 1 \\ 0 & \text{otherwise} \end{cases}$$



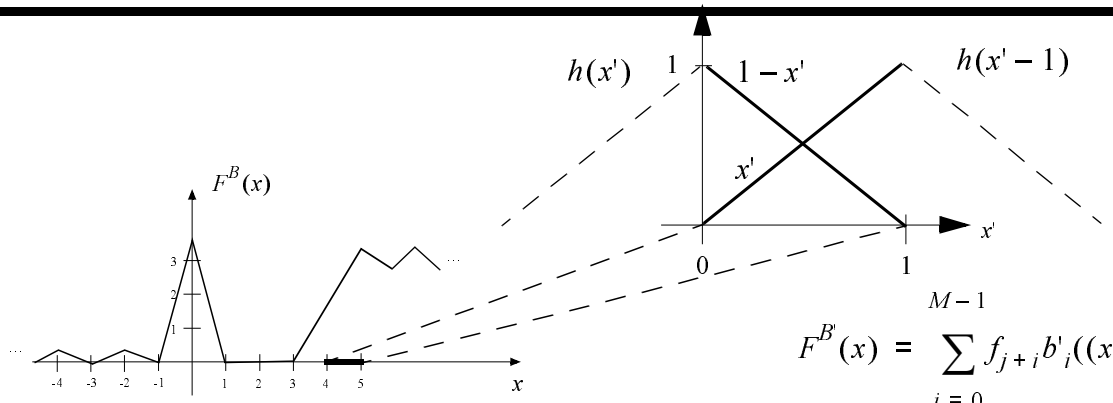
$$\left\{ b_i^B(x) \right\} = \{ h(x-i) | i \in -[4, 5] \}$$

$$F^B(x) = \sum_{i=-4}^{5} f_i^B b_i^B(x)$$

$$\left\{ b_i^B(x) \right\} = \{ h(x-i) | i \in [-4, 5] \}$$

$$\left\{ f_i^B \right\} = \{ 0.5, 0, 0.5, 0, 3.5, 0, 0, 0, 1.75, 3.5 \}$$

Four Different Representations Example #3:



$$F^{B'}(x) = \sum_{i=0}^{M-1} f_{j+i} b'_i((x - x_j)/(x_{j+1} - x_j))$$

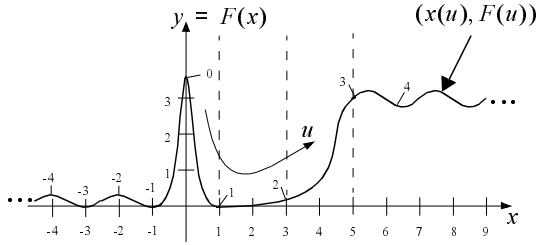
where j is determined such that $x_j \le x \le x_{j+1}$

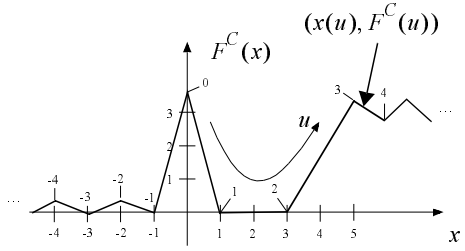
$$F^{B'}(x) = f_j b_0(x - j) + f_{j+1} b_1(x - j), j = \lfloor x \rfloor$$

$$\left\{ b_i^{B'}(x') \right\} = \{1 - x', x'\}$$

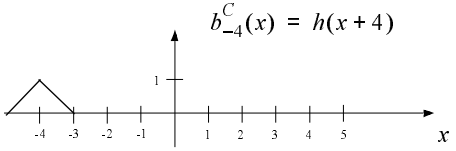
$$\left\{ f_i^B \right\} = \{0.5, 0, 0.5, 0, 3.5, 0, 0, 0, 1.75, 3.5\}$$

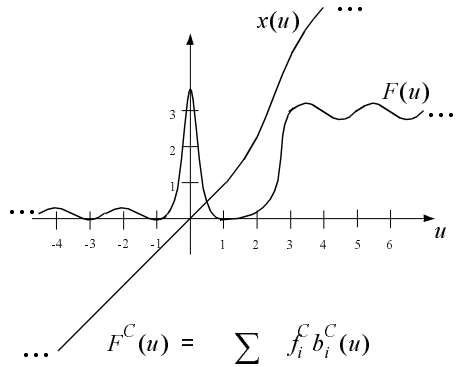
Four Different Representations Example #4:

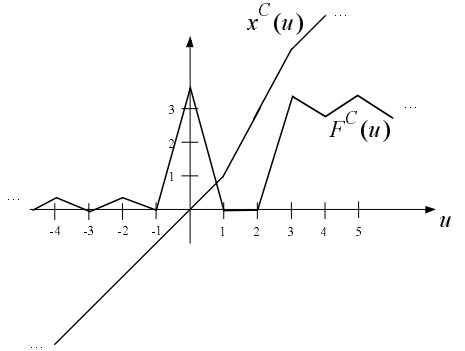


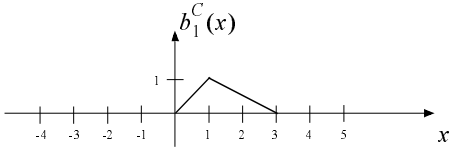


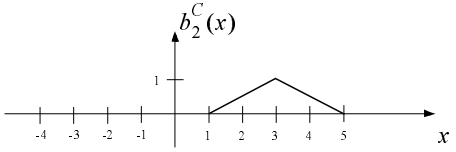
$$\{b_i^C(u)\} = \{h(u-i) | i \in [-4, 4]\}$$

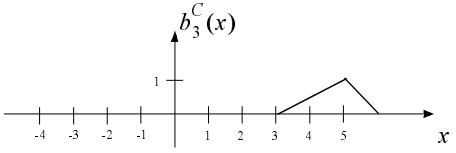












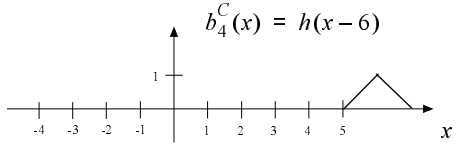
$$F^C(u) = \sum_{i \in [-4, 4]} f_i^C b_i^C(u)$$

$$x^C(u) = \sum_{i \in [-4, 4]} x_i^C b_i^C(u)$$

$$\{f_i^C\} = \{0.5, 0, 0.5, 0, 3.5, 0, 0, 3.25, 2.75\}$$
$$i = \quad -4 \quad -3 \quad -2 \quad -1 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4$$

$$\{x_i^C\} = \{-4, -3, -2, -1, 0, 1, 3, 5, 6\}$$
$$i = \quad -4 \quad -3 \quad -2 \quad -1 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4$$

$$\{b_i^C(x)\} = \left\{ \begin{array}{l} h(x+4), h(x+3), h(x+2), h(x+1) \\ h(x), b_1^C(x), b_2^C(x), b_3^C(x), h(x-6) \end{array} \right\}$$



What does it mean for a field to be "interoperable"?

That one application (the data producer) can give a field to another application (a data consumer) and the data consumer can "do something useful" with it...

- *how many bytes does it take to store? NOT VERY USEFUL*
- *Compute extrema (somewhat useful)*
- *Compute derivative or integral*
- *Compute the difference between it and some other field representing the same thing*
- *Make a change in basis*
- *"Graph" it ==> Implies everything above*

That different applications can *evaluate* the field anywhere

“Coordinates” are just a field

Among many fields, its essential to know which is the coordinate field

But that is about all that should distinguish the coordinate field from any other field

iMesh is already treating this field “specially”

Other useful things...

Can't have a field without a mesh

- *where "mesh" means domain upon which its defined*

Need to be able to define a field on any subset of a mesh

Would like to manage memory for a field separately from mesh and from other fields

- *Don't want to get ALL fields defined on a mesh when load/store*

Fields need units (and quantities)

Need to represent discontinuous fields

Need to define basis functions completely and precisely

- *not just node orderings over canonical shapes*
- *might even need to specify order of mathematcal operations in an evaluation*

Other useful things...

What about fields defined in terms of a "local coordinate system" derived from other fields?

- *Ex: Gradient of "pressure" field used as a local coordinate system for a "flux" field.*

When different processing elements evaluate on shared entities, may need a way of specifying which is "coorect"

- *Is that always going to be same as entity owning processor?*

Can we build iField on top of existing "tags" interface?

- *Only if iMesh_load can be modified to load SOME tags*
- *Only if we decide we'll only need to deal with fields whose dofs are n:1 w/mesh ents*

What is a Data Model?

It is a formal, conceptual tool for describing data, data relationships, data semantics and consistency constraints[1]...

...in a representation independent manner

In truth, representation independence falls on a spectrum

- *Whats one person's representation is another person's abstraction*

As a conceptual tool, a data model is always evolving!

1.Paraphrased from Silbershatz, Korth and Sudarshan, "Database System Concepts"

Why is a Data Model Important?

Because the manner in which data is described governs completely the generality of tools that can be developed to process it

In fact, a data model has far more compelling consequences for the software we write than the data we generate

- *Design of a data model is driven by its impact on the generality of software development it enables*

It Directly Effects Data and Software Interoperability