Computational Geometry Lab: FEM MESH FUNCTIONS ON A TRIANGULATION

John Burkardt
Information Technology Department
Virginia Tech
http://people.sc.fsu.edu/~jburkardt/presentations/cg_lab_mesh_function_triangle.pdf

August 28, 2018

1 Introduction

This *Computational Geometry* lab looks at *mesh functions*, that is, functions whose definition depends on the existence of some underlying mesh or triangulation of a region. To emphasize the special nature of such functions, we will often identify them as $\mathbf{f^h}(\mathbf{x}, \mathbf{y})$, where the h superscript is to remind us that the function is associated with a mesh with some characteristic mesh size h.

This lab follows the discussion of finite element basis functions over a single triangle. In that lab, it was easy to see how to define basis functions, evaluate and differentiate them, and thereby define interpolation functions, whether on the reference triangle or on a general triangle.

It might seem that, since a triangulation is simply a collection of triangles, there is little new or interesting to consider if we simply transfer what we know from the single triangle case. However, a new issue will complicate the picture immediately. The triangulation consists of neighboring triangles, and neighboring triangles share nodes. This means that when we consider a node, we will be considering the several elements that share it. When we define a basis function as a function that is 1 at a particular node and 0 at the others, this fact takes on new meaning for triangulations. And now, instead of the interpolation function being a simple linear function, it will be a piecewise polynomial function. That means, in particular, that our function will no longer be differentiable everywhere.

In this lab, we will try to get familiar with how the interpretation of basis functions must be extended, how we can define a mesh function as a linear combination of basis functions, and how we can differentiate and integrate a mesh function in a way that easily lends itself to automatic computation.

2 Overview

3 The Support of a Basis Function

4 Program #1: The Support of a Basis Function

5 Evaluating a Mesh Function

6 Program #2: Evaluating a Mesh Function

7 The Derivative of a Mesh Function

We have seen that a mesh function $\mathbf{f^h}(\mathbf{x},\mathbf{y})$ is a linear combination of basis functions. The properties of the basis functions guarantee, in turn, that the mesh function is continuous over the triangulated region, and is a polynomial function *inside* any particular triangular element. Hence, $\mathbf{f^h}(\mathbf{x},\mathbf{y})$ has a well-defined derivative as long as $(\mathbf{x},\mathbf{y})$ is an interior point of some element. However, if $(\mathbf{x},\mathbf{y})$ is a point on an edge shared by two elements, then the mesh function will generally fail to be differentiable.

In the finite element method, derivatives of mesh functions are usually only computed at interior points, as part of an integration process, so discontinuities at the element boundaries are of little concern. So let us concentrate on the case where we wish to know the values of $\mathbf{f^h_x}(\mathbf{x},\mathbf{y})$ and $\mathbf{f^h_y}(\mathbf{x},\mathbf{y})$ at some interior point.

xxxx

There may be cases, though, where derivative values are desired, particularly at the nodes of the mesh. This may be so because plotting software is available which expects data precisely at the nodes of the mesh. It may also be because certain physical quantities can be easily computed from derivatives of the function, and it would seem natural to compute and associate these derivatives at the nodes. If accuracy is important, then it would be far better to create a dual mesh, with nodes at the center of each of the original mesh elements. The values of the derivative computed at those nodes can be trusted. However, if accuracy is less important than a "quick and dirty" estimate, derivative values at the original nodes can be estimated by evaluating the derivative function as defined in each element that shares the node, and then, perhaps, weighting the results by the portion of the 360 degree angle around the node that is represented by that element. Thus a side node, shared by two elements, would simply average the two estimates. A node that is the vertex of several elements would weight each estimate by the element's angle at that vertex, divided by 360 degrees.

8 Program #3: The Derivative of a Mesh Function

9 The Integral of a Mesh Function

10 Program #4: The Integral of a Mesh Function