TTU REU: Exploring FEM with numpy

Bryan Hatton [1] Mentor: Wang Chunmei [2]

[1]UCSD [2]TTU

FEM

Starting with the PDE,

$$u'' = f \text{ on } [0,1] \tag{1}$$

$$u(0) = u(1) = 0 \tag{2}$$

we can consider the solution u to live in some ambient function space V, which is so far unspecified. Multiplying by an arbitrary $v \in V$, we can integrate by parts to obtain

$$(u', v') = (f, v) \tag{3}$$

Here, we have denoted $\int_0^1 fg\,dx$ as (f,g). Using (3), we can construct the ambient space V as

$$V = H_0^1([0,1]) = \left\{ v \in L^2([0,1]) \mid v' \in L^2([0,1]) \mathrm{v}(0) = v(1) = 0 \right\}$$

Notably, $v \in V$ are not necessarily twice differentiable, so we call (3) the weak formulation of (1). To solve the weak formulation, we look for u that satisfies (3) for all $v \in V$.

While V is a linear space, it is intractably large. The FEM introduces a sequence of finite dimensional sub-spaces V_h indexed by a parameter h. The true solution u is then approximated by u_h which satisfies (3) for only all $v \in V_h \subset V$. It turns out that u_h is the subspace-projection of u onto V_h.

Selecting a basis $\{\phi_i\}_{i=1}^n$ of V_h, we take $u_h = \sum_i c_i \phi_i$ and $v = \phi_j$ to obtain

$$\sum_i c_i(\phi_i', \phi_j') = (u_h', \phi_j') = (f, \phi_j)$$

Varying v over all basis $\{\phi_i\}_{i=1}^n$ yields the linear system

$$Ac = L \tag{4}$$

where $a_{ij} = (\phi_i', \phi_j')$ and $L_i = (f, \phi_i)$. Due to FEM's roots in engineering, the matrix A is often called the stiffness matrix. L is referred to as the load vector. We can solve (4) with traditional techniques to find the coefficients of u_h.

Our Implementation

We evenly space $2^n + 1$ points across $[0,1]$. We label these points x_i and take h to be the common inter-point distance $h = 2^{-n}$. We then form V_h as the piece-wise linear functions over this mesh. Specifically $v \in V_h$ is s.t. v restricted to $[x_i, x_{i+1}]$ is linear for any i.

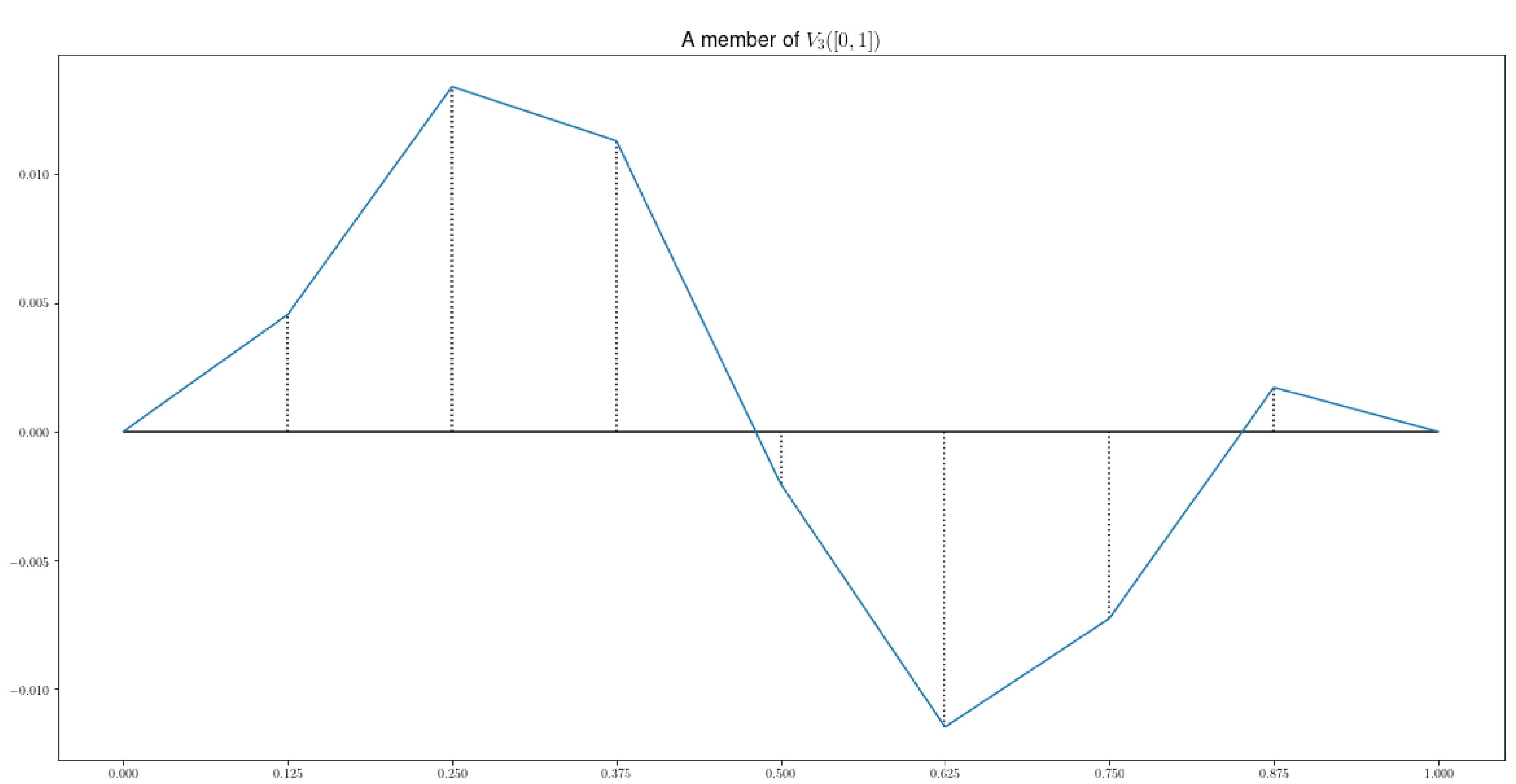


Figure 1:An example member of our choice of V_h.

Abstract

In this poster I present what I learned over the course of the 2021 summer Research Experience for Undergraduates (REU) program at Texas Tech University. I outline a standard Finite Element Method (FEM), then present code implementing the method to solve Poisson's equation in 1 dimension with Dirichlet boundary conditions. I then outline a Weak Galerkin method, a FEM variant, and present code solving Poisson's equation in 2 dimensions with Neumann boundary conditions.

W.G. Approximation

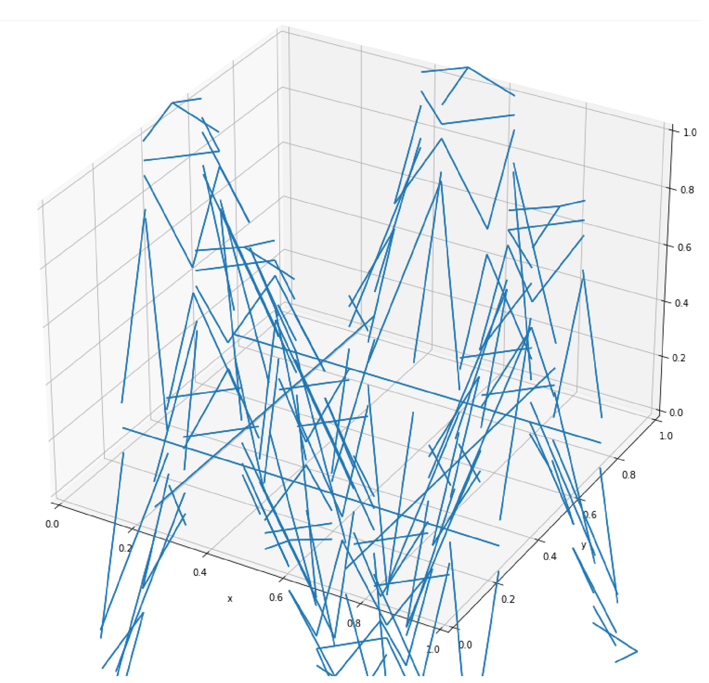


Figure 2:u_b with $h = 2^{-3}$

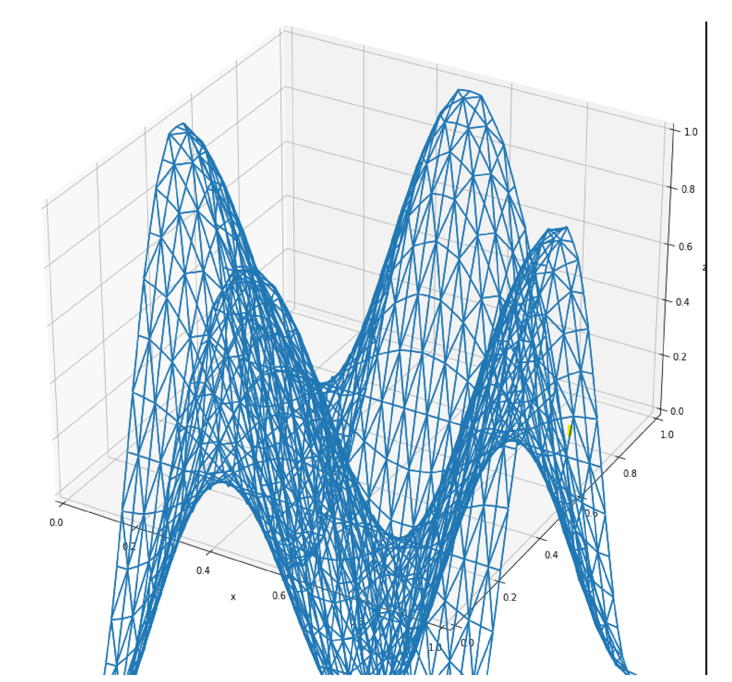
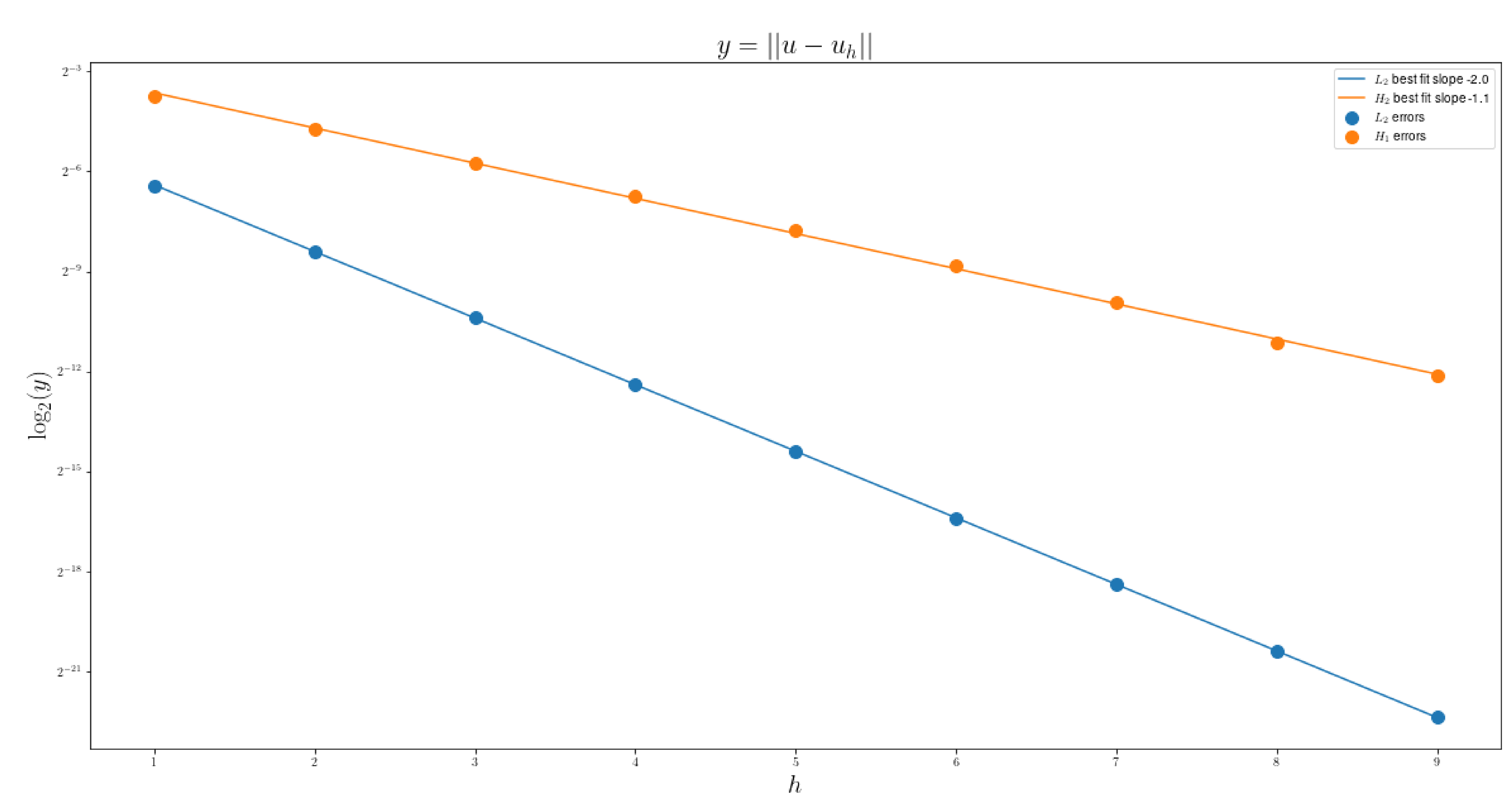


Figure 3:u_b with $h = 2^{-5}$

Convergence



Weak Galerkin

Solve

$$\begin{cases} -\Delta u &= f \text{ on } \Omega = [0,1]^2 \\ u &= g \text{ on } \partial\Omega \end{cases}$$

For the variational form, we find $u \in V_h$ s.t. $\forall v \in V_h$

$$(\nabla u, \nabla v) - < \nabla u \cdot \hat{n}, v >_{\partial\Omega} = (f, v)$$

By contrast, with the Weak Galerkin method, we look for $u = \{u_o, u_b\} \in W_h$ such that $u_b = Q_b\, g$ on $\partial\Omega$ and

$$(\nabla_w u, \nabla_w v) + S(u, v) = (f, v_o), \quad \forall v \in W_h^0$$

Here, Q_b is the L^2 projection onto $P_k(e)$, the polynomials of degree k. $S(u,v)$ is the stabilizer term defined by

$$S(u, v) = \sum_T h_T^{-1} < u_o - u_b, v_o - v_b >_{\partial T}$$

and the function spaces are

$$W_h = \{\{u_o, u_b\}|_T \in P_k(T) \times P_k(e)\}$$

$\mathrm{W}_h^0 = \{\{u_o, u_b\} \in W_h$ and $u_b = 0$ on $\partial\Omega\}$

$\nabla_w u \in [P_{k-1}(T)]^2$ is defined by

$$(\nabla_w u, \psi)_T = -(u_0, \nabla \cdot \psi)_T + < u_b, \psi \cdot \hat{n} >_{\partial T} \quad \forall \psi \in [P_{k-1}(T)]^2$$

We consider the case $k = 1$

Weak Gradient and Stabilizer

Unlike with the normal gradient, you cannot directly compute the weak gradient. The weak gradient, $\nabla_w u$ is a *distribution*. It is not a function per say, rather it is an operator on a space of functions. It "eats" another *test function* to produce a vector. Different u produce different distributions $\nabla_w u$.

$\nabla_w : W_h \times [P_{k-1}(T)]^2 \to \mathbb{R}$ is bilinear. So, if $u = \sum_i c_i \phi_i$, then $\nabla_w u = \sum_i c_i \nabla_w \phi_i$ as distributions. Moreover for any particular ϕ_i, the action of $\nabla_w \phi_i$ on $[P_{k-1}(T)]^2$ is determined by its action on the basis functions. Once we choose u, $\nabla_w u$ is just a row vector. We can therefore compute $(\nabla_w u, \nabla_w v)$ as $\int_\Omega (\nabla_w u) \cdot (\nabla_w v)$, which is just a dot product.

The stabilizer is also bilinear. So taking $u = \sum_i c_i \phi_i$ and $v = \phi_j$, we obtain,

$$(f, \phi_j) = -(\nabla_w \sum_i c_i \phi_i, \phi_j) + S(\sum_i c_i \phi_i, \phi_j) = \sum_i c_i \left[-(\nabla_w \phi_i, \nabla_w \phi_j) + S(\phi_i, \phi_j) \right]$$

This is the jth row of the linear system

$$L = Ac$$

with $L_i = (f, \phi_i)$ and $A_{ij} = -(\nabla_w \phi_i, \nabla_w \phi_j) + S(\phi_i, \phi_j)$