Analysis of Fekete Points

Jesse Chan, Joseph Munar

Abstract

Fekete Points are an adequate apporximation of Gauss-Legendre-Lobatto points for higher dimensions. This article attempts to analyze the effectiveness of Spline interpolation and applications using Fekete points. First, we attempt to recreate 1-Dimensional Gauss-Legendre-Lobatto points from Fekete points. Then we perform an error analysis on interpolation points using Fekete points for various maximum polynomial degrees and different sized knot vectors. We then compare the results to those performed for Gauss-Legendre-Lobatto points, uniform points, and Greville Abscissae. The report concludes with a formulation of variable problems for PDEs as well as an optimization to this process.

Introduction

Throughout the field of optimization, interpolation is extensively used. One of the many types of interpolation points used is the Gauss-Legendre-Lobatto points. However, this set of interpolation points becomes expensive very quickly and very little is known about this set in higher dimensions. Thanks to *insert cite here*, we know that Fekete points are adequate approximations of GLL points in higher dimensions. This article investigates the usefulness of Fekete points as well as its effects on numerical methods.

Fekete Points

We begin with the realization that GLL points are calculated using the standard Lagrange basis, and thus utilizes the standard Vandermonde matrix. The easiest way to extrapolate this to a higher tensor product of lines in 2D space is by projecting this interpolation over fixed intervals within the domain. As such, utilizing Spline spaces is the most appropriate direction for this project.

0.1 Spline Basis

A general Spline Basis (or B-Spline) for this project will be defined as: A basis of piecewise polynomials of max N degrees. Spline spaces are constructed using a knot vector **t**

$$\mathbf{t} = \{a = t_1, ..., t_{N+K+1} = b\}, \qquad t_i < t_{i+1}.$$

where K is the number of non-overlapping sub-intervals within the spline space. The B-splines can thus be calculated recursively using:

$$B_i^0(x) = \begin{cases} 1, & t_i \leq x \leq t_{i+1} \\ 0, & otherwise. \end{cases} \qquad B_i^K(x) = \frac{x - t_i}{t_{i+N} - t_i} B_i^{K-1}(x) + \frac{t_{i+N+1} - x}{t_{i+N+1} - t_{i+1}} B_{i+1}^{K-1}(x).$$

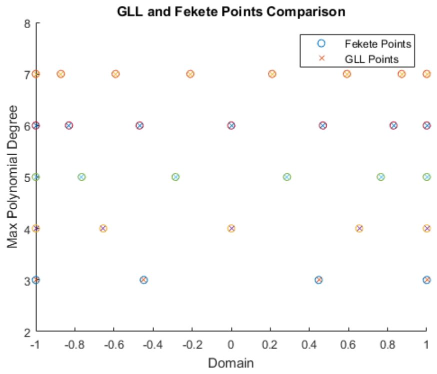
To achieve a unique solution to a spline interpolation, the knot vector must contain N+1 copies of the boundary terms, and there must be $N + K$ interpolation points. This allows for the existence of a square matrix which contains the coefficients of the B-splines for a unique set of interpolation points and knot vector. The entries of this matrix can be calculated using deBoor$'$s algorithm. This matrix will be referred to in this article as the Spline Vandermonde Matrix.

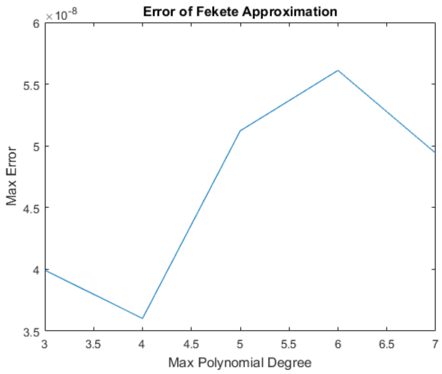
0.2 Fekete Points Generation Algorithm

A gradient ascension algorithm was used to generate the Fekete points for this project. As explained by (insert cite), one can apply $\frac{d\phi_i}{dr_i}$ to the i^{th} node in the set of interpolation points recursively. Fekete points are achieved once

$$\frac{d\phi_i}{dr_i} \leq 10^{-12}.$$

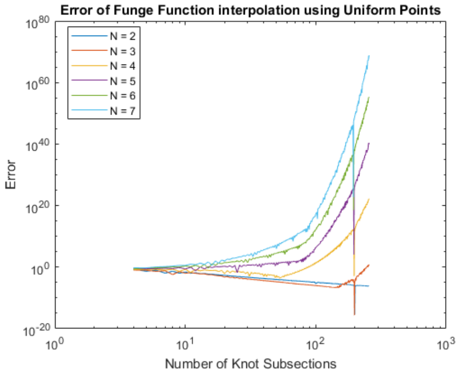
Since we are working with a square Spline Vandermonde matrix, then $\frac{d\phi_i}{dr_i}$ for all global nodes will be the diagonal entries of the square derivative matrix. In general, the derivative of a spline can be calculated through the same method as the spline itself, but will be of max $N - 1$ degrees and have a knot vector similar to the original spline but with the interior points duplicated. However, these derivatives are only locally defined, and require a map that will lead us to the global derivatives we desire. Applying a time stepping scheme such as Forward Euler or Runge Kutta 45 to the set of interpolation points and diagonal entries of the now-square derivative matrix give the desired Fekete interpolation points. As a precautionary check, setting $K = 1$ in the algorithm should provide Fekete points that are equivalent to GLL points for any N.

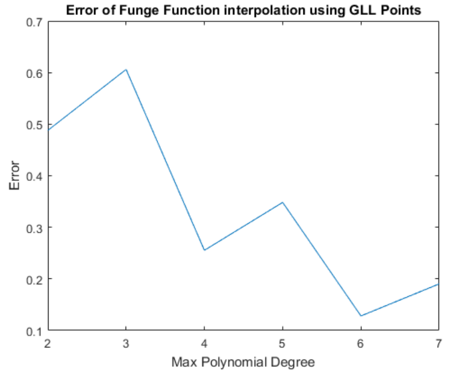


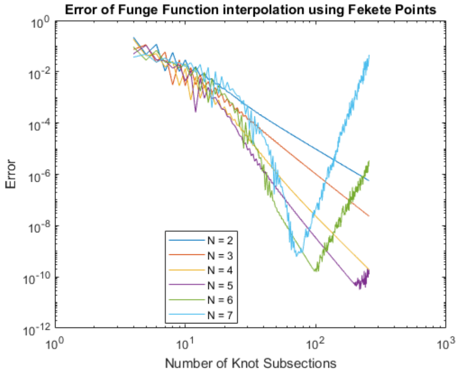


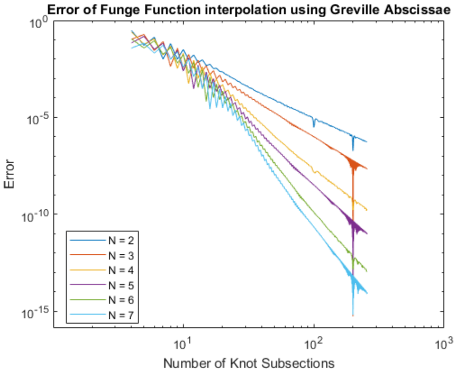
Error Analysis

We approximated the Lebesque constant of serveral types of points for several different functions. Fekete points, uniform points, GLL points, and Greville Abscissae were all compared in this analysis.









Variational Formulation for Time Dependent Problems

Finally, we attempt a classic finite element method approach to time dependent partial differential equations using B-Splines. We'll attempt this on the 1-D Wave Equation with homogeneous Neumann boundary and initial conditions:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}, \quad \frac{\partial u}{\partial x}(-1,t) = \frac{\partial u}{\partial x}(1,t) = 0, \frac{\partial u}{\partial t}(x,0) = 0$$

Where c is a constant and u is a solution function dependent on variables x and t. Inserting a test function, $v(x)$, and integrating gives:

$$\int \frac{\partial^2 u(x,t)}{\partial t^2} v(x) dx = c^2 \int \frac{\partial^2 u(x,t)}{\partial x^2} v(x) dx$$

Applying a discretization, $u(x,t) = \sum a(t)\phi(x)$, and invoking the boundary conditions simplifies the equation into a linear system:

$$\frac{\partial^2 \vec{a}}{\partial t^2} M + c^2 \vec{a} K = 0$$

where M is the mass matrix and K is the stiffness matrix defined as:

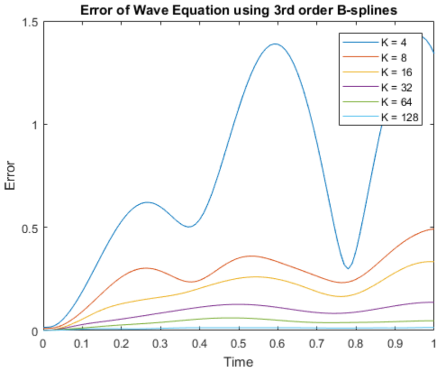
$$M_{i,j} = \int_{-1}^{1} \phi_i(x)\phi_j(x)dx, \qquad K_{i,j} = \int_{-1}^{1} \frac{\partial \phi_i}{\partial x}\frac{\partial \phi_j}{\partial x}dx$$

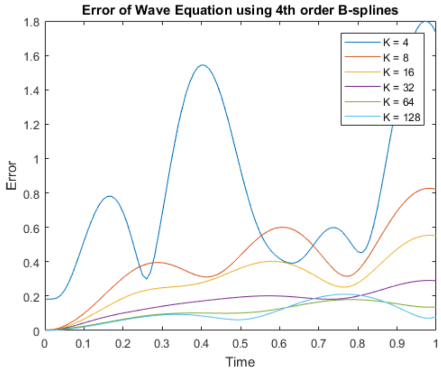
These equations can be solved using quadrature approximations, so we used Gaussian Quadrature points for this process. Using a Forward Euler approach, we get an explicit time leapfrogging method.

Next, we attempt to optimize the system using a mass lumping technique. That is, we force the mass matrix to be a diagonal matrix. For this, we switch from Gaussian Quadrature points to a set of nodal points. GLL points would be perfect, however, we'll switch to Fekete points to use the increased accuracy of higher tensor products. Let us introduce the equality of interpolation points:

$$(qF)^T \times \vec{wF} = (qQ)^T \times \vec{wQ}$$

where qF and qQ are the quadrature matrices for Fekete and Gaussian Quadrature points, and $\vec{wF}$ and $\vec{wQ}$ are their respective weight vectors. We can derive the weights of Fekete points from Gaussian Quadrature points using this equation. The only issue is that Fekete points are only an approximation of GLL points. Thus this equality may not be exact for higher tensor products. The error, as a result, may increase at a higher rate than expected. The results of the mass lumping approximation for time, $t = 1$ are below:


Error of Wave Equation using 3rd order B-splines
Error
Time
K = 4
K = 8
K = 16
K = 32
K = 64
K = 128


Error of Wave Equation using 4th order B-splines
Error
Time
K = 4
K = 8
K = 16
K = 32
K = 64
K = 128