Analysis of Fekete Points

Jesse Chan, Joseph Munar

Abstract

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. We then proced to demonstrate a formulation of variable problems for PDEs as well as an optimization to this process. 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 results and notes on the collocation procedure and optimization.

Introduction

Fekete Points are known to be the Gauss-Legendre-Lobatto points for the line and its tensor products, and has been determined to be an adequate approximation of Gauss-Legendre-Lobatto points on the traingle and its tensor products. (cite here) has determined an approach to calculating Fekete points which allows further study into its applications. In this article, we analyze the effectiveness of Fekete points for interpolation using B-splines. A general algorithm for calculating B-splines was created by deBoor. He also showed a very important note for this article: that the condition number of a B-spline basis increases exponentially as K increases.

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 singular dimension is by projecting this interpolation over fixed intervals within the domain. As such, utilizing Spline spaces is the most appropriate direction for this project.

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:

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

Interpolation

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. Inverting the Spline Vandermonde matrix gives the solution vector to the interpolation points.

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 ith 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.

GLL and Fekete Points Comparison / **Error of Fekete Approximation**
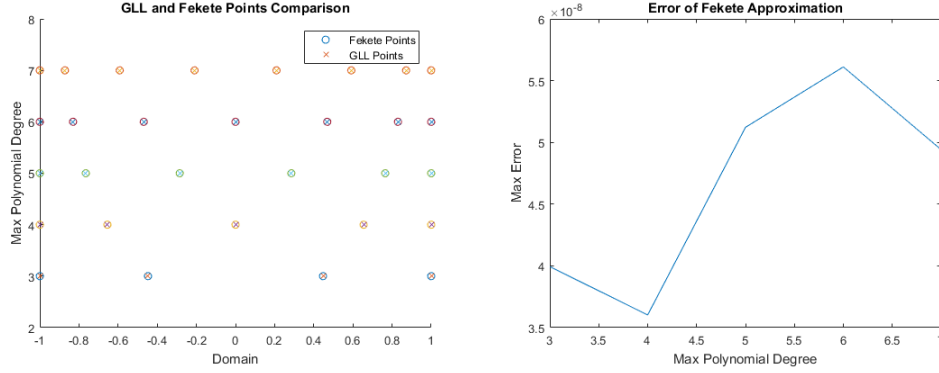
Variational Formulation for Time Dependent Problems

Finite Element Formulation

Next, 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:

$$M_{i,j} = \sum_{i,j=0}^{N} w_i \phi_i(x)\phi_j(x), \qquad K_{i,j} = \sum_{i,j=0}^{N} w_i \frac{\partial \phi_i}{\partial x} \frac{\partial \phi_j}{\partial x}$$

Where w_i is the weight of the ith interpolation point. To easily calculate these weights, we used Gaussian Quadrature points. Using a Forward Euler approach, we get an explicit time leapfrogging method:

$$\vec{a}^{(t+2)} = -M^{-1}K\vec{a}^{(t+1)}(c \times dt)^2 + 2\vec{a}^{(t+1)} - \vec{a}^{(t)}$$

Mass-Lumping

As for any explicit time stepping scheme, the mass matrix must be solved. However, these matrices are completely dense, and inversion quickly becomes less feasible and more numerically taxing as K and N increase. We wish to simplify this hardship by replacing the dense matrix with a diagonal matrix - a mass-lumped matrix. For this, we switch from Gaussian Quadrature points to a set of nodal points such as GLL points. Since we already have calculated the weights of the Gaussian Quadrature points and we can easily calculate the quadrature matrices of both types of points, we can derive the weights of the GLL points using:

$$(qG)^T \times \vec{wG} = (qQ)^T \times \vec{wQ}$$

where qG and qQ are the quadrature matrices for GLL and Gaussian Quadrature points, and $\vec{wG}$ and $\vec{wQ}$ are their respective weight vectors. Choosing to use Fekete points allow us to utilize the increased accuracy provided from the subintervals of the B-Spline basis. Using a nodal basis forces the Spline Vandermonde matrix to be the identity matrix. Therefore, the now-mass-lumped mass matrix, M_L is defined as:

$$M_{L_{i,j}} = \begin{cases} \vec{wG}_i, & i = j \\ 0, & otherwise. \end{cases}$$

However, since Fekete points are only approximations of GLL points, this diagonal matrix will not naturally be achieved. To fix this,we force the diagonalization of M by defining a diagonal matrix, M_L, whose diagonal entries are the weights of the Fekete points. Next, we implement a predictor-corrector method to revert our diagonal matrix to the originally dense mass matrix.
If we consider the general form of a Forward Euler system, we have:

$$Mx = b$$

Using the recursion relationship:

$$M_L x^{(i+1)} = b - AM_L x^{(i)}$$

where $A = MM^{-1} - Id$. According to (insert cite here), these matrices are guaranteed to converge to the solution, M. The numeric approximation achieved will be labeled M_r where r is arbitrarily chosen and M_r is defined as:

$$M_r := \left(\sum_{i=0}^{r-1}(-A)^i\right)^{-1} M_L$$

We then replace the previous system with the new numeric approximation:

$$M_r x = b$$

When can then expand the equation using the definition of M_r and solve for x. This gives the final form of the equation:

$$x = M_L^{-1} \sum_{i=0}^{r-1} (-A)^i b$$

In this form, both M and M_L are used, however, it is important to note that only M_L is inverted.
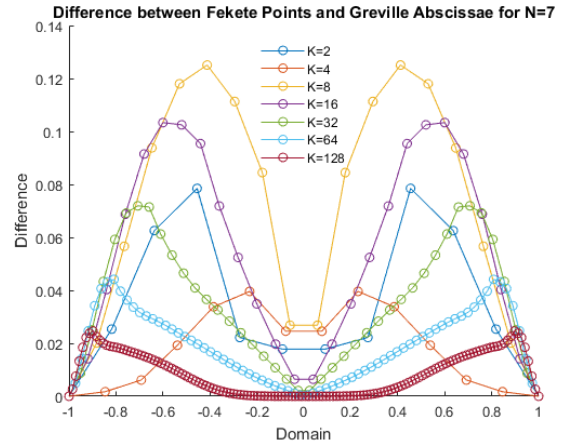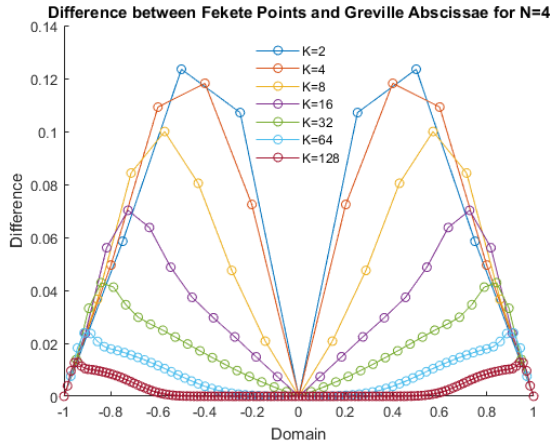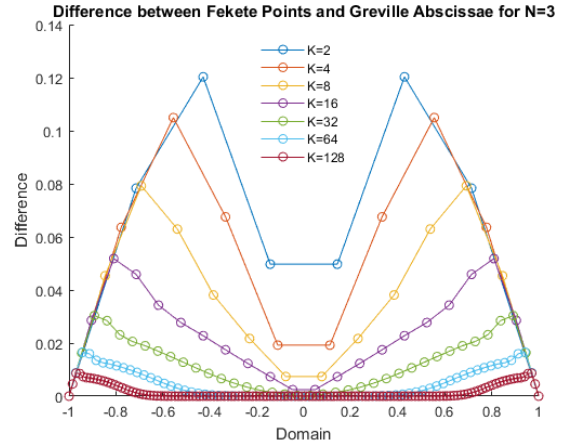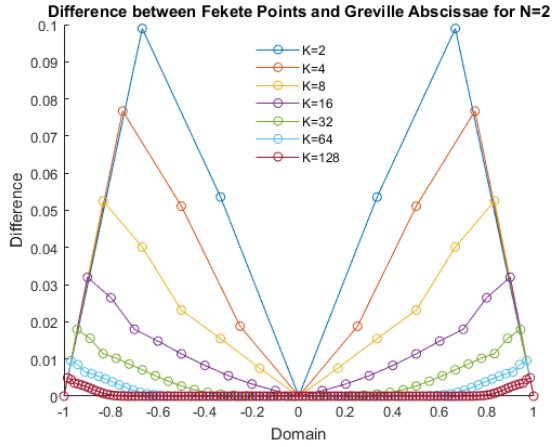
Numerical Experiments

Convergence of Time-Stepping to Achieve Fekete Points

Part of any gradient ascension algorithm is the time-step needed. In the case of this generation algorithm, it seems a very small time-step is needed. Through experimentation, it was observed that a higher N and/or K requires a smaller time-step. This is most likely a side effect of the Spline Vandermonde matrix becoming more unstable with higher N and K. A suitable formula to determine the time step needed seems to be $dt = \frac{1}{200 \times N \times K}$

## Analysis on Fekete Points and Greville Abscissae

Greville Abscissae are the averages of a knot vector. Because of this, they have been used extensively for interpolation using B-Splines. Initial error analysis of the two types of interpolation points show they have very similar results.

Further inspection of the points show they are almost identical. As N and K increase, more interior points of both types of interpolation points converge to the same results. The exterior points will never be the same due to how they are defined. For all other points, Fekete points seem to be more spread out whereas Greville abscissae seem to clump more towards the middle of the knot vector.

Difference between Fekete Points and Greville Abscissae for N=2

Difference between Fekete Points and Greville Abscissae for N=3

Difference between Fekete Points and Greville Abscissae for N=4

Difference between Fekete Points and Greville Abscissae for N=7

Interpolation Errors

We first examine the error for two smooth functions using GLL points, Fekete Points, uniform points, and Greville abscissae. As previously mentioned, the Spline Vandermonde matrix has a higher condition number as K and N increase. This effect is seen in both the uniform points graph and the Fekete Points graph. Fekete Points are affected by this more than Greville Abscissae most likely because Fekete Points are generated using a B-spline basis. The results are shown below:
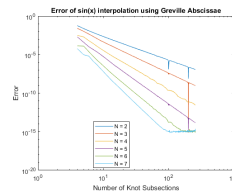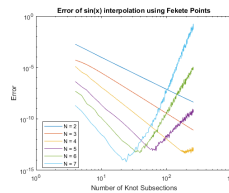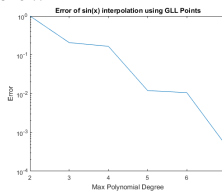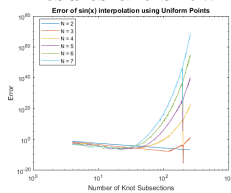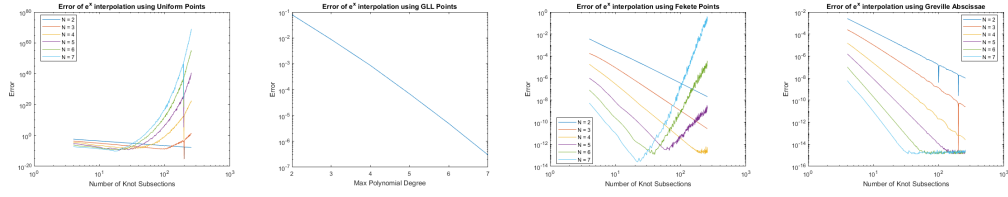
We then decided to examine the error for two smooth functions that experience the Runge phenomenon when interpolated. The same types of interpolation points were used. It was at this point where we first noticed the similarities between the Greville Abscissae and Fekete Points. Another important note is that a B-spline basis helps dramatically reduce the effects of Runge's Phenomenon:
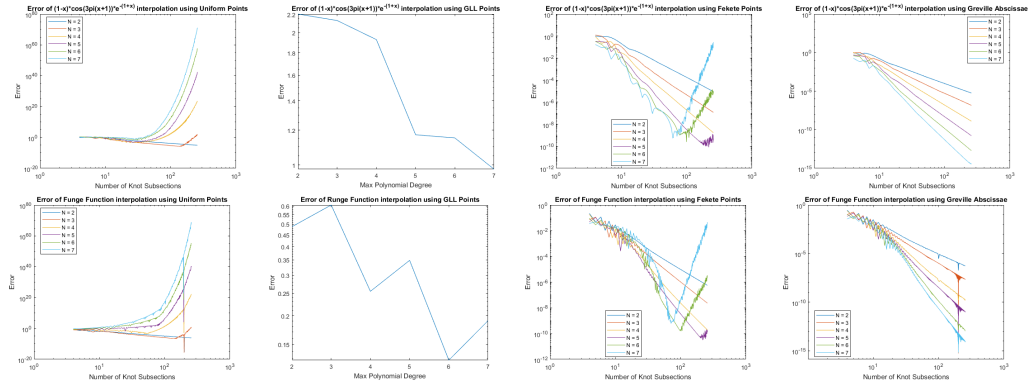
Mass-Lumping

The results of the mass lumping approximation for time, $t = 1$ are below:
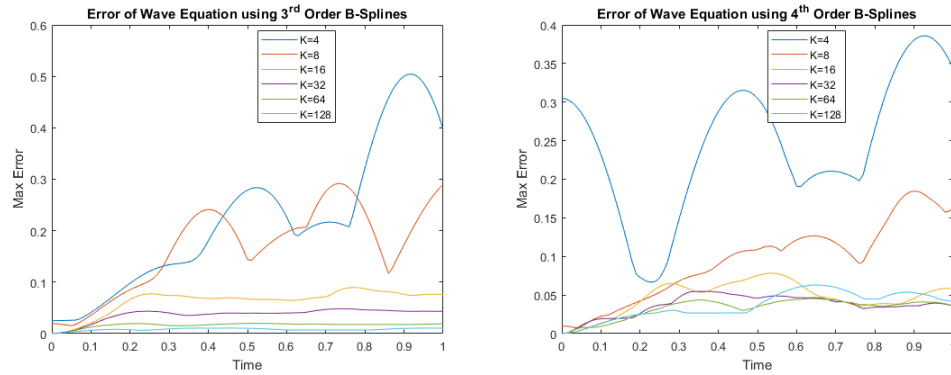
The variable r was never defined as a limit of any kind. It is simply an arbitrarily indicator for how accurate a system is desired to be - in this case, that system is the Mass matrix. An important observation made was that r does not depend on N nor K. Therefore, r can remain a singular constant for

any desire Spline Vandermonde matrix. Numerical experiments found r to be acceptably accurate at $r = 80$.

Appendix

A list of Fekete points for $K = 4 - 256$ for $N = 2 - 7$ have been written into 6 Excel documents saved on the public Github repository.