A Bezier Curve Method to Solving Boundary Value Problems

Nathan Halsey

November 2019

1 Introduction

A collocation method can be used to solve a boundary value problem using a Bezier curve, in a cannonical way, with great accuracy. In this project, I attempted to test this method using a quadratic Bezier curve, with mixed results. I did find that Bezier curves made it much easier to approximate a BVP, but also that without minimizing some approximated error, the error is too large to use my program to seriously solve BVP's.

2 Conceptual Framework of The Problem

For the remainder of the paper, we will be discussing boundary value problems of the form: $u''(t)+m(t)u'(t)+n(t)u(t) = f(t)$ with boundary values $u(0) = a$ and $u(1) = b$.

A Bezier curve can be written as $B(t) = \sum_{i=0}^{n} C_i B_i^n$, where C_i denotes a control point of our curve. The problem is of finding an accurate solution to this boundary value problem using some reiterative method. The finite difference method or the Runge-kutte method come to mind, but there may be a better method which approximates the curve better, without ”losing” information about the curve.

To go even further, solving with the quadratic does not require the solving of any linear systems.

3 Numerical model to be used to solve the problem

The numerical model used is the collocation method. The collocation method is utilised by approximating the solution of a differential equation as $U_n(t) = \sum_{i=0}^{n} P_i \mu(t)_i$, where $\mu(t)_i$ is a basis function in t, and P_i is a constant. It is without loss of generality that we can say that this greatly resembles a Bezier curve, so choosing $\mu(t)$ to be a set of Bernstein polynomials converts this problem into finding control point C_1 s.t $aB_0^2(t) + C_1 B_1^2(t) + C_2 B_2^2(t)$ approximates the solution to our differential equation.

The algorithm is as follows:

Step 1: Evaluate the Bezier curve at the given t value, based on what we define as our functions of $m(t)$, $n(t)$ and $f(t)$, evaluate even if the function is equal to 0. Store these values into an array. Then the array has values $[x_i]$ s.t $\sum_{i=0}^{n} [x_i P_i] = f(t)$.

Step 2: Solve for P_1 in this array, as it is the only unknown and this is one equation, it is very simple to do so.

Step 3: Put $P_0 = a, P_2 = b$ and the newly solved P_1 back into the approximate solution to receive the value at t.

4 Results

The results were just okay. The margin of error was anywhere between $0.001 \leq x \leq 0.01$, which is not a good approximation at these small values for h.

For the BVP $x''(t) - x(t) = 0$ and $x(0) = 1$, $x(1) = e$, the exact solution is $x(t) = e^t$.

And my margin of error was in the domain of $[0.00282, 0.01688]$.

This margin of error is not good, and I need to find an algorithm to achieve better results.

5 Conclusion

The collocation method with Bernstein polynomials works well, but I need to extend the program to calculate some error and make a better interpolation. In Gianluca Argentini's paper (which I'm finding difficult to believe) he mentions a way to continue to find more points and increase the degree of the Bezier curve, this should be explored and results should be disputed or not. It remains to be seen that the collocation method is a powerful method, but do Bernstein polynomials work better than Lagrange or Spline choices of polynomials?

6 Sources

Zheng et al, Least Squares Method for Solving Differential Equations using Bezier Control Points,
Gianluca Argentini, Numerical resolution of some BVP using Bernstein polynomials,
Conte, Elementary Numerical Analysis, an Algorithmic Approach (Section 9.3),
Bellomo et al, Generalized Collocation Methods