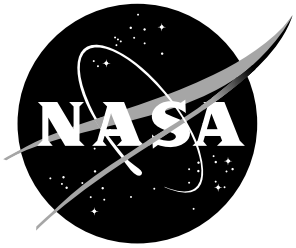
NASA/TM–20230013467



On Hermite Interpolation using Bernstein Polynomials for Trajectory Generation

Andrew Patterson
Langley Research Center, Hampton, Virginia

Gage MacLin
University of Iowa, Iowa City, Iowa

Michael Acheson
Langley Research Center, Hampton, Virginia

Camilla Tabasso
University of Iowa, Iowa City, Iowa

Venanzio Cichella
University of Iowa, Iowa City, Iowa

Irene Gregory
Langley Research Center, Hampton, Virginia

December 2023

NASA STI Program Report Series

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NTRS Registered and its public interface, the NASA Technical Reports Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA Programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.
- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.
- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.
- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.
- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.
- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

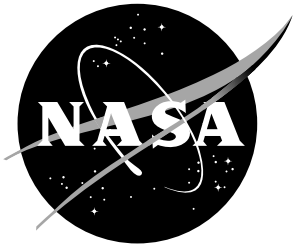
Specialized services also include organizing and publishing research results, distributing specialized research announcements and feeds, providing information desk and personal search support, and enabling data exchange services.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at http://www.sti.nasa.gov
- Help desk contact information:

https://www.sti.nasa.gov/sti-contact-form/ and select the "General" help request type.

NASA/TM–20230013467



On Hermite Interpolation using Bernstein Polynomials for Trajectory Generation

Andrew Patterson
Langley Research Center, Hampton, Virginia

Gage MacLin
University of Iowa, Iowa City, Iowa

Michael Acheson
Langley Research Center, Hampton, Virginia

Camilla Tabasso
University of Iowa, Iowa City, Iowa

Venanzio Cichella
University of Iowa, Iowa City, Iowa

Irene Gregory
Langley Research Center, Hampton, Virginia

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

December 2023

Available from:

NASA STI Program / Mail Stop 148
NASA Langley Research Center
Hampton, VA 23681-2199
Fax: 757-864-6500

Abstract

This work presents a solution to the two-point Hermite interpolation problem using Bernstein polynomials. The Hermite interpolation problem is of particular interest in aerospace applications where boundary conditions for trajectories often specify derivative constraints. In the examples shown, a trajectory will be generated between an initial condition and a final condition. For example, a trajectory is generated that connects an aircraft's current position and velocity with a point on the runway at a desired landing velocity. The numerical stability of the proposed algorithms is analyzed empirically.

1 Introduction

One goal of trajectory generation is to connect the initial and final state of an object with a trajectory that satisfies a set of constraints. In general, creating a trajectory that satisfies arbitrary constraints, e.g., obstacle avoidance, dynamic feasibility, or energy usage, is difficult and requires significant computational effort. However, certain trajectory representations, and their associated algorithms, are more naturally suited to guaranteeing that specific constraints are satisfied.

Bernstein polynomials, or Bézier curves, are a popular choice for trajectory generation because their properties make it simple to guarantee that spatial and dynamic constraints are satisfied for the entire curve [1]. These guarantees make the curves attractive for aeronautics applications, such as those in Refs. [2–4]. Furthermore, by the nature of their construction, the start and end points of the Bernstein polynomial curve are explicitly defined as the first and last parameters of the curve. A growing body of literature, including Refs. [5–9], underscores their utility in optimal motion planning. These polynomials have been demonstrated in several applications, such as target monitoring [10], search and rescue operations [11], and mine counter-measures [12], among others. Researchers in Refs. [13] and [14] propose the use of these polynomials to discretize optimal control problems, specifically for differentially flat systems and more general non-linear systems, respectively. By recasting the motion planning problem as an optimal control problem, the algorithm gains flexibility, allowing for the integration of various costs and constraints tailored to the specific scenario. Additionally, the work in Ref. [15] offers a software implementation leveraging Bernstein polynomial properties for motion planning.

If the only constraints on the terminal condition of the trajectory are spatial, then Bernstein polynomials are a natural choice. However, in many applications, the terminal condition also includes derivative specifications. To satisfy this terminal condition, an optimization procedure would be needed to drive free coefficients of the polynomial to values that satisfy this terminal constraint.

The Hermite interpolation problem, as outlined in Ref. [16], is solved by a parametric curve that satisfies initial and final conditions that specify start and end locations along with n-th order derivative constraints at both of these points. This problem can be solved with a polynomial equation where the parameters of the polynomial are chosen to meet these constraints.

In this work, we wish to retain the benefits of representing trajectories with Bernstein polynomials while initially specifying the curve in terms of the boundary conditions on both positions and derivatives. To this end, we use the properties of Bernstein polynomials to define a linear map between the coefficients of Bernstein polynomials and the boundary conditions of the curve. By inverting this map, we produce a solution to the two-point Hermite interpolation problem in Bernstein polynomial form.

Previous work considering Bernstein polynomial solutions to the Hermite interpolation problem can be found in Refs. [17–19]. These papers are expressly concerned with optimal trajectory generation for dynamical systems and their connection to Bernstein polynomials. While these papers consider the mapping from control points to terminal constraints, they do not make explicit the reverse mapping. The presented work focuses on this reverse mapping: taking a set of terminal constraints and computing the control points necessary for the curve to meet the given constraints. Furthermore, we present example results to demonstrate applicability to trajectory generation problems and experimental analysis of the stability of the mapping using the condition number, which captures the sensitivity of algorithms to small numeric errors in the algorithm input.

The necessary background for the procedure is provided in Section 2. The interpolation equations are derived in Section 3. Section 4 demonstrates the interpolation method for motion planning problems. Finally, the conclusions are presented in Section 5.

2 Background

This section provides an overview of Bernstein polynomials, their definition, and selected properties. It also covers some concepts in numerical stability.

2.1 Bernstein Polynomials

Bernstein polynomials (or Bézier curves) are parametric polynomial curves and are defined on a closed interval, $\zeta \in [0, 1]$ by the equation

$$B(\zeta) = \sum_{k=0}^{n} b_k^n(\zeta) p_k, \tag{1}$$

where n is the degree of the polynomial, p_k is the k^{th} polynomial coefficient, called a control point, and b_k^n is a Bernstein basis polynomial given by the equation

$$b_k^n(\zeta) = \binom{n}{k} (1-\zeta)^{n-k} \zeta^k. \tag{2}$$

An example polynomial is shown in Figure 1. The curve, evaluated along the interval, is shown as the black line. The black diamonds are the polynomial control points, which are associated with specific locations in the parameter space.

There are several properties that make Bernstein polynomials conducive to both the trajectory generation problem and the two-point Hermite interpolation problem. A more complete list of properties can be found in Ref. [20].

The first property considered is the affine change of the independent variable. While this property is often overlooked in the creation of spatial curves, the duration of the curve immediately affects the endpoint derivatives. This scaling property will allow us to capture these changes for the true duration of the curve. Note that the Bernstein basis functions can be defined on an arbitrary interval, $t \in [t_0, t_1] \to \zeta$, with an affine change to the parameter:

$$\zeta(t) = (t - t_0)/(t_1 - t_0). \tag{3}$$

Substituting into the basis functions, we arrive at

$$\bar{b}_k^n(t) = \binom{n}{k} (t_1 - t)^{n-k} (t - t_0)^k / (t_1 - t_0)^n, \tag{4}$$

where the bar represents that the function is defined on an arbitrary interval. Furthermore, the polynomial interpolates between the initial and final control points. From Equation 1, we can verify that $B(0) = p_0$ and $B(1) = p_n$.

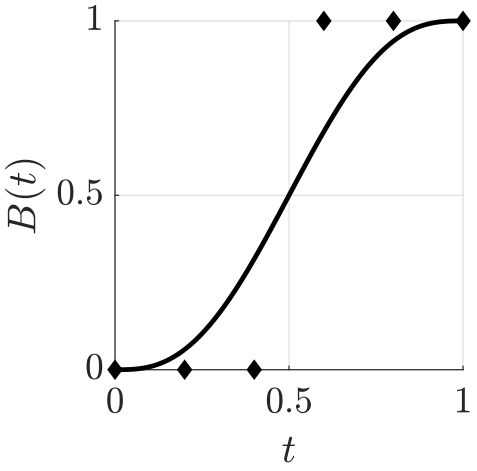


Figure 1: *Bernstein polynomial: Evaluated curve and control points as the solid line and diamonds respectively.*

Finally, the Bernstein polynomial structure is preserved under differentiation. The derivative of a Bernstein polynomial is given by the equation:

$$\frac{\mathrm{d}}{\mathrm{d}\zeta} B(\zeta) = n \sum_{k=0}^{n-1} b_k^{n-1}(\zeta)(p_{k+1} - p_k). \tag{5}$$

This property creates a natural connection between the Hermite interpolation problem, which has derivative constraints, and Bernstein polynomials, which will be used to find a solution.

2.2 Finite Differences

The solution to the two-point Hermite interpolation problem will require derivatives of order greater than one. To find these derivatives, it is convenient to define the forward and backwards difference operators, which are used to compute repeated differences. Difference operators and their inverses are discussed further in Ref. [21]. The matrix forms for these operations are briefly discussed in Appendix A. The forward difference operator, $\Delta^j(x_i)$, is defined recursively as

$$\begin{aligned} \Delta^j(x_i) &= \Delta^{j-1}(x_{i+1}) - \Delta^{j-1}(x_i), \quad j = 1, 2, \dots \\ \Delta^0(x_i) &= x_i, \end{aligned} \tag{6}$$

where x_i is the i-th element of an input vector X and j is the number of forward differences to take. The backwards difference operator is defined recursively as

$$\begin{aligned} \nabla^j(x_i) &= \nabla^{j-1}(x_i) - \nabla^{j-1}(x_{i-1}), \quad j = 1, 2, \dots \\ \nabla^0(x_i) &= x_i. \end{aligned} \tag{7}$$

The difference operators depend on both the element of the vector and the number of differences applied. However, for a vector of length n, only certain vector elements are defined up to $j = n-1$. The forward difference operation is applied to the first element of the vector, x_0, and the backwards difference operation is applied to the final element of the vector, x_n. These are the only elements for which all differences, $j = 0, 1, \dots, n-1$, are well defined. Therefore, the inverse operations are provided only for the associated element of the array. The inverse operation for the forward difference operator, the anti-difference operator, can be defined recursively as follows:

$$\begin{aligned} \Delta^{-j}(y_i) &= \Delta^{j+1}(y_{i+1}) + \Delta^{j+1}(y_i), \quad j = 1, 2, \dots, n-1 \\ \Delta^0(y_0) &= y_0, \end{aligned} \tag{8}$$

for the input vector y. Note that the negative superscript indicates that summation is performed. The backwards difference operator is self inverse for the final element in the array, so we can recursively compute the initial array from the result of the backwards difference operation, in Equation 7, denoted with the vector y:

$$\begin{aligned} \nabla^j(y_i) &= \nabla^{j-1}(y_i) - \nabla^{j-1}(y_{i-1}), \quad j = 1, 2, \dots, n-1 \\ \nabla^0(y_n) &= y_n. \end{aligned} \tag{9}$$

2.3 Numerical Stability

The numerical stability of this transformation at higher orders is of particular importance for aerospace applications, as these problems often require boundary conditions that define initial and final derivative values. For example, a vehicle attempting a landing must move from their current location to the runway while also slowing down from their cruise speed to an appropriate landing speed. With each additional derivative boundary condition that is defined, the order of the polynomial is increased by two to accommodate this constraint. For example, if the initial and final constraints up to the tenth derivative are given, the resulting polynomial will have an order of twenty-one.

To quantify the sensitivity of the proposed interpolation method to small numeric errors in the input, we use the condition number, computed using the ratio of the maximum and minimum singular values of a square matrix, A:

$$\kappa(A) = \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)}. \tag{10}$$

The condition number is used to capture the numerical stability of the Hermite interpolation solution, and will be computed based on a matrix-based implementation of the algorithm. However, the actual numerical stability and condition depend heavily on the implementation details of the algorithm.

Bernstein polynomials are well conditioned for a very large number of parameters, see Ref. [20]. However, the Hermite interpolation solution is expected to be sensitive to input perturbations, since a small change in a high-order derivative, allowed to integrate over a long time period, will lead to a large change in the position of the curve.

3 Two-Point Hermite Interpolation

The objective of the two-point Hermite interpolation problem is to generate a curve that interpolates between two points while meeting derivative constraints at the start and end points. We further require that the interpolating curve be parameterized as a Bernstein polynomial. Therefore, the derivative information at two locations, x_0, $x_1 \in \mathbb{R}$, must be converted into a set of Bernstein polynomial control points, $P \in \mathbb{R}^n$. The boundary constraints are collected in the vector

$$H = \begin{bmatrix} H^0 & | & H^1 \end{bmatrix}^\top = \begin{bmatrix} x_0 & x_0^{(1)} & x_0^{(2)} & \dots & x_0^{(m)} & | & x_1 & x_1^{(1)} & x_1^{(2)} & \dots & x_1^{(m)} \end{bmatrix}^\top, \tag{11}$$

where the superscript on H indicates the boundary index and the parenthetical superscript indicates the order of the derivative constraint at the associated boundary condition. Restated, we wish to find the following mapping, $f : H \in \mathbb{R}^n \to P \in \mathbb{R}^n$, where $n = 2(m+1)$, solving the two-point Hermite interpolation problem for m-th order derivative constraints.

3.1 Derivation

From the definition of the Bernstein polynomial, and its derivatives, we can create a mapping from the set of control points to the boundary conditions of the curve. We demonstrate a method for inverting this mapping to solve for the control points, given the boundary conditions, solving the two-point Hermite interpolation problem.

Recall that the derivative of a Bernstein polynomial is given by:

$$\frac{\mathrm{d}}{\mathrm{d}\zeta} B(\zeta) = n \sum_{k=0}^{n-1} b_k^{n-1}(\zeta)(p_{k+1} - p_k).$$

Since the derivative of the curve is again a Bernstein polynomial one can show that the j^{th} derivative of the curve is given by the equation

$$\frac{\mathrm{d}^j}{\mathrm{d}\zeta^j} B(\zeta) = \frac{n!}{(n-j)!} \sum_{k=0}^{n-j} b_k^{n-j}(\zeta) \Delta^j(p_k). \tag{12}$$

To compute the derivative over an arbitrary parameter interval, Equation 12 can be rewritten as

$$\frac{\mathrm{d}^j}{\mathrm{d}t^j}B(t) = \frac{1}{(t_1 - t_0)^j}\frac{n!}{(n-j)!}\sum_{k=0}^{n-j}\bar{b}_k^{n-j}(t)\Delta^j(p_k), \tag{13}$$

with a scaling factor based on the duration of the parameter interval.

The derivative equations provide a closed form relationship between the Bernstein parameters and the derivatives of the curve. This relationship can be used to define the initial elements of H in terms of Bernstein control points:

$$H^0 = [x_0 \quad x_0^{(1)} \quad \dots \quad x_0^{(m)}]^\top = [p_0 \quad n\frac{\Delta^1(p_0)}{(t_1 - t_0)} \quad \dots \quad \frac{n!}{(n-m)!}\frac{\Delta^m(p_0)}{(t_1 - t_0)^m} \quad]^\top. \tag{14}$$

The final elements of H require more work to construct since they are defined in terms of the final control point in the series, requiring the backwards difference operator. Then the final values of H can be computed as

$$H^1 = [x_1 \quad x_1^{(1)} \quad \dots \quad x_1^{(m)}]^\top = [p_n \quad n\frac{\nabla^1(p_n)}{(t_1 - t_0)} \quad \dots \quad \frac{n!}{(n-m)!}\frac{\nabla^m(p_n)}{(t_1 - t_0)^m} \quad]^\top. \tag{15}$$

Taken together, these equations can be solved for the control points needed to solve the interpolation problem. First, note the triangular structure of the problem. The value of x_0 explicitly constrains the value of p_0. Then the only free variable in the equality $x_0^{(1)} = n\Delta^1(p_0) = n(p_1 - p_0)$ is p_1. These equations can be recursively solved in this manner using the inverse recursions established in Section 2. That is, let

$$\begin{aligned} Q^0 &= \left[x_0 \quad \tfrac{t_1-t_0}{n}x_0^{(1)} \quad \dots \quad (t_1 - t_0)^m\,\tfrac{(n-m)!}{n!}x_0^{(m)}\right]^\top, \quad \text{then} \\ \left[p_0 \quad p_1 \quad \dots \quad p_{m+1}\right]^\top &= \left[q_0^0 \quad \Delta^{-1}(q_0^0) \quad \dots \quad \Delta^{-m}(q_0^0)\right]^\top, \end{aligned} \tag{16}$$

where q_0^0 is the first element of the vector Q^0. The terminal control points are determined in a similar fashion. Let

$$\begin{aligned} Q^1 &= \left[(t_1 - t_0)^m\,\tfrac{(n-m)!}{n!}x_1^{(m)} \quad \dots \quad \tfrac{t_1-t_0}{n}x_1^{(1)} \quad x_1\right]^\top, \quad \text{then} \\ \left[p_{m+2} \quad \dots \quad p_{2m} \quad p_{2m+1}\right]^\top &= \left[\nabla^m(q_n^1) \quad \dots \quad \nabla^1(q_n^1) \quad (q_n^1)\right]^\top, \end{aligned} \tag{17}$$

where q_n^1 is the last element of the vector Q^1. Note that to satisfy the terminal conditions, the control points are computed in reverse order. Instead of starting from a fixed point and computing the sum to find the next point, we start from an ending point and work backwards, using the backwards difference operator. Taken together, we have that

$$\begin{aligned} P &= \left[p_0 \quad p_1 \quad \dots \quad p_{m+1} \quad | \quad p_{m+2} \quad \dots \quad p_{2m} \quad p_{2m+1}\right]^\top \\ &= \left[q_0^0 \quad \Delta^{-1}(q_0^0) \quad \dots \quad \Delta^{-m}(q_0^0) \quad | \quad \nabla^m(q_n^1) \dots \quad \nabla^1(q_n^1) \quad q_n^1\right]^\top. \end{aligned} \tag{18}$$

4 Trajectory Generation

This section presents the application of the interpolation solution to trajectory generation problems using a series of examples.

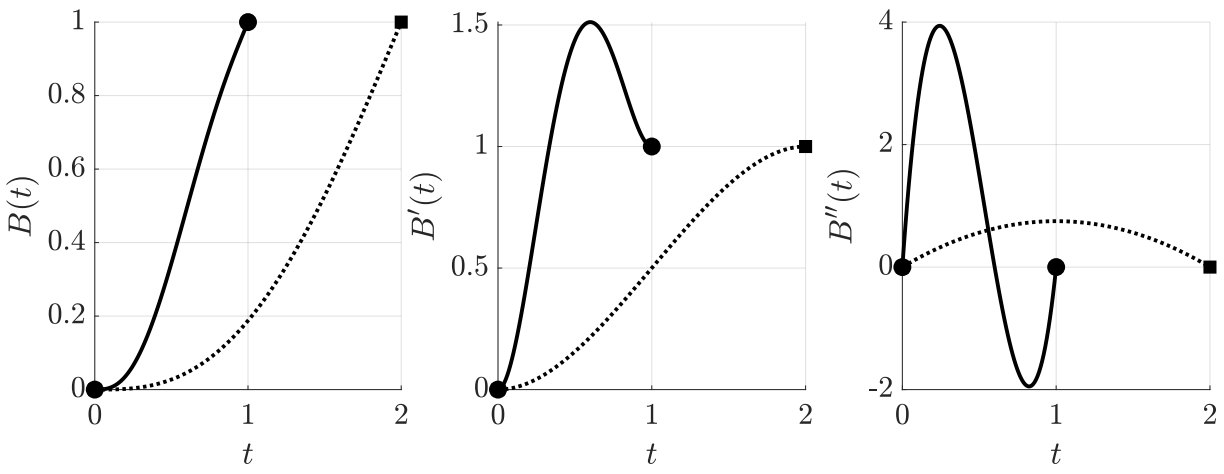


Figure 2: *Two interpolation problems with different duration but the same terminal constraints. The black circles and squares in each plot show the position, velocity, and acceleration constraints, left to right, respectively. The curves are the Bernstein polynomial evaluation at each time.*

The first example is shown in Figure 2. Here, $H = [0\ 0\ 0\ |\ 1\ 1\ 0]^\top$, indicating that the desired polynomial should have zero initial position, velocity, and acceleration but a terminal position and velocity of one, and acceleration of zero. These constraints are shown for two different durations in Figure 2, with the circles indicating the constraints at a terminal time of one, and the squares indicating the constraint at a terminal time of two. Using Equation (18), the Bernstein polynomial meeting these constraints can be computed. The solution curves, evaluated over their duration, are shown as black and dotted lines for their respective durations. The control point vector, from Equation (18), defining the polynomial for the faster curve is $[0\ 0\ 0\ 0.6\ 0.8\ 1]^\top$, and the control point vector for the slower curve is $[0\ 0\ 0\ 0.2\ 0.6\ 1]^\top$.

Both curves meet their associated constraints but only when connected to the duration of the underlying parameter, t. This coupling arises from the derivative constraints being a function of the duration given to satisfy the constraints. In this example, the lower duration curve must have a faster average velocity to meet the position constraints and subsequently must have larger acceleration (and deceleration) to realize that velocity change.

The duration dependence of the solution is further demonstrated in Figure 3, where fixed terminal constraints but different times lead to very different spatial behavior. In this figure, the two-dimensional curve is constrained with identical start and end speeds but different directions. When the given duration is appropriately chosen, the curve approximates a circular turn, given by the solid black line. Reducing the duration leads to an almost straight-line connection, shown as a dashed line, with high velocities and accelerations needed to meet the temporal constraint. Finally, the dotted line shows the effect of an increased duration, the connecting curve loops away from

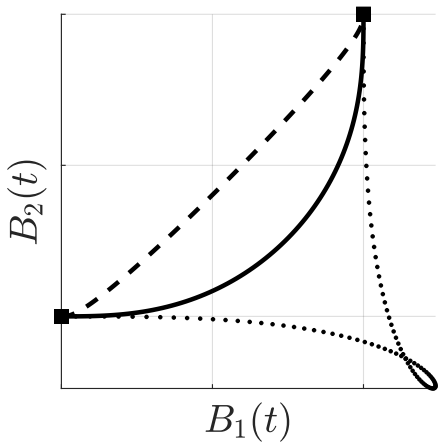


Figure 3: *Two-dimensional curve, with durations scaled by 0.2, 1, and 3 times corresponding to the dashed, solid, and dotted lines respectively.*

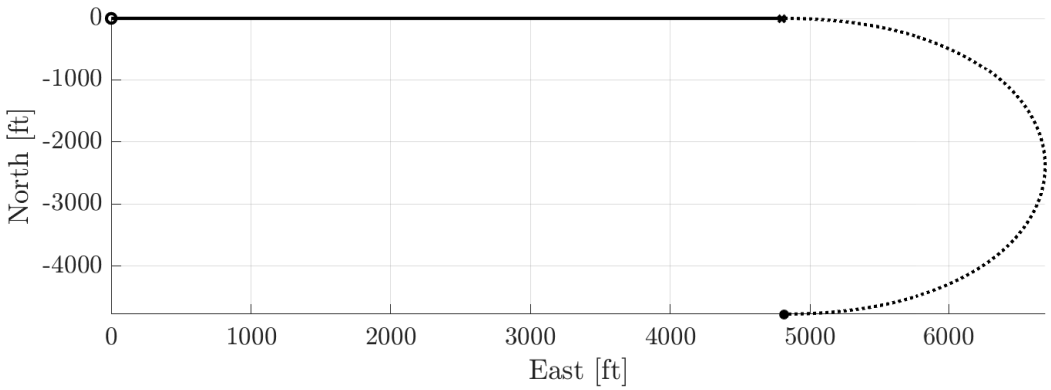


Figure 4: *Landing example. The solid line is a runway. The dotted line is a trajectory that connects a vehicle's initial position, velocity, and acceleration, with a desired landing speed at the runway threshold.*

the terminal point.

In certain applications, the duration of desired trajectories is known beforehand. Consider the landing maneuver shown in Figure 4. In this case, the terminal position is fixed as a point on the runway, and terminal velocity is fixed at landing speed in the direction of the runway. The initial condition is set as the vehicle's current position and velocity for each dimension. The duration is set to correspond to a standard-rate turn, for a duration of 1 minute for a 180° turn. Using the interpolation solution immediately produces a Bernstein polynomial that creates a feasible landing trajectory.

For many path planning applications, only a few derivatives need to be constrained. However, it is important to consider the limits of the numerical stability of the presented transform when a large number of derivatives are needed. Figure 5 shows the condition number of the matrix form of the presented transform, depending on the number of derivatives specified. As mentioned previously, the condition number captures the sensitivity of the matrix transform output to small input errors. From this figure, we see that the condition number grows with the number of derivatives, implying that small input errors would be scaled proportionally to increased condition number. These condition numbers are computed for a unit parameter interval. Increasing or decreasing the interval will have an impact on the condition number. It is also important to note that this condition number does not limit the number of control points used in subsequent planning operations. While this work exactly solves for a Bernstein polynomial that satisfies terminal constraints, higher dimensional curves can also be used. To this end, a Bernstein polynomial can be degree elevated, that is, the curve can be exactly represented by a higher order curve with more control points. These additional control points can be added to increase the degrees of freedom in subsequent optimization procedures.

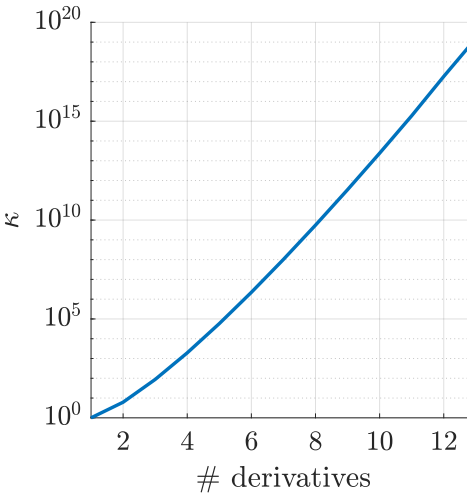


Figure 5: *Condition number vs number of derivatives used to solve the Hermite interpolation problem.*

5 Conclusions

In this work, we present an explicit solution to the two-point Hermite interpolation problem using Bernstein polynomials. The Hermite interpolation problem is of particular interest in aerospace applications where initial and terminal constraints often include derivative constraints. The Bernstein polynomial solution enables the generated parametric curve to be evaluated using a large body of existing algorithms, to check constraints throughout the trajectory.

Future improvements will consider using degree elevated curves to increase the number degrees of freedom for further trajectory optimization and will address partial constraints, where some time and some terminal constraints may be left free.

Acknowledgments

This work is funded by the NASA Transformational Tools and Technologies project and NASA Grant #80NSSC23M0117.

References

1. Lakshmanan, A., Patterson, A., Cichella, V., and Hovakimyan, N., "Proximity Queries for Absolutely Continuous Parametric Curves," *Robotics: Science and Systems XV*, Freiburg im Breisgau, Germany, 2019, p. 42.

2. Patterson, A., Ackerman, K. A., Hovakimyan, N., and Gregory, I. M., "Trajectory Generation for Distributed Electric Propulsion Vehicles with Propeller Synchronization," *AIAA SciTech*, Virtual, 2021, pp. AIAA 2021–0586.

3. Ackerman, K. A., and Gregory, I. M., "Comparison of Acoustic Models and Trajectory Generation Methods for an Acoustically-Aware Aircraft," *AIAA SciTech*, National Harbor, MD, USA, 2023, pp. AIAA 2023–2543.

4. Houghton, M. D., Acheson, M. J., Patterson, A., Oshin, A., and Gregory, I. M., "Combined Bernstein Polynomial Optimal Reciprocal Collision Avoidance Differential Dynamic Programming for Trajectory Replanning and Collision Avoidance for UAM Vehicles," *AIAA SciTech*, National Harbor, MD, USA, 2023, pp. AIAA 2023–2544.

5. Choe, R., Puig-Navarro, J., Cichella, V., Xargay, E., and Hovakimyan, N., "Cooperative Trajectory Generation Using Pythagorean Hodograph Bézier Curves," *Journal of Guidance, Control, and Dynamics*, 2016, pp. 1744–1763.

6. Yang, L., Song, D., Xiao, J., Han, J., Yang, L., and Cao, Y., "Generation of Dynamically Feasible and Collision Free Trajectory by Applying Six-order Bézier Curve and Local Optimal Reshaping," *Intelligent Robots and Systems (IROS)*, IEEE, 2015, pp. 643–648.

7. Strano, B., Furci, M., and Marconi, L., "Kinodynamic Optimal Trajectories Generation in Cluttered Environments using Bernstein Polynomials," *Mechatronics*, Vol. 92, 2023. Article no. 102969.

8. Park, B., Lee, Y.-C., and Han, W. Y., "Trajectory Generation Method using Bézier Spiral Curves for High-speed On-road Autonomous Vehicles," *Conference on Automation Science and Engineering (CASE)*, 2014, pp. 927–932.

9. Ricciardi, L. A., and Vasile, M., "Direct Transcription of Optimal Control Problems with Finite Elements on Bernstein Basis," *Journal of Guidance, Control, and Dynamics*, 2018, pp. 1–15.

10. Kielas-Jensen, C., Cichella, V., Casbeer, D., Manyam, S. G., and Weintraub, I., "Persistent Monitoring by Multiple Unmanned Aerial Vehicles Using Bernstein Polynomials," *Journal of Optimization Theory and Applications*, Vol. 191, No. 2-3, 2021, pp. 899–916.

11. Tabasso, C., Mimmo, N., Cichella, V., and Marconi, L., "Optimal Motion Planning for Localization of Avalanche Victims by Multiple UAVs," *IEEE Control Systems Letters*, Vol. 5, No. 6, 2021, pp. 2054–2059.

12. Kragelund, S., and Kaminer, I., "Clandestine Mine Countermeasures Optimization for Autonomy and Risk Assessment," Tech. rep., Naval Postgraduate School, Monterey, CA, 2022.

13. Cichella, V., Kaminer, I., Walton, C., and Hovakimyan, N., "Optimal Motion Planning for Differentially Flat Systems Using Bernstein Approximation," *IEEE Control Systems Letters*, Vol. 2, No. 1, 2018, pp. 181–186.

14. Cichella, V., Kaminer, I., Walton, C., Hovakimyan, N., and Pascoal, A. M., "Optimal Multivehicle Motion Planning using Bernstein Approximants," *IEEE Transactions on Automatic Control*, Vol. 66, No. 4, 2020, pp. 1453–1467.

15. Kielas-Jensen, C., Cichella, V., Berry, T., Kaminer, I., Walton, C., and Pascoal, A., "Bernstein Polynomial-Based Method for Solving Optimal Trajectory Generation Problems," *Sensors*, Vol. 22, No. 5, 2022, p. 1869.

16. Davis, P. J., *Interpolation and Approximation*, 1st ed., Dover, New York, NY, USA, 1975.

17. Egerstedt, M. B., and Martin, C. F., "A Note on the Connection Between Bezier Curves and Linear Optimal Control," *IEEE Transactions on Automatic Control*, Vol. 49, No. 10, 2004, pp. 1728–1731.

18. Lee, S., and Kim, Y., "Optimal Output Trajectory Shaping Using Bézier Curves," *Journal of Guidance, Control, and Dynamics*, Vol. 44, No. 5, 2021, pp. 1027–1035.

19. Zhimin, Z., Tomlinson, J., and Martin, C., "Splines and Linear Control Theory," *Acta Applicandae Mathematicae*, Vol. 49, No. 1, 1997, pp. 1–34.

20. Farouki, R. T., "The Bernstein Polynomial Basis: A Centennial Retrospective," *Computer Aided Geometric Design*, Vol. 29, No. 6, 2012, pp. 379–419.

21. Elaydi, S., *An Introduction to Difference Equations*, 3rd ed., Springer, New York, NY, USA, 2005.

22. Forgues, D., and Hasler, M. F., "Binomial Transform," , 2014. URL https://oeis.org/wiki/Binomial_transform, accessed: 2023-10-06 From the On-Line Encyclopedia of Integer Sequences (OEIS) wiki.

23. Donaghey, R., "Binomial Self-Inverse Sequences and Tangent Coefficients," *Journal of Combinatorial Theory*, Vol. 21, 1976, pp. 155–163.

Appendix A

Matrix Forms

The matrix forms of the difference equations are used for condition number analysis and to connect them to other linear operators. The algorithms used in this paper use the recursive forms to avoid matrix multiplication. The forward difference operation as defined in Equation 6, and computed for the first element is given by the equation:

$$Y^F = \begin{bmatrix} \Delta^0(x_0) \\ \Delta^1(x_0) \\ \Delta^2(x_0) \\ \Delta^3(x_0) \\ \vdots \\ \Delta^n(x_0) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ -1 & 1 & 0 & 0 & & \\ 1 & -2 & 1 & 0 & & \\ -1 & 3 & -3 & 1 & & \\ \vdots & & & & \ddots & \\ (-1)^n & & & & & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}, \tag{A1}$$

where the superscript indicates the resulting vector is the output of the forward difference matrix. The inverse operation, given in Equation 8, can then be represented in matrix form:

$$X = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & 0 & & \\ 1 & 2 & 1 & 0 & & \\ 1 & 3 & 3 & 1 & & \\ \vdots & & & & \ddots & \\ 1 & & & & & 1 \end{bmatrix} \begin{bmatrix} y_0^F \\ y_1^F \\ y_2^F \\ y_3^F \\ \vdots \\ y_n^F \end{bmatrix}. \tag{A2}$$

Notice the relationship to Pascal's triangle of binomial coefficients. These matrices are called the inverse binomial transform and binomial transform as defined in Ref. [22].

The backwards difference operation as defined in Equation 7, and computed for the last element in a vector is given by the equation:

$$Y^B = \begin{bmatrix} \nabla^0(x_n) \\ \nabla^1(x_n) \\ \nabla^2(x_n) \\ \nabla^3(x_n) \\ \vdots \\ \nabla^n(x_n) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & -1 & 0 & 0 & & \\ 1 & -2 & 1 & 0 & & \\ 1 & -3 & 3 & -1 & & \\ \vdots & & & & \ddots & \\ 1 & & & & & (-1)^n \end{bmatrix} \begin{bmatrix} x_n \\ x_{n-1} \\ x_{n-2} \\ x_{n-3} \\ \vdots \\ x_0 \end{bmatrix}, \tag{A3}$$

where the superscript indicates the resulting vector is the output of the backwards difference matrix. This matrix operation may also be referred to as the binomial transform, or Euler Transform, and is discussed in Ref. [23]. This transform is self inverse.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
01-12-2023	Technical Memorandum	

4. TITLE AND SUBTITLE
On Hermite Interpolation using Bernstein Polynomials for Trajectory Generation

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Andrew Patterson
Gage MacLin
Michael Acheson
Camilla Tabasso
Venanzio Cichella
Irene Gregory

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
xxxxxxxxxxxxxx

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, Virginia 23681-2199

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSOR/MONITOR'S ACRONYM(S)
NASA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
NASA/TM–20230013467

12. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified-Unlimited
Subject Category
Availability: NASA STI Program (757) 864-9658

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This work presents a solution to the two-point Hermite interpolation problem using Bernstein polynomials. The Hermite interpolation problem is of particular interest in aerospace applications where boundary conditions for trajectories often specify derivative constraints. In the examples shown, a trajectory will be generated between an initial condition and a final condition. For example, a trajectory is generated that connects an aircraft's current position and velocity with a point on the runway at a desired landing velocity. The numerical stability of the proposed algorithms is analyzed empirically.

15. SUBJECT TERMS
Bezier curves, Hermite interpolation, Bernstein polynomials, Trajectory generation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			HQ-STI-infodesk@mail.nasa.gov
U	U	U	UU	16	19b. TELEPHONE NUMBER *(Include area code)* (757) 864-9658