Barycentric Lagrange Interpolation
As discussed by Jean-Paul Berrut and Lloyd N. Trefethen (2004)

Maximilian Jentzsch
Math 56 Final Project, Spring 2014, Prof. Barnett

Abstract

This text discusses barycentric Lagrange interpolation based on the SIAM REVIEW article of Jean-Paul Berrut and Lloyd N. Trefethen [1]. It also offers additional background information, as well as some MATLAB demonstrations.

Interpolation

Given a set D_n of $n+1$ nodes x_j with corresponding values f_j where $j = 0,\dots,n$, we aim to construct the polynomial that satisfies

$$p(x_j) = f_j \; j = 0,\dots,n$$

i.e. the polynomial interpolates D_n. Note that the f_j do not necessarily have to correspond to a function.

Theorem 1. *There exists a unique polynomial $p_n(x) = p_{01\dots n}(x)$ of degree less than or equal to n interpolating D_n [3]*

There are many different kinds of interpolation; here we focus on Lagrange interpolation.

Lagrange interpolation

This data set can be interpolated by the Lagrange form of the *interpolation polynomial* [3],

$$p_{01\dots n}(x) = \sum_{j=0}^{n} l_j(x) f(x_j), \text{ where} \tag{0.1}$$

$$l_j(x) = \frac{\prod_{k=0,k\neq j}^{n}(x - x_k)}{\prod_{k=0,k\neq j}^{n}(x_j - x_k)}, \tag{0.2}$$

also called the *Lagrangian cardinal* functions [3]. These satisfy, $\forall i, j = 0, 1 \dots n$:

$$l_i(x_j) = \delta_{ij} = \begin{cases} 0 & if\ i \neq j \\ 1 & if\ i = j \end{cases}$$

$$p_{01\dots n}(x_i) = f_i$$

Issues with Lagrange Interpolation One must note that there are several issues with Lagrange's formula. Here we shall focus on two specific issues:

1. Each evaluation of $p(x)$ requires $O(n^2)$ additions and multiplications.
2. Adding a new data pair (x_{n+1}, f_{n+1}) requires an entirely new computation of every l_j.
3. The computation can be numerically unstable.

This gave rise to recurrence forumulae such as Neville's and Newton's formula, but these will not be discussed here. Instead, we shall focus on an improved Lagrange formula using barycentric weights.

Improved Lagrange Formula

Note that the numerator of (0.2) can be written as

$$l(x) = (x - x_0)(x - x_1)\dots(x - x_n)$$

Then define the *barycentric weights* by

$$w_j = \frac{1}{\prod_{k \neq j}(x_j - x_k)},\ j = 0, \dots, n \tag{0.3}$$

Now, l_j can be written as

$$l_j(x) = l(x) \sum_{j=0}^{n} \frac{w_j}{x - x_j}$$

which yields the new form of (0.1):

$$p(x) = l(x) \sum_{j=0}^{n} \frac{w_j}{x - x_j} f_j \tag{0.4}$$

This improved formula now requires $O(n^2)$ floating point operations (flops) to calculate w_j: $n\, subtractions \times n\, multiplications + one\ division$. After that, evaluating p, only requires $O(n)$ flops.

In addition, this formula can now easily be updated with a new Data set (x_{n+1}, y_{n+1}) just by dividing each w_j from (0.3) by $(x_j - x_{n+1})$, and then computing w_j using (0.3). The addition of a new Data set therefore requires a mere $2n+2$ flops, instead of an entire recalculation of all l_j!

The Barycentric Formula

Equation (0.4) can still be written in an even nicer form. Dividing by one will achieve this result. The interpolant of the constant function 1 is itself. Plugging 1 into (0.4) yields

$$1 = \sum_{j=0}^{n} l_j(x) = l(x) \sum_{j=0}^{n} \frac{w_j}{x - x_j}.$$

Dividing (0.4) by the expression above, $l(x)$ cancels and gives the so-called *barycentric formula* for p:

$$p(x) = \frac{\sum_{j=0}^{n} \frac{w_j}{x-x_j} f_j}{\sum_{j=0}^{n} \frac{w_j}{x-x_j}} \tag{0.5}$$

The barycentric formula is still a Lagrange formula where the weights w_j can be updated with a new data pair (x_{n+1}, y_{n+1}) using $O(n)$ flops. Apart from the factor of f_j, the w_j in the numerator exactly the same as the ones in the denominator. Therefore, any common factor in all of the weights can be canceled without affecting p. For some node distributions, this allows for explicit formulae to calculate the w_j [1].

Using equidistant nodes for the interpolation, the barycentric weights w_j can be written in an explicit form. Let the spacing between the nodes be $h = \frac{2}{n}$ on the interval $[-1,1]$. The weights can be directly calculated by

$$w_j = (-1)^j \binom{n}{j}. \tag{0.6}$$

The use of equidistant nodes has one major disadvantage, discussed in the next section.

Runge Phenomenon

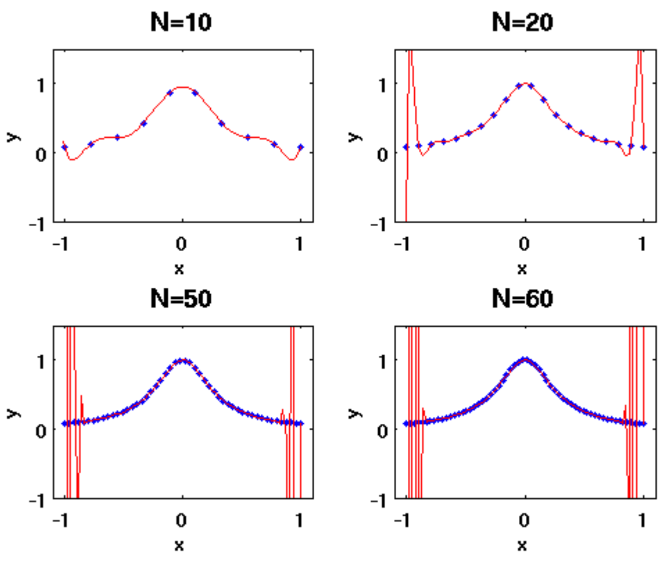
For large n, the different weights w_j vary by exponentially large factors [1]. This makes polynomial interpolation with equidistant points *ill-conditioned*. Figures 0.1a and 0.1b demonstrate this problem. Here, the function $\frac{1}{1+12x^2}$ was interpolated via barycentric Lagrange interpolation (See *RungeDemo.m* for more information).

Towards the "edges" of the interpolation, the approximations do not converge with increasing N. The maximum error eventually grows exponentially (Figure 0.1b). It can be shown that the error can grow as fast as 2^N [4]. This phenomenon, which is more extreme than the Gibbs phenomenon, is called the *Runge phenomenon*.

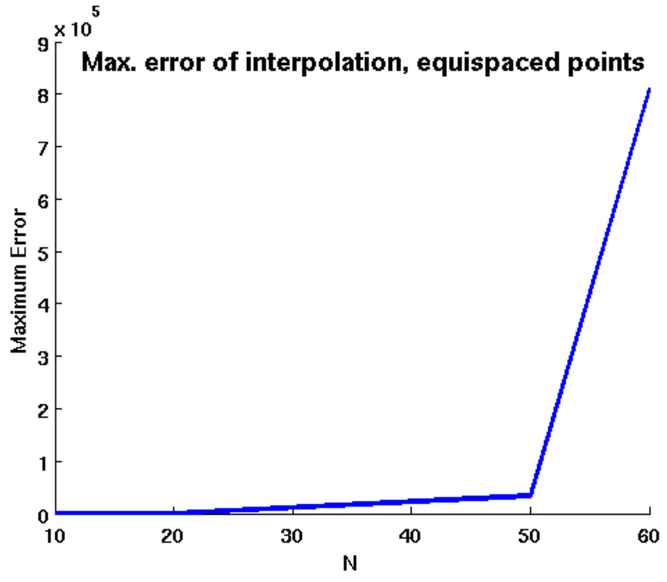
Chebyshev Points Figure 0.1a shows that the interpolation works well in the center, but fails at the "edges". To overcome this problem, unevenly distributed interpolation points must be used. There are different types of such point distributions, but here we shall focus on the most common and simplest kind, the *Chebyshev points*. These have a density $\sim \frac{N}{\pi\sqrt{1-x^2}}$, so there will be more points at the towards the ends, and less points in the center. The Chebyshev points of the *second kind* can be computed by

$$x_j = cos\frac{j\pi}{n},\ j = 0, \ldots, n \tag{0.7}$$

In Figure 0.2, Chebyshev points are used to interpolate the same function. There is no Runge phenomenon.



(a) Interpolation using equidistant nodes



(b) Error of interpolation using equidistant nodes

Figure 0.1: Demonstration of the Runge phenomenon

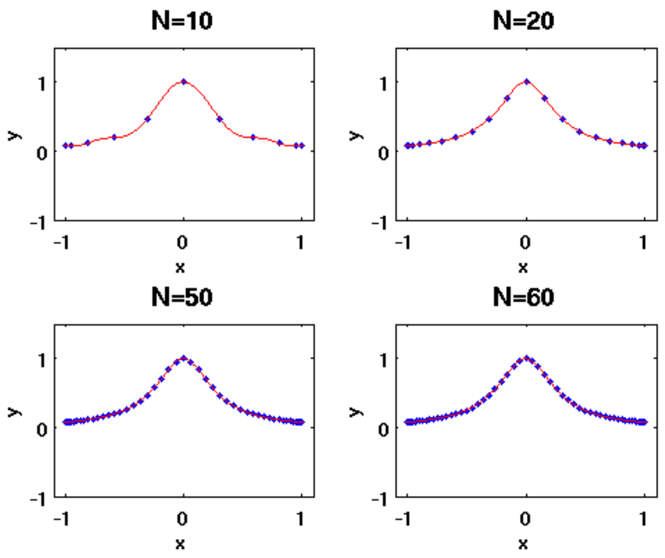


Figure 0.2: Overcoming the Runge Phenomenon with Chebyshev points

The weights w_j for the barycentric formula (0.5) can also be calculated explicitly when using Chebyshev points [1]:

$$w_j = (-1)^j \delta_j, \;\; \delta_j = \begin{cases} \frac{1}{2} & when\ j = 0\ or\ j = n \\ 1 & otherwise. \end{cases} \tag{0.8}$$

Every part in the interpolation now only requires $O(n)$ steps.

Convergence Rates of Smooth Functions

Computation complexity is important, but whether or how fast the interpolations converges to the function is at least as important.

Let f be analytic on an inside an ellipse in the complex plane with foci [-1,1] and axis lengths 2L and 2l. When Chebyshev points are used, the interpolant converges exponentially as $N \rightarrow \infty$ [4]. In addition, the interpolants p_N satisfy the error estimate

$$max_{x\varepsilon[-1,1]} |f(x) - p_N(x)| \leq CK^{-N}$$

for some constants C and K>1 [1, 2, p. 508, p. 173]. If the conditions above are satisfied, then $K = L + l$, and K denotes the convergence rate. Note that the convergence rate depends on the poles of f, and that a larger region of analyticity also results in a higher convergence rate [1]. Figure shows the convergence of several functions with increasing N. We will also calculate the convergence rate, which we can observe to be the slope of the corresponding graphs.

1. $f(x) = \frac{exp(x)}{cos(x)}$: $K = \frac{\pi}{2} + \sqrt{\frac{\pi^2}{4} - 1} \approx 2.7822.$

   Note that there is a mistake in [1], where the term under the square root is $\pi^2 - 1$.

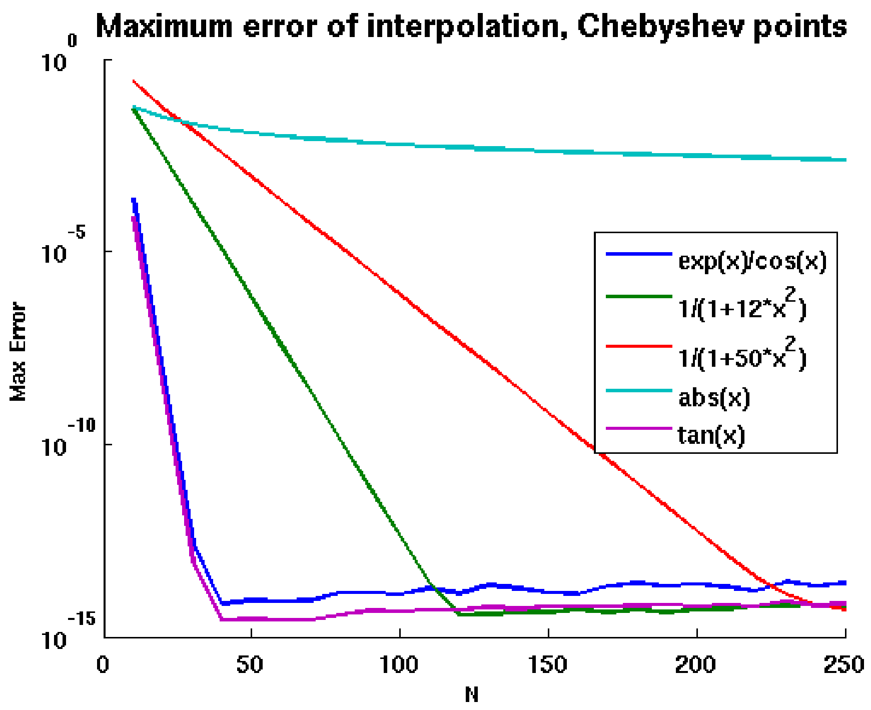


Figure 0.3: Convergence of different functions

2. $f(x) = \frac{1}{1+12x^2} : K = \frac{1}{\sqrt{12}} + \sqrt{\frac{13}{12}} \approx 1.3295$
3. $f(x) = \frac{1}{1+50x^2} : K = \frac{1}{\sqrt{50}} + \sqrt{\frac{51}{50}} \approx 1.1514$
4. $f(x) = |x|$ is not analytic on [-1,1] because it is not differentiable at $x = 0$. The interpolation does not converge exponentially.

Convergence rate of |x| and further exploration

It is not surprising that $|x|$ did not converge as rapidly as the other functions tested since analyticity was a prerequisite for fast convergence. In *Approximation Theory and Approximation Praxis* [5], Trefethen discusses the convergence rate of $|x|$. Using the concept of *bounded variation*, the algebraic convergence rate of $|x|$ was calculated (and observed) to be $\frac{1}{n}$. However, this was not explicitly for barycentric Lagrange interpolation, but for Chebyshev interpolants.

The convergence of $|x|$ using barycentric Lagrange interpolation also seems to be algebraic (see Figure 0.4), but by far not as fast. In addition, the maximum error always seems to come from one particular area of the interpolant, which is not at $x = 0$. See supplemental MATLAB codes and figures for some experimentation.

An additional area that would be very interesting to explore would be combining piecewise barycentric Lagrange interpolants to improve the interpolation of functions such as $|x|$.

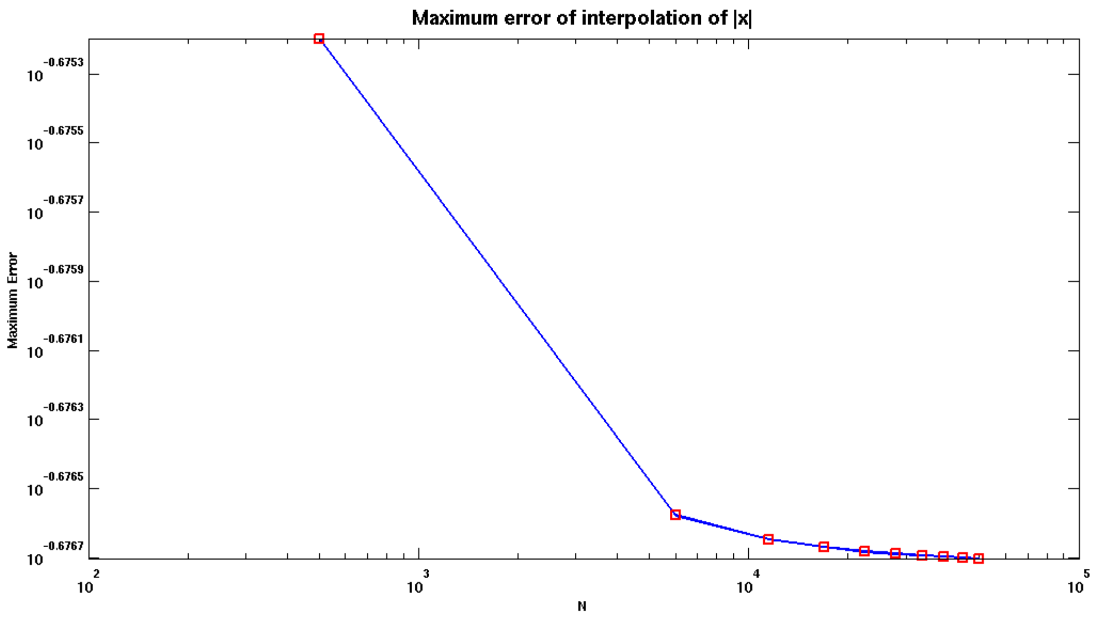


Figure 0.4: Absolute Error of the interpolation of |x|

References

[1] Jean-Paul Berrut and Lloyd N. Trefethen. Barycentric Lagrange Interpolation. *SIAM REVIEW*, 46(3):501–517, 2004.

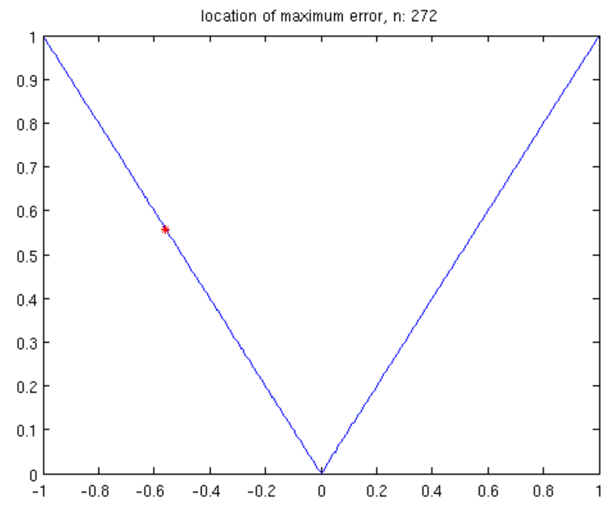
[2] Bengt Fornberg. *A Practical Guide to Pseudospectral Methods*. Cambridge University Press, 1996.

[3] Nabil R. Nassif and Dolly K. Fayyad. *Numerical Analysis and Scientific Computing*. CRC Press, 2014.

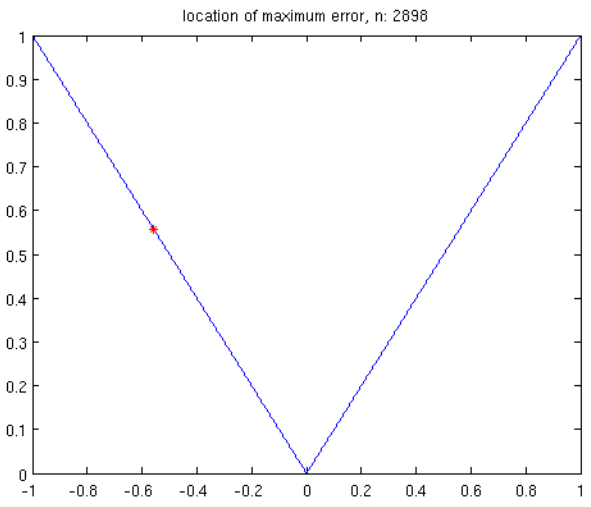
[4] Lloyd N. Trefethen. *Spectral Methods in Matlab*, chapter 5. Polynomial Interplation and Clustered Grids, pages 41–50. SIAM, 2000.

[5] Lloyd N. Trefethen. *Approximation Theory and Approximation Praxis*, chapter 7. Convergence for differentiable functions. SIAM, 2013.

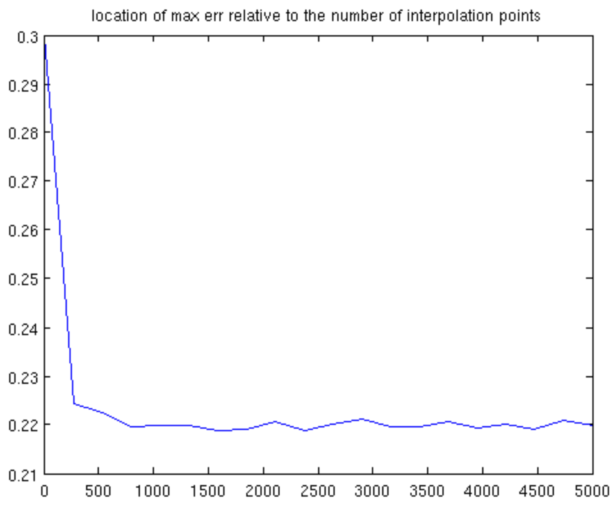
Supplemental Figures: exploration of |x| convergence.



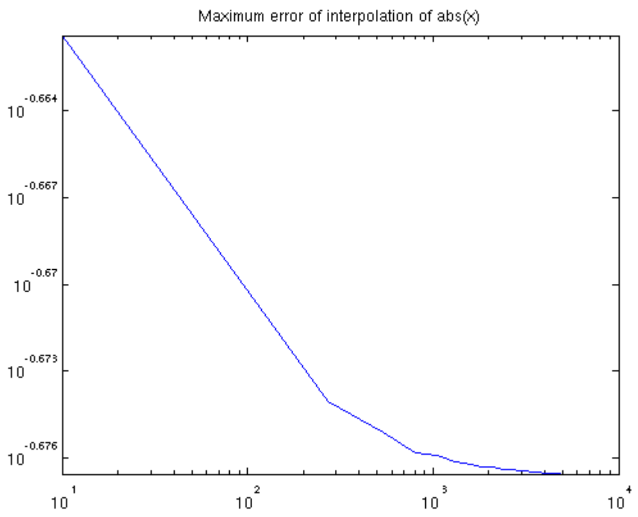
(a)



(b)



(c)



(d)

Figure 0.5: |x| exploration