Polynomial interpolants of nonperiodic functions

Alex Townsend

May 5, 2015

Continuous functions can be hard to deal with. They may only allow pointwise evaluation such as $f(x) = \{\text{wind speed in Boston at location } x\}$ or involve complicated algebraic expressions (calculate the formula for $\int_0^x \log(2+s)^3 \log(3+s) s^3 ds$). We want life to be easy, practical, and efficient. Replacing functions by polynomials can be a great idea.

Problem. Continuous functions are far wilder than polynomials. Can we really replace every continuous function by a polynomial? Surprizingly, Weierstrass tells us we can.

> **Weierstrass' fact:** Given a continuous function f and an accuracy goal of $\epsilon > 0$, there is a polynomial p such that
>
> $$\max_{x \in [-1,1]} |f(x) - p(x)| < \epsilon.$$

This is a truly remarkable fact and means polynomials are super-useful. But, Weierstrass did not tell us how to construct such a p. We will use interpolation.

We have two choices to make: (1) The interpolation points, and (2) The polynomial basis to represent the interpolant.

Interpolation points

We will select the Chebyshev points:

$$x_j = \cos\left(\frac{(N-j)\pi}{N}\right), \qquad 0 \leq j \leq N.$$

These points cluster at ± 1 just right.

Polynomial basis

The monomial basis is seriously bad news. Instead, we will select the Chebyshev polynomial basis, where

$$T_k(x) = \cos(k \cos^{-1}(x)), \qquad k \geq 0.$$

The formula for $T_k(x)$ does not look like a polynomial but it really is one!

OK. So we want to interpolant. We want to find $c_0, \dots, c_N$ such that

$$f(x_j) = p(x_j) = \sum_{k=0}^{N} c_k T_k(x_j), \qquad 0 \le j \le N.$$

That is, plugging in all the formulas,

$$f(x_{N-j}) = \sum_{k=0}^{N} c_k \cos(k \cos^{-1}(\cos(\tfrac{j\pi}{N}))) = \sum_{k=0}^{N} c_k \cos(\tfrac{kj\pi}{N}) = \sum_{k=0}^{N} c_k \cos(2\pi k \tfrac{j}{2N}),$$

for $0 \le j \le N$. This task is closely related to that found in signal processing. It can also be written in terms of the DFT (you do not need to know how to rewrite it) and calculated in $\mathcal{O}(N \log N)$ operations. Hooary for the FFT.

Here is a degree 1,000,000 polynomial interpolant of a truly wild function. (It does not look much like a polynomial, but Weierstrass's fact tells us that it does not have to.)