APPROXIMATION THEORY AND PRACTICE

Nick Trefethen, Computing Laboratory

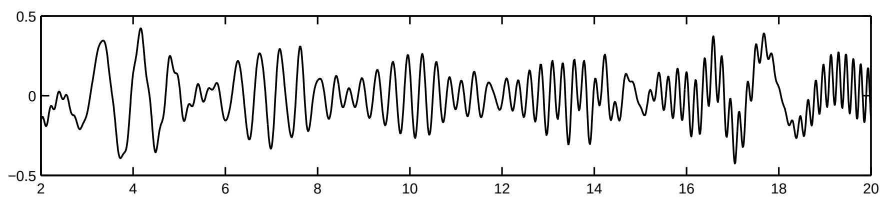
Did you know that it is perfectly feasible to interpolate 1,000,001 data values by a polynomial of degree 1,000,000? The power of high-order polynomial interpolation is being exploited by the chebfun project based in the Numerical Analysis Group.

Two tricks are needed to make the process practical. First, the interpolation points must be clustered near the boundary, and as you might guess from the name, we use use Chebyshev points: for the interval $[-1, 1]$, $x_j = \cos(j\pi/n)$, $0 \le j \le n$. Second, the interpolant must be evaluated by the barycentric formula published by Salzer in 1972. The combination of these ideas leads to methods as efficient and robust as the much more famous Fourier techniques for periodic problems.

The chebfun project is a software system built on these facts and on the remarkable way in which Matlab has grown to codify generations of advances in numerical linear algebra. The system overloads Matlab commands for vectors and matrices so that they apply to functions and operators, with everything implemented by Chebyshev interpolants. For example, suppose we type

```
f = chebfun('sin(x^2)*besselj(0,x)+sin(x)*besselj(1,(20-x)^2)',[2 20])
```

The resulting chebfun `f` matches $\sin(x^2)J_0(x) + \sin(x)J_1((20-x)^2)$ on $[2, 20]$ to about 15 digits of precision and can be manipulated by dozens of familiar Matlab commands. We can plot it with `plot(f)`:



The command `length(f)` reveals that `f` is actually a polynomial of degreee 492. Here are its maximum, integral, total varation, and tenth zero counted from the left, all computed to about 15-digit precision in a fraction of a second using just the right numerical algorithms (how would you do these things?):

```
max(f)                              norm(diff(f),1)
ans = 0.422423192472707             ans = 40.021432099272715

sum(f)                              r = roots(f); r(10)
ans = -0.010800293307751            ans = 5.025153047777319
```

The chebfun team, funded by EPSRC, currently consists of myself, post-doc Rodrigo Platte, DPhil student Ricardo Pachón, regular visitor Toby Driscoll from Delaware, and two MSc students. Users are beginning to pop up around the world. Our vision is a computational tool that feels symbolic, like Maple or Mathematica, but runs at the speed of numerics. The system already handles piecewise smooth functions, infinite domains, and ODE initial-, boundary-, and eigenvalue problems. For details, or to download, google chebfun.