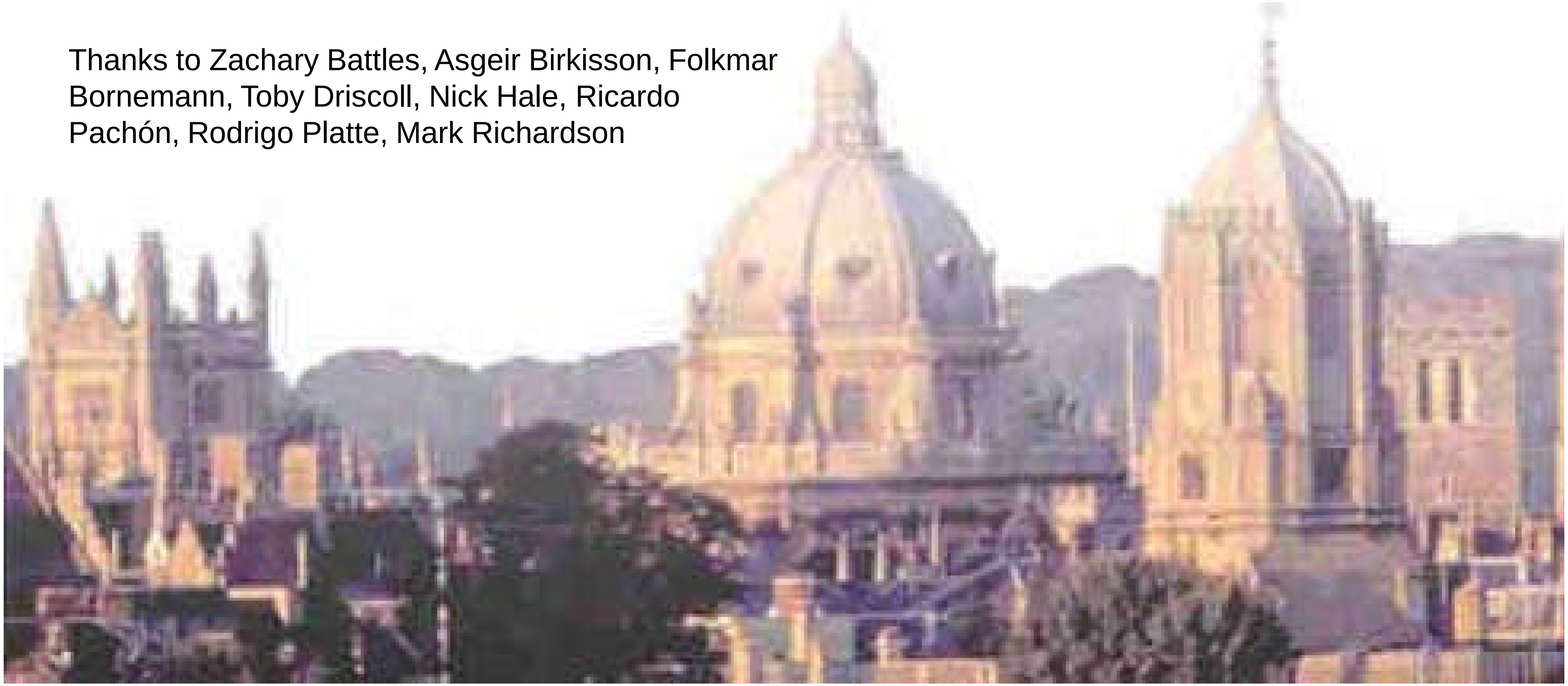
Chebfun: A new kind of numerical computing

Nick Trefethen, Oxford University

Thanks to Zachary Battles, Asgeir Birkisson, Folkmar Bornemann, Toby Driscoll, Nick Hale, Ricardo Pachón, Rodrigo Platte, Mark Richardson



SYMBOLIC COMPUTING (e.g. Maple, Mathematica)

Manipulate formulas exactly.
When you want numbers, evaluate the formulas.

PROBLEM: most problems cannot be solved symbolically. Even when they can, symbolic expressions tend to grow exponentially.

E.G., what's the integral of exp(-x) $\cos(6x)^5 \sin(5x)^6$ from -1 to 1?

Maple or Mathematica can figure out the answer symbolically:

$$\frac{6}{37} e^{-1} \sin(1) \cos(1)^5 - \frac{324}{629} e^{-1} \sin(1) \cos(1)^3 - \frac{45}{512} e \cos(2) + \frac{63}{2368} e \sin(6) + \frac{15}{25856} e \sin(10) + \frac{21}{8704} e \cos(4) + \frac{75}{640256} e \sin(50) + \frac{105}{25216}$$

$$e \sin(14) + \frac{15}{50432} e \cos(14) + \frac{21}{4736} e \cos(6) - \frac{3}{7424} e \sin(12) - \frac{15}{1664} e \sin(8) + \frac{75}{115328} e \sin(30) - \frac{45}{256} e \sin(2) + \frac{366}{629} e^{-1} \sin(1) \cos(1)$$

$$+ \frac{6}{37} e \sin(1) \cos(1)^5 - \frac{15}{410624} e \cos(20) - \frac{75}{102656} e \sin(20) + \frac{57}{73984} e \sin(38) + \frac{3}{147968} e \cos(38) + \frac{68661}{644096} e^{-1} - \frac{45}{166016} e \sin(36) + \frac{1}{37}$$

$$e \cos(1)^6 + \frac{21}{2176} e \sin(4) - \frac{1}{29696} e \cos(12) - \frac{105}{40192} e \sin(28) - \frac{81}{629} e \cos(1)^4 + \frac{9}{1664} e \sin(18) + \frac{1}{3328} e \cos(18) - \frac{75}{204928} e \sin(40)$$

$$- \frac{15}{1639424} e \cos(40) - \frac{3}{29504} e \sin(48) - \frac{75}{8224} e \sin(16) - \frac{5}{664064} e \cos(36) + \frac{3}{51712} e \cos(10) - \frac{15}{160768} e \cos(28) + \frac{5}{230656} e \cos(30)$$

$$- \frac{15}{13312} e \cos(8) + \frac{183}{629} e \cos(1)^2 - \frac{75}{131584} e \cos(16) + \frac{3}{1280512} e \cos(50) - \frac{1}{472064} e \cos(48) - \frac{1}{3687424} e \cos(60) - \frac{15}{921856} e \sin(60)$$

$$+ \frac{195}{86656} e \sin(26) + \frac{15}{173312} e \cos(26) - \frac{3}{51712} e^{-1} \cos(10) + \frac{15}{160768} e^{-1} \cos(28) - \frac{5}{230656} e^{-1} \cos(30) + \frac{15}{13312} e^{-1} \cos(8) - \frac{183}{629} e^{-1} \cos(1)^2$$

$$+ \frac{75}{131584} e^{-1} \cos(16) - \frac{3}{1280512} e^{-1} \cos(50) + \frac{1}{472064} e^{-1} \cos(48) + \frac{1}{3687424} e^{-1} \cos(60) - \frac{15}{921856} e^{-1} \sin(60) + \frac{195}{86656} e^{-1} \sin(26)$$

$$- \frac{15}{173312} e^{-1} \cos(26) - \frac{324}{629} e \sin(1) \cos(1)^3 + \frac{63}{2368} e^{-1} \sin(6) + \frac{15}{25856} e^{-1} \sin(10) - \frac{21}{8704} e^{-1} \cos(4) - \frac{68661}{644096} e - \frac{15}{36928} e \sin(24)$$

$$- \frac{5}{295424} e \cos(24) + \frac{366}{629} e \sin(1) \cos(1) + \frac{45}{512} e^{-1} \cos(2) + \frac{75}{640256} e^{-1} \sin(50) + \frac{105}{25216} e^{-1} \sin(14) - \frac{15}{50432} e^{-1} \cos(14) - \frac{21}{4736} e^{-1} \cos(6)$$

$$- \frac{3}{7424} e^{-1} \sin(12) - \frac{15}{1664} e^{-1} \sin(8) + \frac{75}{115328} e^{-1} \sin(30) - \frac{45}{256} e^{-1} \sin(2) - \frac{15}{36928} e^{-1} \sin(24) + \frac{5}{295424} e^{-1} \cos(24) + \frac{15}{410624} e^{-1} \cos(20)$$

$$- \frac{75}{102656} e^{-1} \sin(20) + \frac{57}{73984} e^{-1} \sin(38) - \frac{3}{147968} e^{-1} \cos(38) - \frac{45}{166016} e^{-1} \sin(36) - \frac{1}{37} e^{-1} \cos(1)^6 + \frac{21}{2176} e^{-1} \sin(4) + \frac{1}{29696} e^{-1} \cos(12)$$

$$- \frac{105}{40192} e^{-1} \sin(28) + \frac{81}{629} e^{-1} \cos(1)^4 + \frac{9}{1664} e^{-1} \sin(18) - \frac{1}{3328} e^{-1} \cos(18) - \frac{75}{204928} e^{-1} \sin(40) + \frac{15}{1639424} e^{-1} \cos(40)$$

$$- \frac{3}{29504} e^{-1} \sin(48) - \frac{75}{8224} e^{-1} \sin(16) + \frac{5}{664064} e^{-1} \cos(36)$$

NUMERICAL COMPUTING (e.g. Matlab, C, Fortran)

Work with numerical approximations instead of exact expressions.
Perform each operation to relative accuracy of about 10^{-16}.

By evaluating at each step in this way rather than just
at the end, we avoid the combinatorial explosion.

PROBLEM: what if we want not just numbers but functions like $f(x)$?

> OUR VISION
> to compute with functions numerically
>
> *"Computing with symbolic feel and numerical speed"*

HOW FLOATING-POINT ARITHMETIC COUNTERS THE COMBINATORIAL EXPLOSION

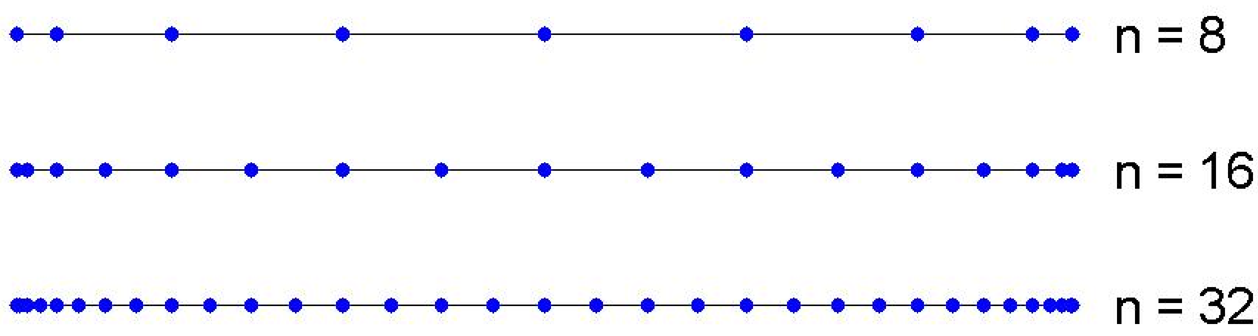
Symbolic computing
find the solution exactly, then round to 16 digits

Numerical computing
round to 16 digits at every step along the way

Our plan
to compute with functions in this round-at-every-step way

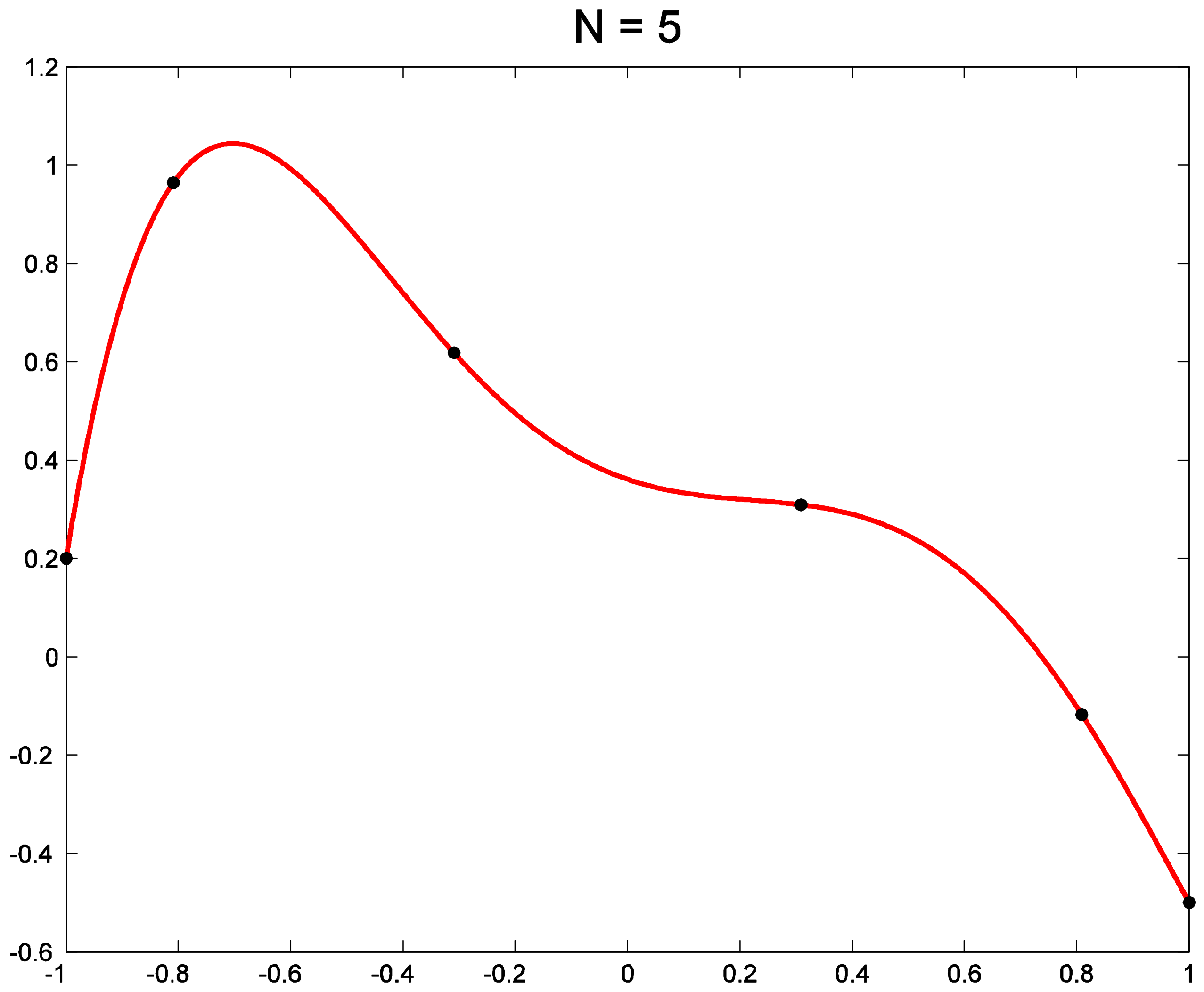
CHEBYSHEV POINTS IN [−1,1]

$x_j = \cos(j\pi/n)$, $0 \le j \le n$.   Clustered near the boundaries.
Outstanding properties for polynomial interpolation.

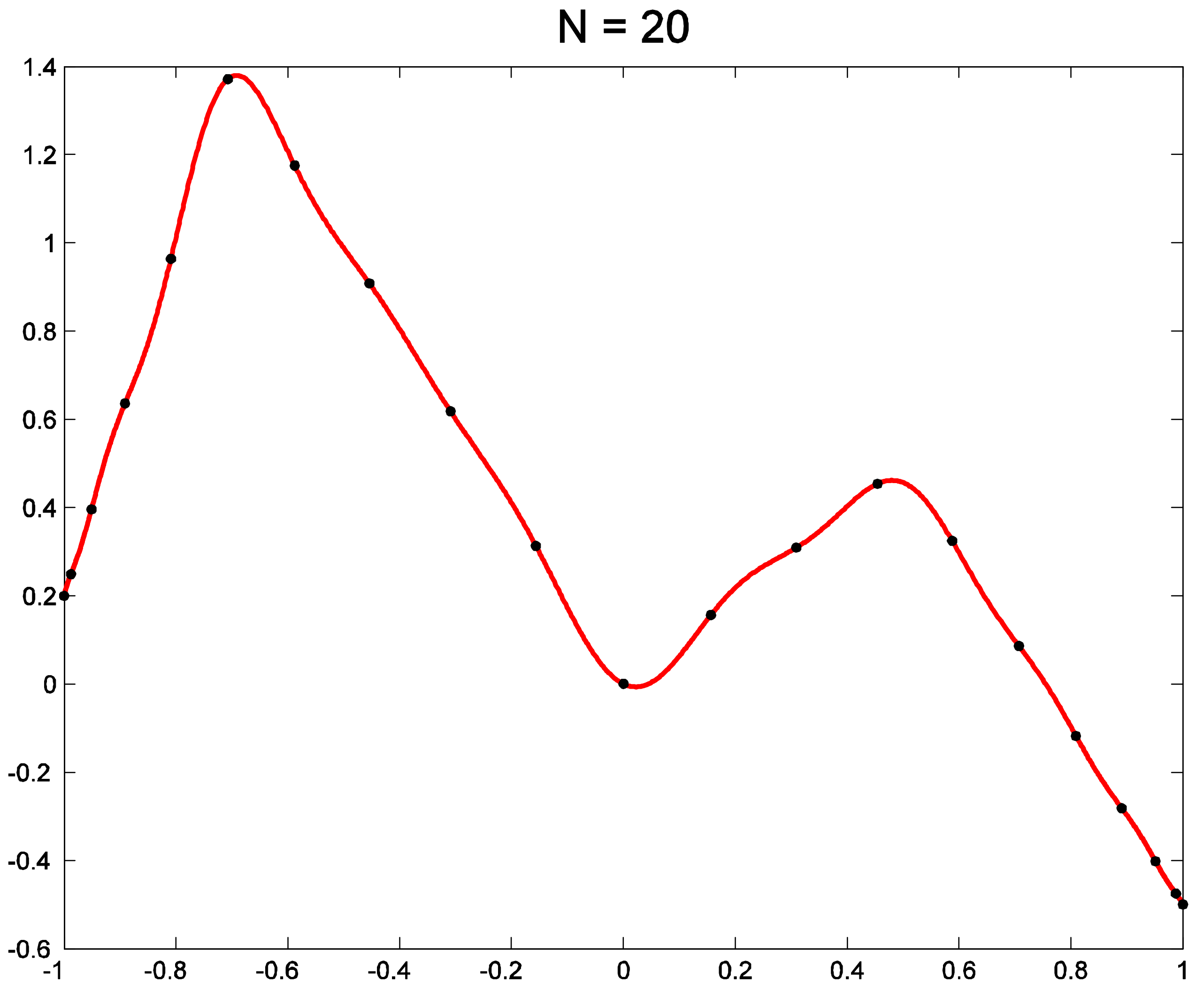


Chebyshev… Bernstein… Lanczos… Clenshaw… Fox… Elliott… Mason… Rivlin… Good… Salzer… Orszag… Geddes…

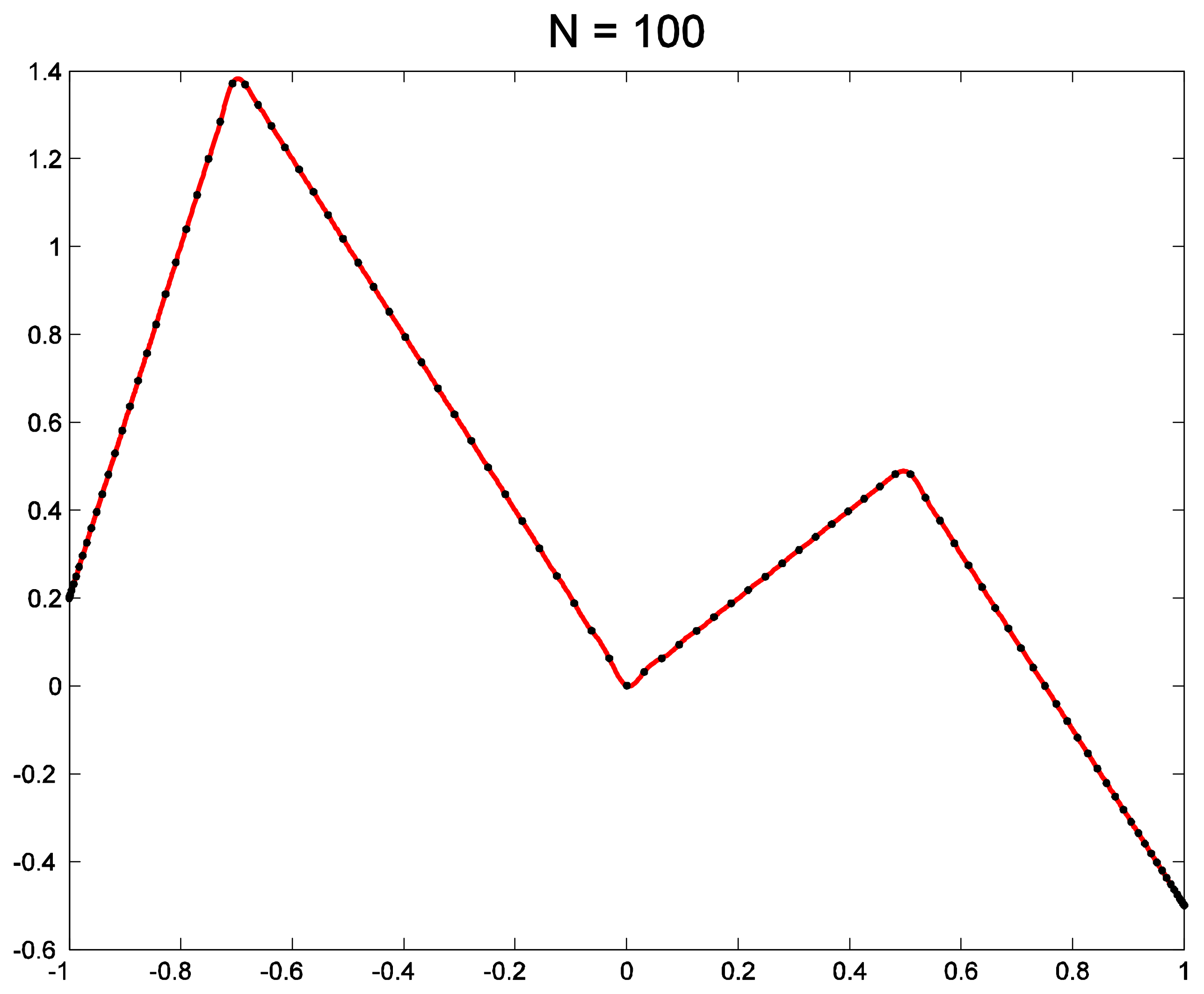
Example of polynomial interpolation in Chebyshev points



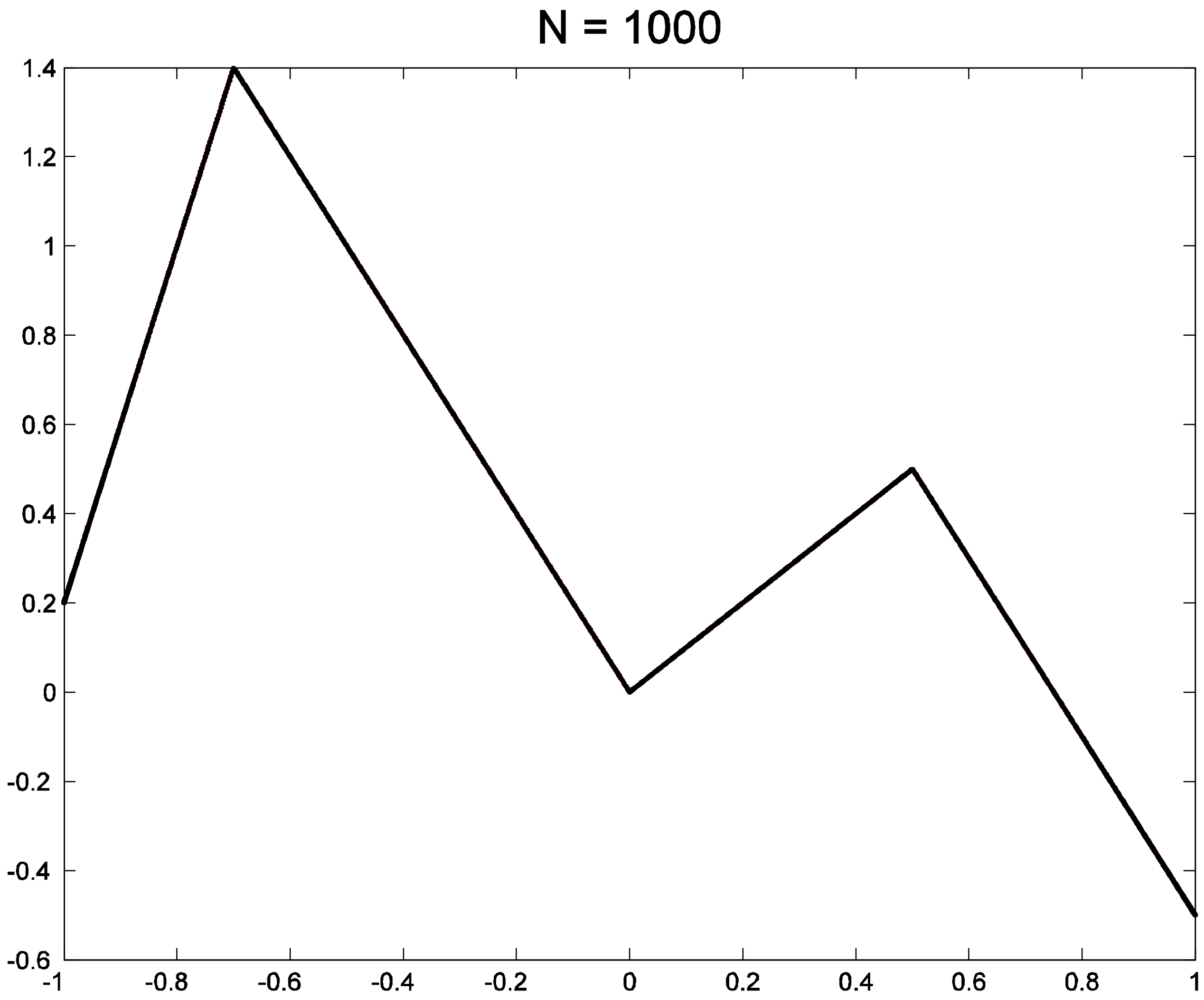
Example of polynomial interpolation in Chebyshev points



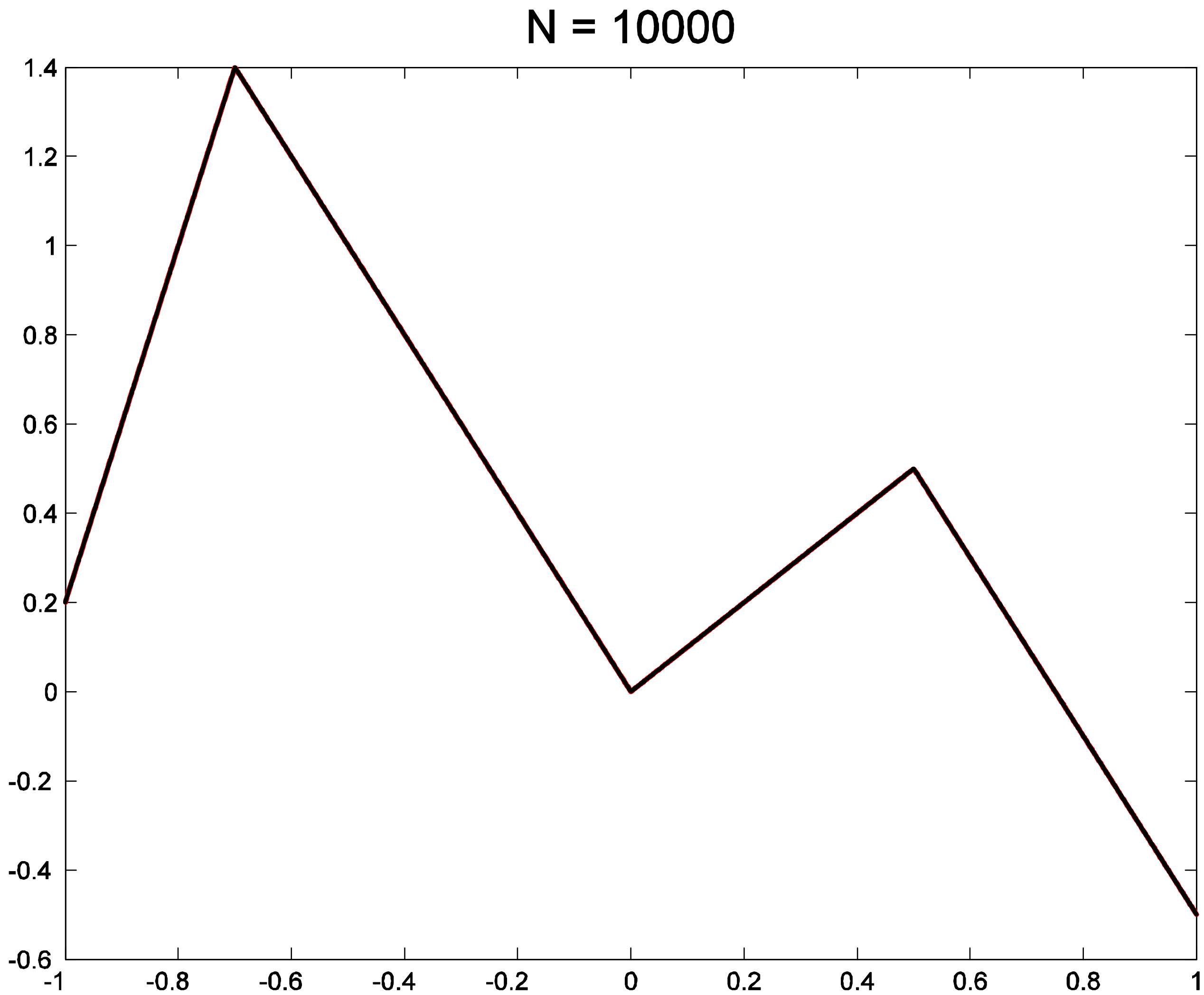
Example of polynomial interpolation in Chebyshev points



Example of polynomial interpolation in Chebyshev points



Example of polynomial interpolation in Chebyshev points



NOTATION FOR FIVE THEOREMS ABOUT POLYNOMIAL INTERPOLATION IN CHEBYSHEV POINTS

f = continuous function on [–1,1]

p^* = best max-norm degree n polynomial approximation of f

p = degree n interpolant of f in the Chebyshev pts.

$\|f - p\|$: error in Chebyshev interpolation

$\|f - p^*\|$: smallest possible error among all polynomials

"NEAR-BEST"

Theorem 1. $\|f - p\| \le [\,2 + (2/\pi)\log n\,]\,\|f - p^*\|$.

Ehlich & Zeller 1966

"SPECTRAL ACCURACY"

Theorem 2. If $f, f', \dots, f^{(k-1)}$ are absolutely continuous and $f^{(k)}$ has bounded variation, then $\|f - p\| = O(n^{-k})$.

Mastroianni & Szabados 1995

Theorem 3. If f is analytic in the closed ellipse with foci ± 1 and semiaxis lengths summing to ρ, then

$$\|f - p\| = O(\rho^{-n}) .$$

follows from Bernstein 1912

BARYCENTRIC INTERP.

Theorem 4. Barycentric interpolation formula:

$$p(x) = \frac{\sum'' (-1)^j f(x_j) / (x - x_j)}{\sum'' (-1)^j / (x - x_j)} .$$

M. Riesz 1916
Salzer 1972

Theorem 5. The barycentric formula is numerically stable.

N. J. Higham 2004

FINDING ROOTS OF A POLYNOMIAL IN AN INTERVAL

First, convert from values at Chebyshev pts to coefficients of expansion in basis of Chebyshev polynomials (FFT: work O(n log n)).

Now compute the zeros as eigenvalues of a colleague matrix
E.G. the roots of $a_0T_0 + a_1T_1 + a_2T_2 + a_3T_3 - \frac{1}{2}T_4$ are the eigs of

$$\begin{pmatrix} & 1 & & \\ \frac{1}{2} & & \frac{1}{2} & \\ & \frac{1}{2} & & \frac{1}{2} \\ & & \frac{1}{2} & & \frac{1}{2} \end{pmatrix} + \begin{pmatrix} & & & \\ & & & \\ & & & \\ a_0 & a_1 & a_2 & a_3 \end{pmatrix}$$

(Specht 1960, Good 1961)

If n is large, use recursive subdivision of intervals to bring dimensions down to O(100) (J. P. Boyd 2002). This improves the overall operation count to $O(n^2)$.

THE CHEBFUN PROJECT

Chebfuns are Matlab vectors overloaded for smooth or piecewise smooth functions defined on an interval [a,b]. Each piece is represented by a Chebyshev interpolant, with the number of points determined automatically to get about 15-digit precision.

Some of the overloaded functions:

abs	atan	coth	erf	horzcat	min	power	set	tan
acos	atanh	csc	erfc	imag	minus	prod	shift	tanh
acosh	ceil	csch	erfcx	isempty	mrdivide	prolong	sign	times
acot	chebfun	cumprod	erfinv	isreal	mtimes	rdivide	sin	uminus
acoth	chebpoly	cumsum	exp	ldivide	ne	real	sinh	uplus
acsc	comet	define	feval	length	norm	restrict	size	var
acsch	conj	diag	fix	log	plot	roots	sqrt	vertcat
asec	conv	diff	flipud	log10	plot3	round	std	svd
asech	cos	display	floor	log2	plus	sec	subsasgn	qr
asin	cosh	domain	get	max	poly	sech	subsref	cond
asinh	cot	eq	gmres	mean	polyval	semilogy	sum	rank

E.G. if f is a chebfun, then

sum(f) evaluates an integral, roots(f) finds zeros,
diff(f) computes the derivative, max(f) finds the maximum.

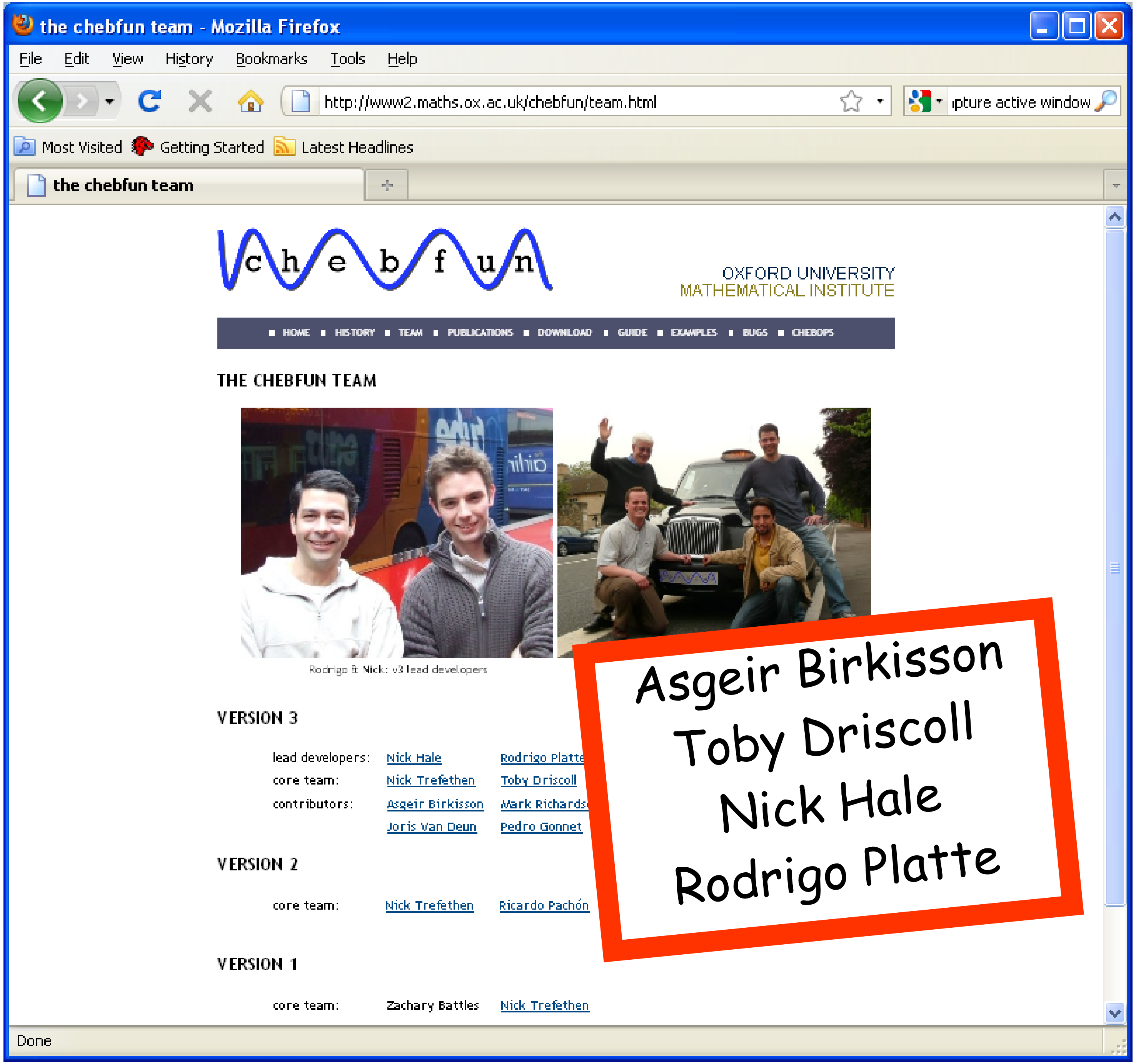
Version 1 was released in 2005

Version 2 was released in 2008

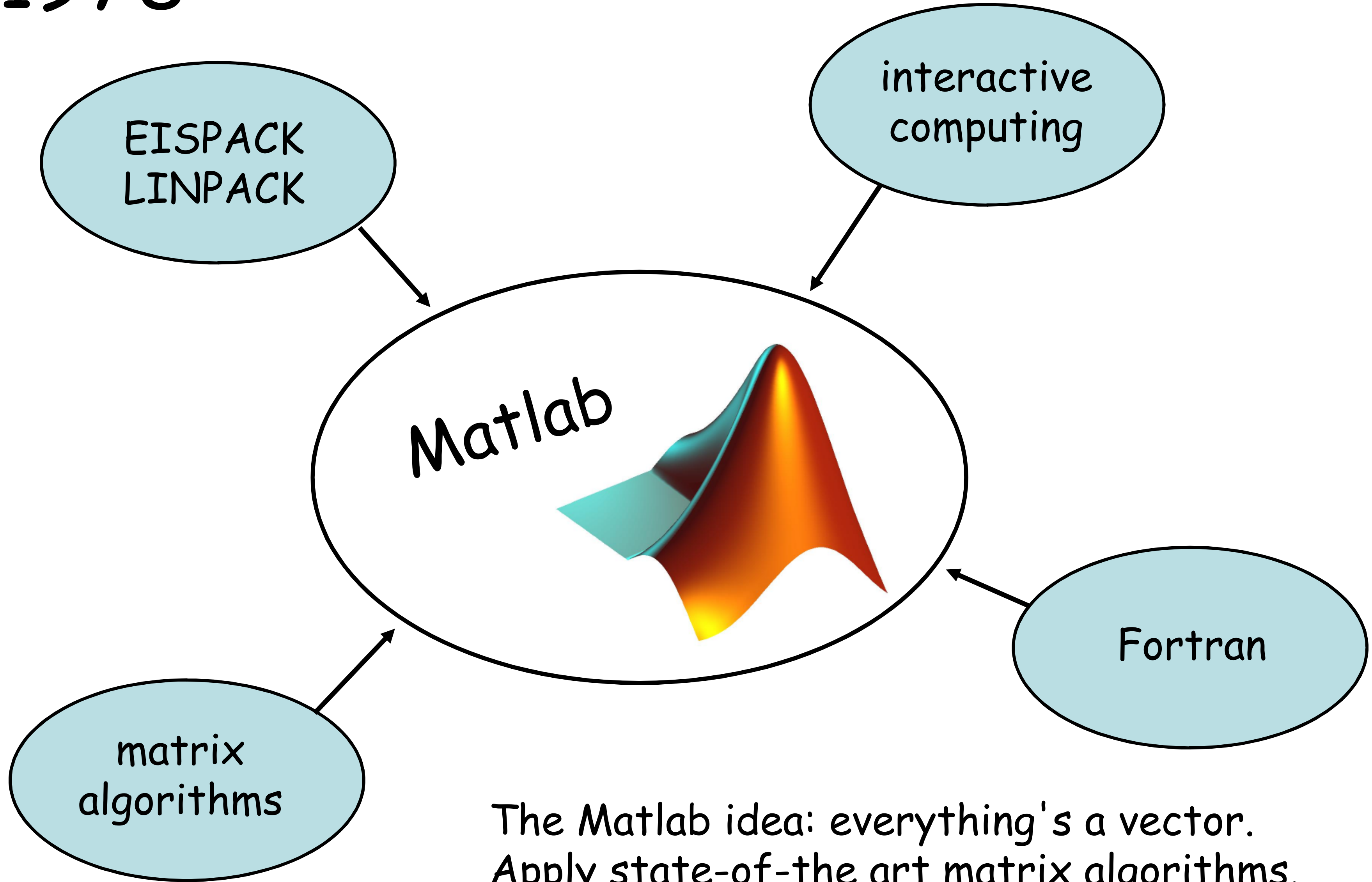
Version 3 was released in 2009

Latest update: last week! (Version 3.1111)

Freely available — just google "chebfun"



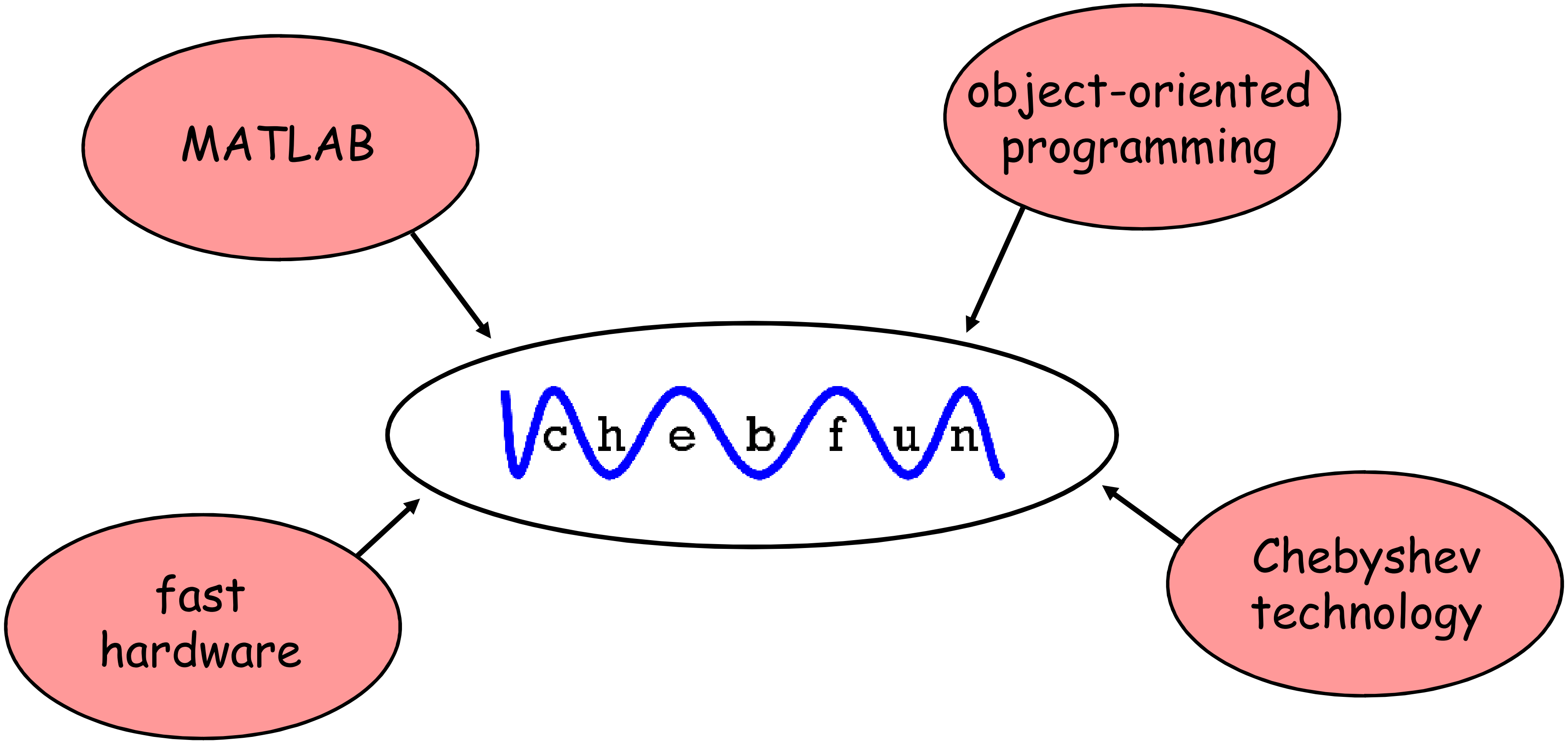
1978



The Matlab idea: everything's a vector.
Apply state-of-the art matrix algorithms.

(LU, QR, SVD, eigenvalues,...)

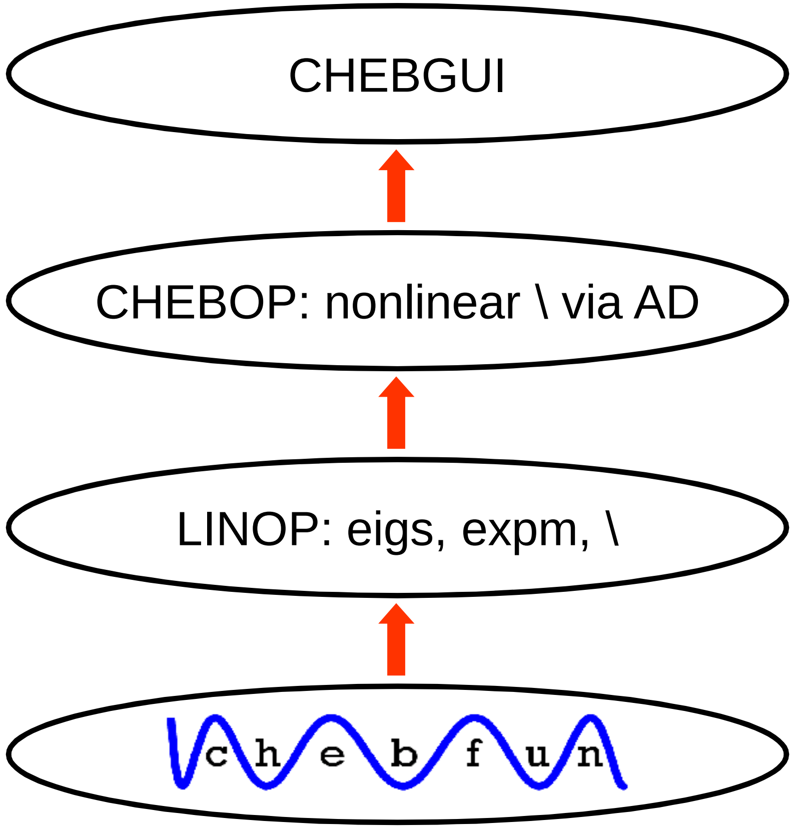
2005



The Chebfun idea: overload Matlab's vectors to functions.
Apply state-of-the art approximation algorithms.

("Chebyshev technology" for interpolation, rootfinding, quadrature, diff eqs,...)

Chebfun has expanded in a dozen directions.
But one line of development seems particularly interesting.




ASGEIR BIRKISSON

Demonstration