Root Finding with Chebyshev Polynomials: Background and Applications in 2 Dimensions

Lucas C. Bouck

George Mason University
Department of Mathematical Sciences

February 2, 2018

StReeTs Seminar
George Mason University
Fairfax, VA

Joint Work with Ian H. Bell (NIST Applied Chemicals and Materials Division)

Outline

Big Picture

NIST Thermophysical Properties of Fluids Group:

- Mission is to provide reference data for thermophysical properties of fluids
- Provide experimental data
- Developed software to fill in the gaps of experimental data
- End products are libraries like NIST REFPROP[1]and CoolProp[2]
- The computational group fills in the gap with empirical equations of state

[1]E. Lemmon et. al. REFPROP Version 10.0, 2017

[2]I. Bell et. al. *Pure and Pseudo-pure Fluid Thermophysical Property Evaluation and the Open-Source Thermophysical Property Library CoolProp*, 2014.

What is an Equation of State (EOS)?

- Empirical equation that is built to fit the experimental data while holding to physical constraints
- Contains information like melting line, gas liquid transition line, critical point, etc
- Fitting EOS involve hard optimization problems that is a research area of its own

Motivation for Robust Root Finding Methods

- Ian Bell was first interested in root finding methods for EOS to compute an inverse problem[1]
- Useful in applications like fluid and steam in a nuclear reactor[2]
- Older EOS can be solved by finding the roots of a cubic polynomial
- More modern EOS involve transcendental functions and are much harder for the inverse problem
- **Goal:** Compromise between the two.
- **Approach:** Approximate the EOS with a polynomial and find the roots
- This approach has been effective and has led to a software library `ChebTools`[3], which is a variant on `chebfun`[4]

---

[1] I. Bell et. al. *Exceptional reliable density solving alg...* 2018.

[2] M. Kunick et. al. *Fast Calculation of Therm...*, 2008.

[3] I. Bell *ChebTools.*, 2018.

[4] T Driscoll. *Chebfun Guide*, 2014.

Polynomial Interpolation

Given

- $n+1$ points $\{x_j\}_{j=1}^{n+1} \subset [a, b]$
- A function, $f : [a, b] \to \mathbb{R}$

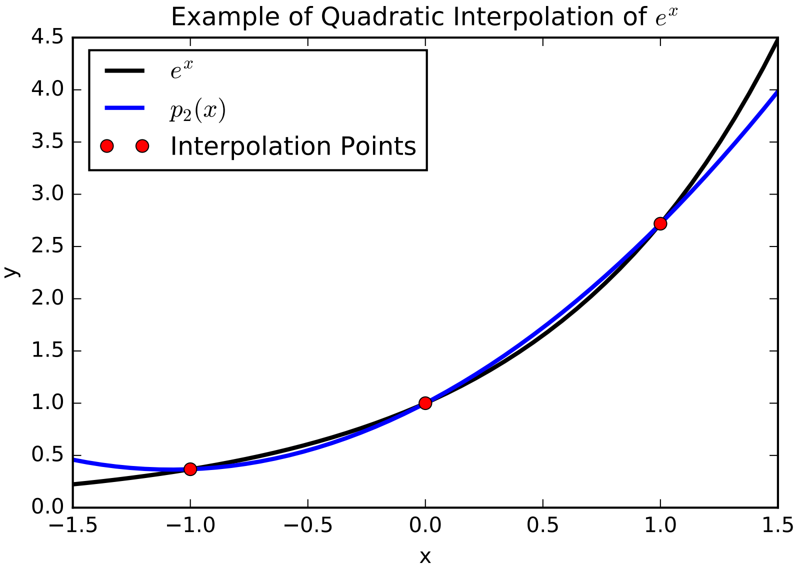
Goal

We want to find a polynomial, p_n, of degree $\leq n$ such that

$$p_n(x_j) = f(x_j), 1 \leq j \leq n+1$$

and provide a good approximation for f

Interpolation Example

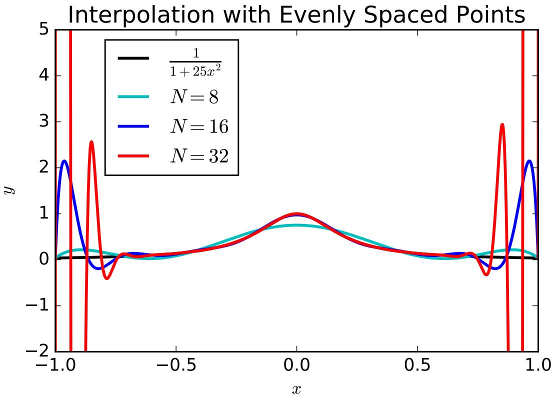


Where Evenly Spaced Interpolation Fails

The classic example is the Runge function

$$f(x) = \frac{1}{1+25x^2}.$$

On evenly spaced points, the polynomial interpolants diverge as $n \to \infty$



Polynomial Interpolation Error

We can actually express the error of polynomial interpolation for a function that is $C^{n+1}[a, b]$:[1]

$$f(x) - p_n(x) = \frac{f^{n+1}(c)}{(n+1)!} \prod_{j=1}^{n+1} (x - x_j),$$

where $c \in [a, b]$. **The Big Question:** What points minimize

$$\max_{x \in [a,b]} \left| \prod_{j=1}^{n+1} (x - x_j) \right| ?$$

For $[-1, 1]$, we'll see that the answer is **The Chebyshev Roots:**

$$x_j = \cos\left(\frac{2j-1}{2(n+1)}\pi\right), 1 \leq j \leq n+1$$

[1] P. Davis *Interpolation and Approximation* 1975. Thm. 3.1.1

Chebyshev Polynomials

Definition

$$T_n = \cos(n \arccos(x)), x \in [-1, 1]$$

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \end{aligned}$$

We can get a recurrence relation for $T_n(x)$ as well.

Let $\theta = \arccos(x)$ and $n \geq 1$.

$$\begin{aligned} T_{n+1}(x) + T_{n-1}(x) =& \cos((n+1)\theta) + \cos((n-1)\theta) \\ =& \cos(n\theta)\cos(\theta) - \sin(n\theta)\sin(\theta) \\ & + \cos(n\theta)\cos(-\theta) - \sin(n\theta)\sin(-\theta) \\ =& 2\cos(n\theta)\cos(\theta) = 2xT_n(x) \\ \implies T_{n+1}(x) =& 2xT_n(x) - T_{n-1}(x) \end{aligned}$$

Figure: First 5 Chebyshev Polynomials

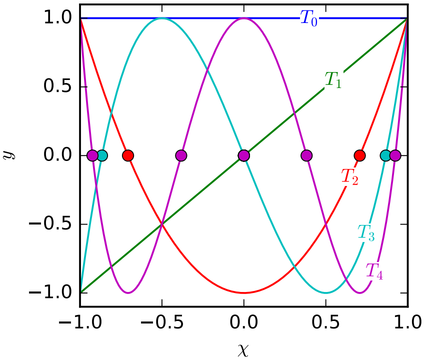
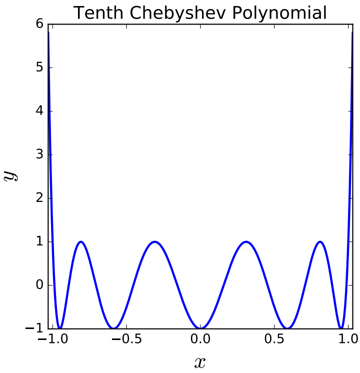


Figure: $T_{10}(x)$



Connection with Error Formula

Recall $T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$ for $n \geq 1$.

- The leading coefficient for $T_n(x)$ is 2^{n-1}.
- Therefore, the polynomial $2^{1-n}T_n(x)$ is monic and we can express

$$2^{1-n}T_n(x) = \left|\prod_{i=1}^{n}(x - x_i)\right|$$

- $\max_{x\in[-1,1]} |2^{1-n}T_n(x)| = 2^{1-n}$
- All monic polynomials have a larger maximum than 2^{1-n}.[1]

Therefore, the Chebyshev roots are the best for our error formula.

[1] P. Davis *Interpolation and Approximation* 1975. Thm. 3.3.4

Error Estimate for Chebyshev Roots Points

We can now have an error estimate for interpolating through the Chebyshev points for $n \geq 1$

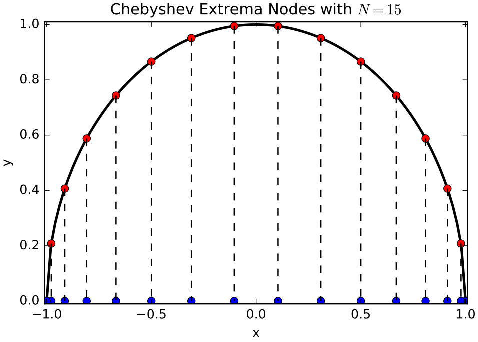
$$\|f(x) - p_n(x)\|_\infty \leq \max_{x\in[-1,1]} \left| \frac{f^{n+1}(c)}{(n+1)!} \prod_{i=1}^{n+1} n(c - x_i) \right|$$
$$\leq \frac{1}{2^n (n+1)!} \max_{c\in[-1,1]} \left| f^{n+1}(c) \right|$$

Other Useful Points

The Chebyshev Extrema Points

$$x_j = \cos\left(\frac{j\pi}{n}\right), 0 \le j \le n$$

are also useful for interpolation. They can be thought of as the x components of points evenly spaced on the unit circle:



The Chebyshev Extrema Points

The Chebyshev Extrema Points $x_j = \cos\left(\frac{j\pi}{n}\right)$ are used by us in `ChebTools` and also are the points used in `chebfun`. It isn't hard to

interpolate with these points because there is a matrix A where

$$\begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix} = A \begin{pmatrix} f(x_0) \\ \vdots \\ f(x_n) \end{pmatrix}$$ [1],

This gives us the advantage of having the coefficients a_j, which we'll need for root finding methods with polynomials.

[1] JP Boyd *Finding the Zeros of a Univariate Equation: Proxy Rootfinders, Chebyshev Interpolation, and the Companion Matrix*, 2013. Appendix A.

An Orthogonal View of Chebyshev Polynomials

Chebyshev polynomials are orthogonal with respect to a weight:

$$\langle T_n, T_m \rangle = \int_{-1}^{1} \frac{T_n(x) T_m(x)}{\sqrt{1-x^2}} \mathrm{d}x = \begin{cases} 0, n \neq m \\ \frac{\pi}{2}, n = m > 0 \\ \pi, n = m = 0 \end{cases}$$

We can approximate a function $f : [-1, 1] \to \mathbb{R}$ as

$$f(x) = \sum_{n=0}^{N} a_n T_n(x), \qquad a_n = \frac{\langle f, T_n \rangle}{\langle T_n, T_n \rangle},$$

which is we'll call a Chebyshev projection of f.

Uniform Convergence[1]

If f is Lipschitz continuous on $[-1, 1]$, then the Chebyshev projections converge uniformly to f on $[-1, 1]$

[1] L. N. Trefethen, *Approximation Theory and Approximation Practice*, 2013. Theorem 3.1

Convergence Properties

Differentiable Functions[1]

Let $f \in C^{\nu-1}[-1,1]$ and have f^{ν} be of bounded variation. Then the interpolating Chebyshev polynomial, p_n, satisfies:

$$\|f - p_n\|_\infty = \mathcal{O}(n^{-\nu})$$

Analytic Functions[2]

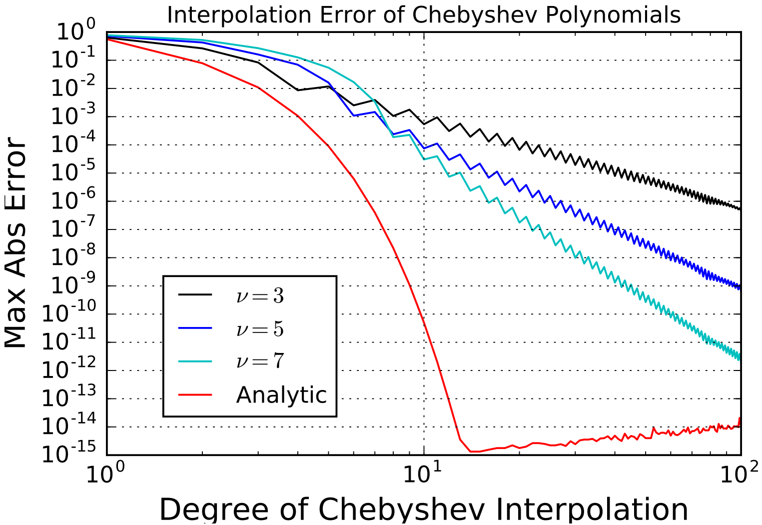
Let f be analytic on $[-1,1]$. Then the interpolating Chebyshev polynomial, p_n, satisfies:

$$\|f - p_n\| = \mathcal{O}(\rho^{-n}),$$

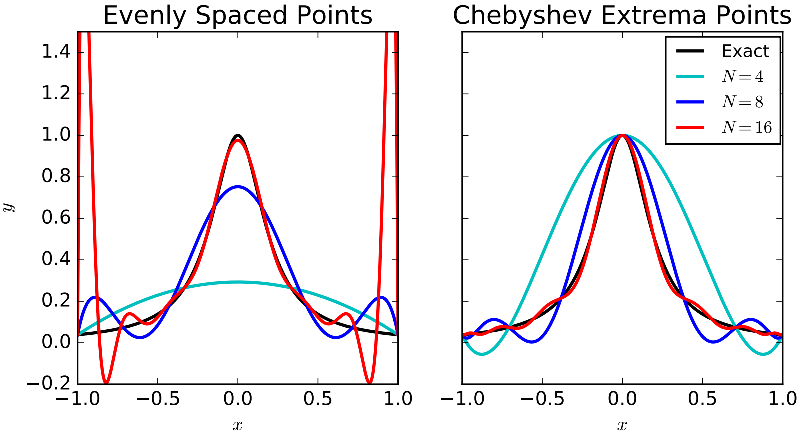
[1] L. N. Trefethen, *Approximation Theory and Approximation Practice*, 2013. Theorem 7.2

[1] Theorem 8.2

Convergence Examples



Chebyshev Interpolation Can Handle the Runge Function



Root Finding: The Companion Matrix Method[1]

Idea: Convert a root x to an eigenvalue of a matrix.
First we start with

$$xT_0(x) = T_1(x)$$

and

$$xT_n(x) = \frac{1}{2}T_{n+1}(x) + \frac{1}{2}T_{n-1}(x), n \geq 1$$

We can arrange the above equation as a matrix equation

$$\begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \ddots & \frac{1}{2} \\ 0 & 0 & \cdots & \frac{1}{2} & 0 \end{pmatrix} \begin{pmatrix} T_0(x) \\ T_1(x) \\ \vdots \\ T_{n-1}(x) \end{pmatrix} = x \begin{pmatrix} T_0(x) \\ T_1(x) \\ \vdots \\ T_{n-1}(x) \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ \vdots \\ \frac{1}{2}T_n(x) \end{pmatrix}$$

We'll denote C to be the LHS matrix

[1] JP Boyd *Finding the Zeros of a Univariate Equation...*, 2013. Appendix C

Now add and subtract away the interpolant, $p_n(x)$

$$C \begin{pmatrix} T_0(x) \\ T_1(x) \\ \vdots \\ T_{n-1}(x) \end{pmatrix} = x \begin{pmatrix} T_0(x) \\ T_1(x) \\ \vdots \\ T_{n-1}(x) \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ \vdots \\ \frac{1}{2} T_n(x) \end{pmatrix}$$
$$+ \begin{pmatrix} 0 \\ 0 \\ \vdots \\ \gamma \end{pmatrix} \left(a_n T_n(x) + \sum_{j=0}^{n-1} a_j T_j(x) - p_n(x) \right)$$

Setting $\gamma = \frac{1}{2a_n}$ we get:

$$C \begin{pmatrix} T_0(x) \\ T_1(x) \\ \vdots \\ T_{n-1}(x) \end{pmatrix} = x \begin{pmatrix} T_0(x) \\ T_1(x) \\ \vdots \\ T_{n-1}(x) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ \frac{1}{2a_n} \end{pmatrix} \left(\sum_{j=0}^{n-1} a_j T_j(x) - p_n(x) \right)$$

The Companion Matrix

Rearranging and setting x to be a root of $p_n(x)$, we end with

$$\begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \ddots & \frac{1}{2} \\ \frac{-a_0}{2a_n} & \frac{-a_1}{2a_n} & \cdots & \frac{1}{2} - \frac{a_{n-2}}{2a_n} & -\frac{a_{n-1}}{2a_n} \end{pmatrix} \mathbf{v} = x\mathbf{v},$$

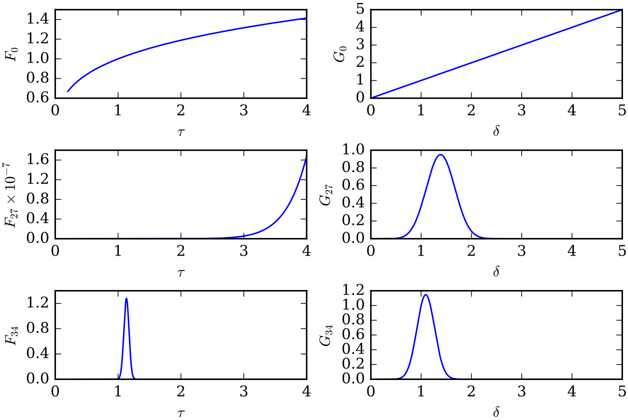
which is an eigenvalue problem whose eigenvalues are the roots of $p_n(x)$. We can now eigenvalue solve to get all roots of $p_n(x)$.

(Boyd, 2013)[1] Shows how to derive companion matrices for a general basis.

[1] JP Boyd *Finding the Zeros of a Univariate Equation...*, 2013. Appendix C

Example EOS

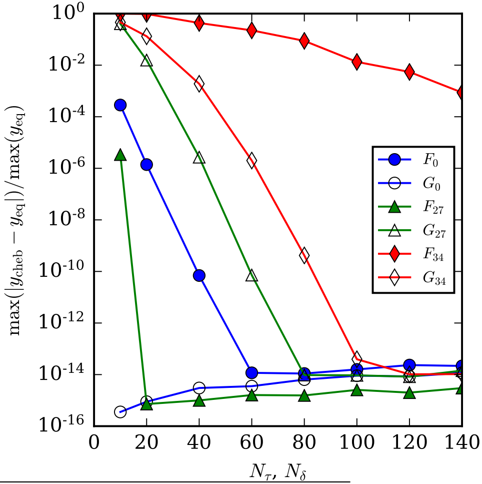
Figure: Example EOS terms for Nitrogen[1]



[1] I. Bell et. al. *Exceptional reliable density solving algorithms for multi parameter mixture models from Chebyshev expansion root finding* 2018.

Effectiveness of Chebyshev Interpolants for Our Application

Figure: Accuracy of Chebyshev Approximations[1]



Speed:

- Have been able to use root finding techniques to get comparable speeds to other options[1]

[1] I. Bell et. al. *Exceptional reliable density solving algorithms for multi parameter mixture models from Chebyshev expansion root finding* 2018.

Why go to Two Dimensions?

- So far we have assumed that we know 1 proprerty
- For Ian Bell's density solver, we assumed we knew the temperature
- The more general problem is finding the temperature and pressure at which two desired thermophysical properties hold

The Problem

Global 2-D Root Finding Problem

We want to find all solutions $\mathbf{x} \in \Omega \subset \mathbb{R}^2$ to

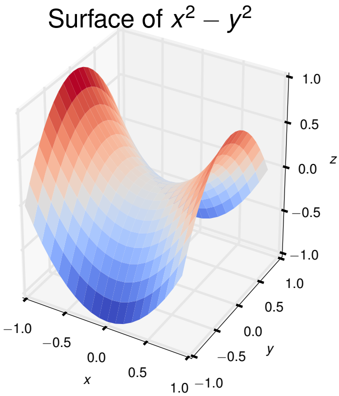
$$\mathbf{F}(\mathbf{x}) = \mathbf{0}$$

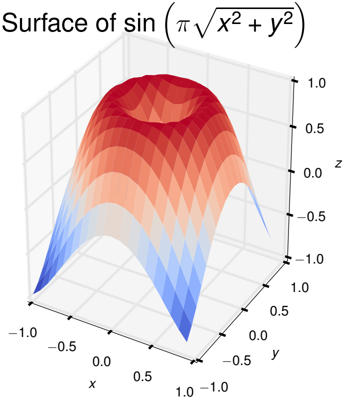
Additional assumptions

- Ω will be a bounded domain and rectangular for now
- Treat **F** as $\begin{pmatrix} f(x,y) \\ g(x,y) \end{pmatrix}$, where $x, y \in \mathbb{R}$ and f, g are scalar functions
- Assume f, g are smooth enough (more on this later)
- Assume finitely many roots

Example

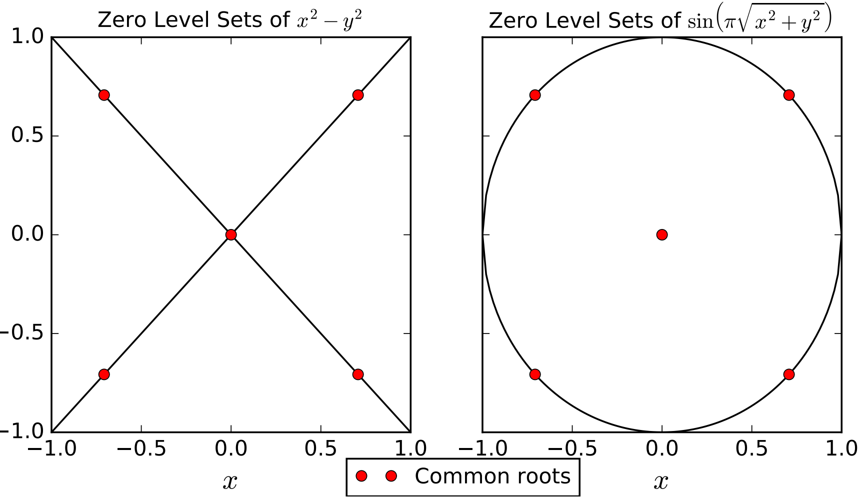
$f(x,y) = x^2 - y^2$ and $g(x,y) = \sin\left(\pi\sqrt{x^2+y^2}\right)$ with the square domain $\Omega = [-1,1]^2$





Example (cont.)

Another view: We are finding where f and g's zero level sets intersect



A Need for a Method to Find All Roots

- Iterative methods may take multiple runs to find all roots.
- We need a method to find all solutions at once, just like the companion matrix methods in 1-D

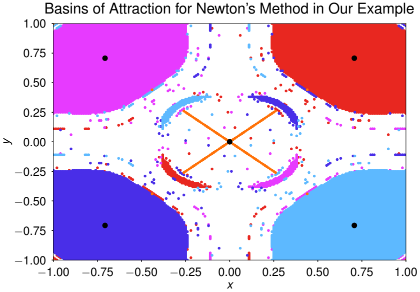


Figure: Initial guesses in white areas did not converge to roots in Ω

Townsend's 2-D Interpolation Idea[1]

Want to express

$$f(x,y) \approx \sum_{i=0}^{n} \sum_{j=0}^{m} T_i(x) T_j(y)$$

in as few 1-D interpolations as possible. One way to do it is express

$$f(x,y) \approx \sum_{i=0}^{K} p_i(x) q_i(y)$$

where p_i, q_i are 1-D Chebyshev expansions K is as small as we can make it to get desired accuracy.

When is K is relatively small compared to the degree of polynomials, Townsend calls it a **Low Rank Approximation** of f

[1] A. Townsend. *Computing with Functions in Two Dimensions*, 2014.

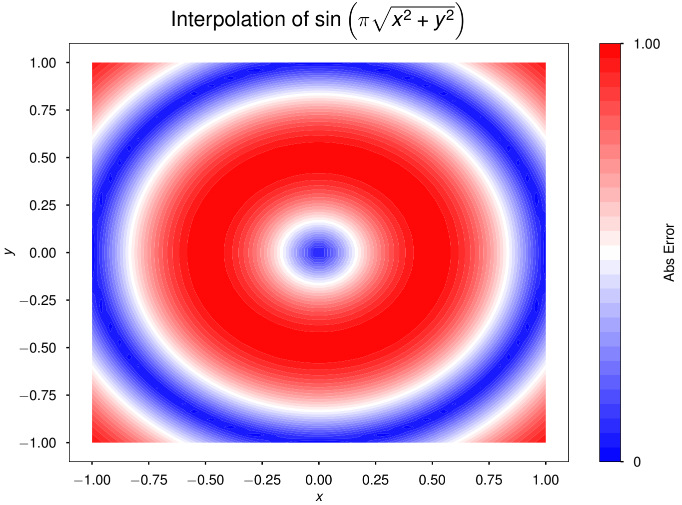
Townsend's 2-D Interpolation Algorithm

- Define our first error function as $e_0(x, y) = f(x, y)$
- Define later error functions as $e_k(x, y) \coloneqq e_{k-1}(x, y) - P_{k-1}(x, y)$ where P_{k-1} is our approximation

Algorithm Outline

- Find (x_k, y_k) s.t. $|e_k(x_k, y_k)| = \max |e_k(x, y)|$
- Do 1-D interpolations of $e_k(x, y_k)$ and $e_k(x_k, y)/e_k(x_k, y_k)$ denoted $p_x(x)$ and $p_y(y)$
- Compute new approximation $P_k(x, y) = P_{k-1}(x, y) + p_x(x)p_y(y)$

Interpolation From Our Example



The Advantage of The Low Rank Approximation

The pivots below are for a 64×64 grid of Chebyshev nodes. Note we do less interpolations than if we used all 64^2 points

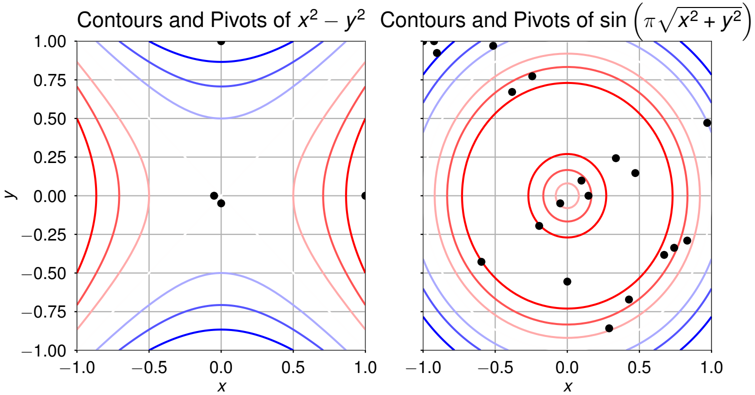
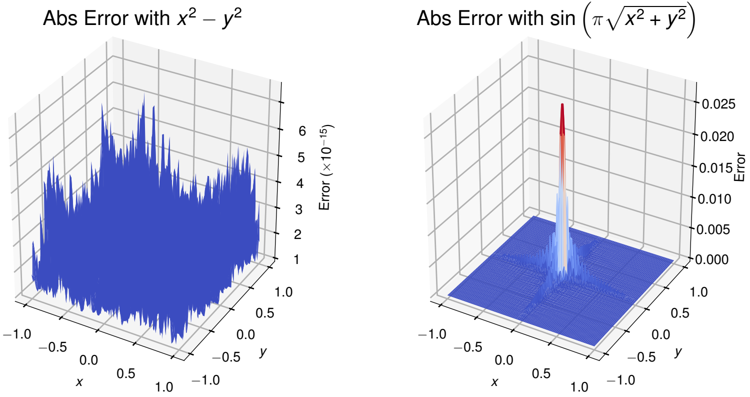


Figure: Plot of the contours of our example functions with pivots from the algorithm

Approximations of Our Example Functions

- Almost machine precision accuracy with $x^2 - y^2$
- Struggles with $\sin\left(\pi\sqrt{x^2+y^2}\right)$ due to the lack of differentiability at $(0,0)$



Bézout Resultant Method

- From our Chebyshev approximations $p_f(x, y), p_g(x, y)$, we can construct a Bézout matrix polynomial $B(x)$
- $\det(B(x_0)) = 0 \iff p_f(x_0, \cdot)$ and $p_g(x_0, \cdot)$ have common root

If p_f, p_g are of degree $(m_f, n_f), (m_g, n_g)$ respectively, then the resulting form of the matrix polynomial of degree M is

$$B(x) = \sum_{i=0}^{M} B_i T_i(x)$$

- B_i are square matrices of size $n = \max(n_f, n_g)$ and $M \leq m_f + m_g$.
- Solving $\det(B(x_0)) = 0$ involves linearizing $B(x)$[1]

[1] Y. Nakatsukasa *Computing the Common Zeros of Two Bivariate Functions Via Beout Resultants* 2014.

Solving the Matrix Polynomial Eigenvalue Problem[1]

Expressing $B(x) = \sum_{i=0}^{M} B_i T_i(x)$, then we can solve $\det\left(B(x)\right) = 0$ by solving the generalized eigenvalue problem

$$\frac{1}{2}\begin{pmatrix} -B_{M-1} & I_n - B_{M-2} & -B_{M-3} & \cdots & -B_0 \\ I_n & 0 & I_n & & \\ & \ddots & \ddots & \ddots & \\ & & I_n & 0 & I_n \\ & & & 2I_n & 0 \end{pmatrix}\mathbf{v}$$

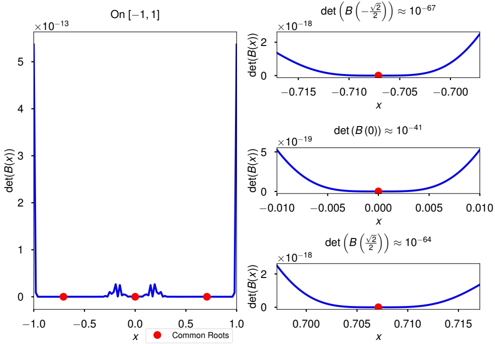
$$= \lambda \begin{pmatrix} B_M & & & \\ & I_n & & \\ & & \ddots & \\ & & & I_n \end{pmatrix}\mathbf{v}$$

Has computational complexity of $\mathcal{O}\left(M^3 n^3\right)$

[1] Y. Nakatsukasa *Computing the Common Zeros of Two Bivariate Functions Via Bezout Resultants* 2014.

Conditioning of the Bézout Matrix Polynomials

- The Bézout matrix polynomial technique can square the condition number of the common root.[1]
- The `chebfun` algorithm refines the roots by recomputing the matrix polynomial problem on a zoomed in region of the root

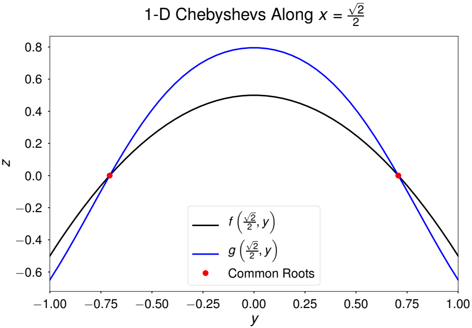


[1] Y. Nakatsukasa *Computing the Common Zeros of Two Bivariate Functions Via Bezout Resultants* 2014.

1-D Root Finding

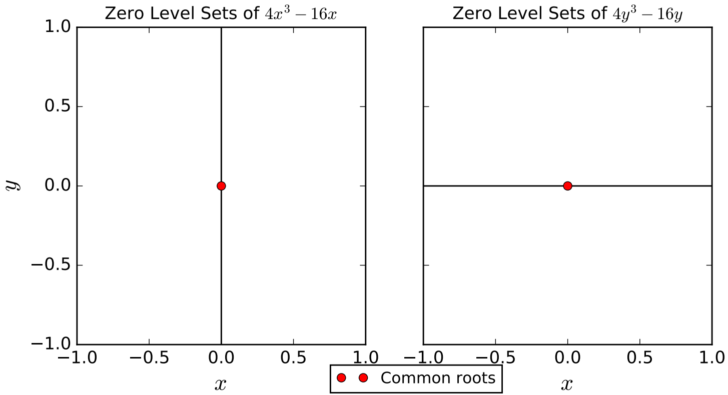
1-D Root Finding

- We now have the possible x values for where there are common roots
- Employ a companion matrix method finding the roots of 1-D Chebyshev polynomials



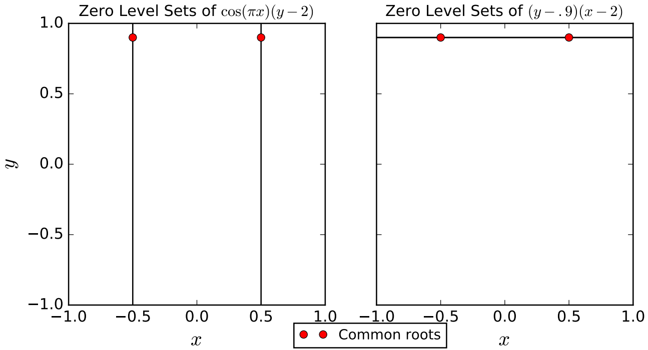
Example 1

	$\|\cdot\|_2$ Error	Time (s)
`ChebTools`	6.97×10^{-32}	0.004268
`chebfun`	7.7×10^{-16}	0.522



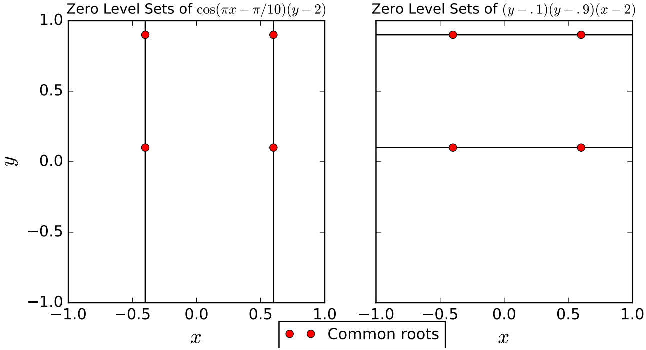
Example 2

	$\|\cdot\|_2$ Error	Time (s)
`ChebTools`	6.21×10^{-16}	214.059
`chebfun`	2.92×10^{-10}	0.294



Example 3

	$\|\cdot\|_2$ Error	Time (s)
`ChebTools`	3.24×10^{-16}	195.903
`chebfun`	2.77×10^{-11}	0.296



Application Motivates Current Work

Subdivision for Non-rectangular Domains

- Right now, we don't have a subdivision technique to reduce the size of the matrix polynomials
- We want a subdivision technique that will be able to handle non-rectangular domains
- Empirical equations of state in thermodynamics have ranges of validity which are not rectangular domains
- Solutions outside the domain will not make physical sense
- Some properties are not defined at certain points (ex: Critical Point)
- The work done by Townsend can not handle this problem in general

Example Domains for Water

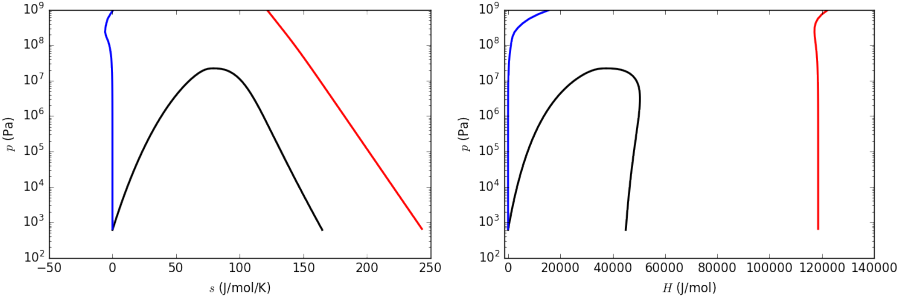


Figure: Examples of liquid/vapor equilibrium region for water.

Domain with Critical Point

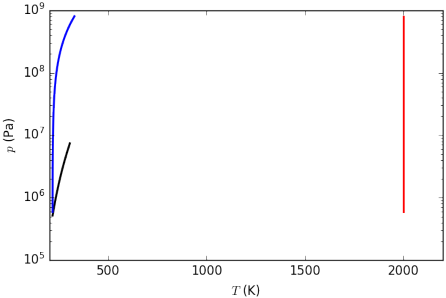


Figure: Example Domain with CO_2 that has a critical point

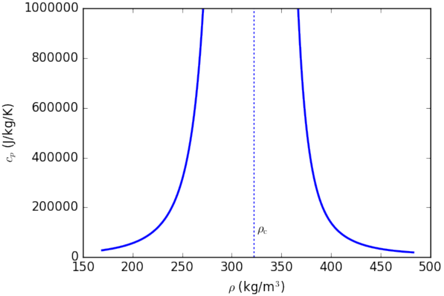


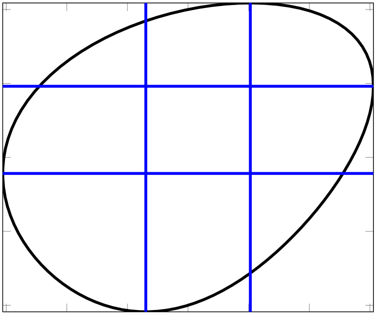
Figure: The specific heat of water becomes undefined at the critical point

Our Subdivision Idea

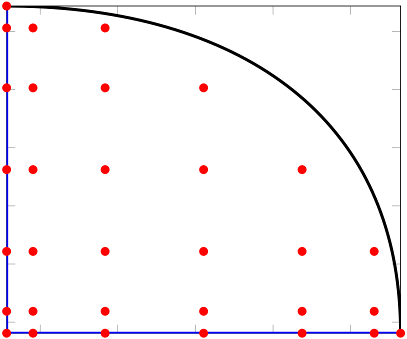
- Subdivide where the boundary is flat, vertical, or has a discontinuity in the derivative
- Our boundary curve can then be represented as a 1 variable function
- Will allow for quick checking of whether points are inside the domain
- Easy to do coordinate transformations to a rectangle

Subdivision Example

Interior of the closed curve $(1.5\cos t + .15\sin 2t, \sin t + .3\cos t)$ with $t \in [0, 2\pi]$



(a) Boundary in black and initial subdivisions in blue.



(b) Interior 7x7 Chebyshev nodes in red.

Figure: Example domain on the left and the upper right subdomain on the right

Future Work

For Non-Rectangular Domains

- Further develop ideas for subdivision methods, such as stopping criterion
- Look into root finding techniques for Chebyshev polynomials other than Bézout Resultant Method
    - Root finding in higher dimensions is needed for mixture EOS
    - Resultant methods will not generalize well for high dimensions[1]
    - Undergrad group at BYU working on applying algebraic geometry ideas to the root finding process

Software:

- Introduce GPU and parallel computing to the `ChebTools` library
- Add other features to the library
- We are looking for more developers
- Link: https://github.com/usnistgov/ChebTools

[1] V. Noferini *Numerical Instability of Resultant Methods for Multidimensional Rootfinding* 2016.

Acknowledgements

- Ian Bell and Bradley Alpert (NIST Applied Math Division)
- The Summer Undergraduate Research Fellowship program at NIST

Questions?