

# Rootfinding with Chebyshev Polynomials in 2 Dimensions

Lucas C. Bouck

George Mason University  
Department of Mathematical Sciences

August 2, 2017

SURF Colloquium  
National Institute of Standards and Technology  
Boulder, CO

Collaborator: Ian H. Bell (NIST Division 647)

- 1 The Problem
- 2 What are Chebyshev Polynomials and why use them?
- 3 The Algorithm

1 The Problem

2 What are Chebyshev Polynomials and why use them?

3 The Algorithm

# The Problem

## Global 2-D Rootfinding Problem

We want to find all solutions to

$$\begin{pmatrix} f(x, y) \\ g(x, y) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

in a given bounded domain  $\Omega$

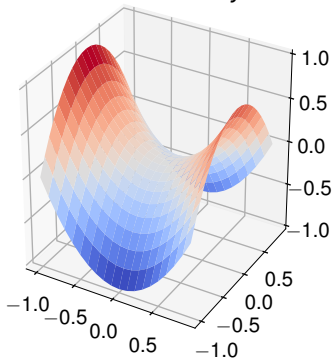
## The Big Picture: ChebTools

- ChebTools is a C++ library for working with Chebyshev expansions
- Inspired by the Matlab library Chebfun
- Solving this problem will be a significant feature for ChebTools
- With a higher level Python interface, ChebTools could be useful for a wide range of users

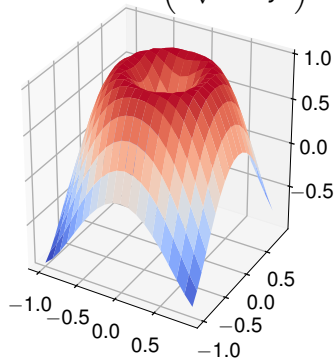
# Example

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

Surface of  $x^2 - y^2$

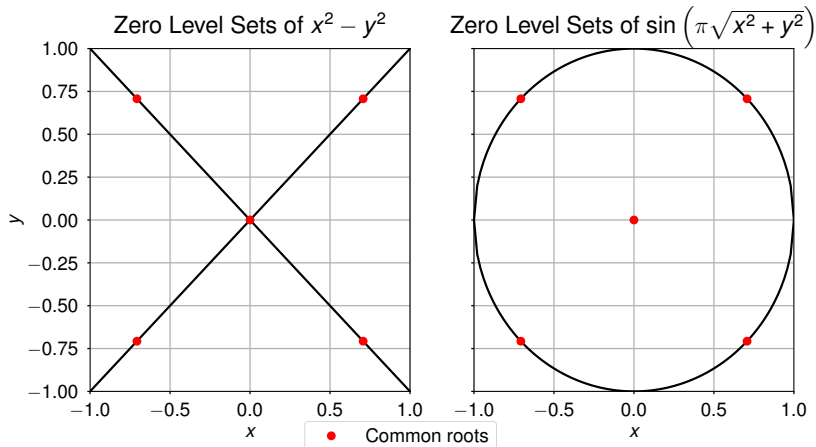


Surface of  $\sin\left(\pi\sqrt{x^2 + y^2}\right)$



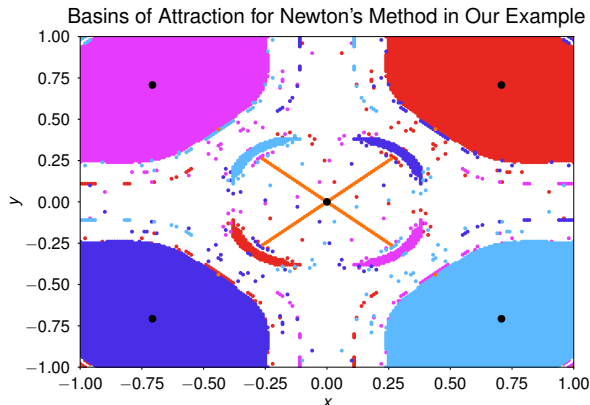
## Example (cont.)

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



# Local Methods May Not Be Good for Global rootfinding

There are actually infinitely many roots outside  $\Omega$  in our example. Local methods may converge to roots outside the domain of interest.



**Figure:** Note that initial guesses in white areas converged to roots outside our domain and the basin of attraction for  $(0,0)$  is relatively small

# A Need for a Global Method

Our example illustrates a few lessons

- Many iterative and local methods like Newton's method can only guarantee convergence to a root if the initial guess is sufficiently close
- Sufficiently close depends on the problem
- This may be extremely close for some roots like  $(0,0)$  in our example

Goal:

To develop a truly global rootfinding method



1 The Problem

2 What are Chebyshev Polynomials and why use them?

3 The Algorithm

# What are Chebyshev polynomials?

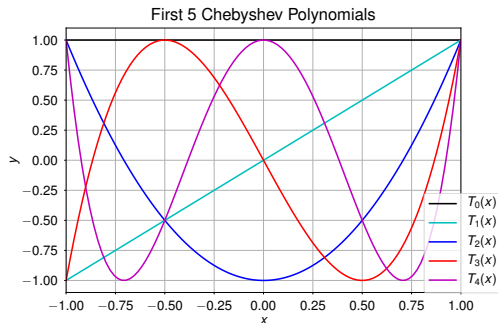
## Chebyshev polynomials

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

Orthogonal under a weighted inner product:

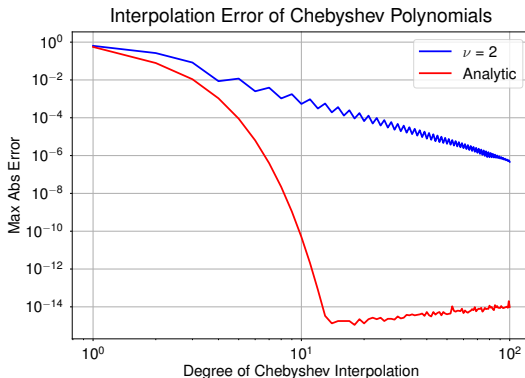
$$\int_{-1}^1 T_n(x) T_k(x) w(x) dx = 0$$

when  $n \neq k$   
and  $w(x) = \frac{1}{\sqrt{1-x^2}}$



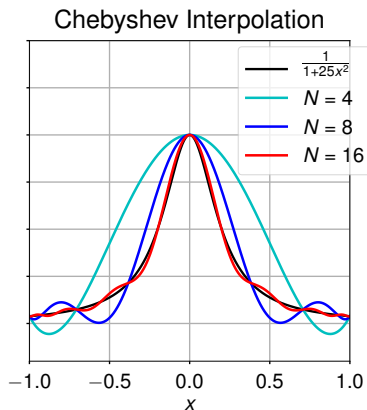
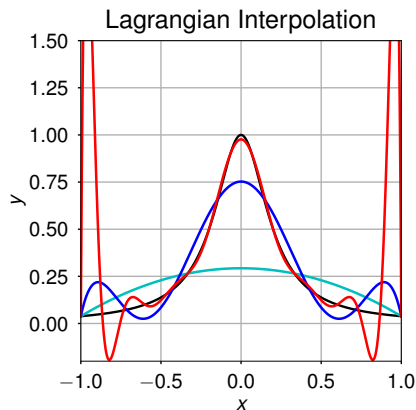
# Chebyshev polynomials provide accurate approximations

- Lipschitz continuity  $\implies$  uniform convergence of Chebyshev interpolations
- $\nu - 1$  continuous derivatives and a  $\nu$ th derivative of bounded variation  $\implies$  Chebyshev interpolation converges like  $\mathcal{O}(n^{-\nu})$
- Analytic function  $\implies$  geometric convergence



# Chebyshev polynomials are numerically stable

- Typical interpolation is susceptible to the **Runge phenomenon**
- Chebyshev interpolation minimizes this effect.



1 The Problem

2 What are Chebyshev Polynomials and why use them?

3 The Algorithm

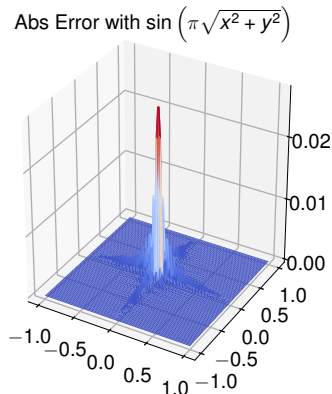
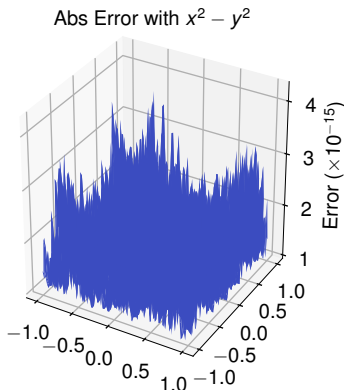
## 2-D Interpolation

### Idea:

- Pick point of maximum error
- Do 1-D interpolations in  $x$  and  $y$  directions

# Our Algorithm Can Provide Accurate Approximations

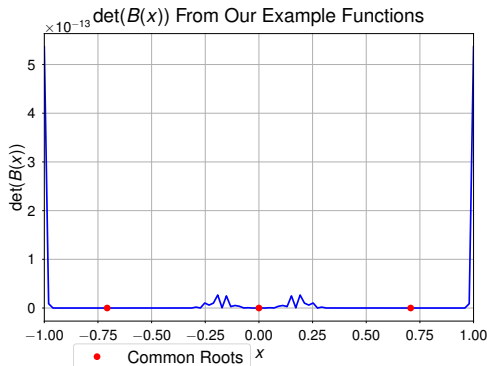
- Almost machine precision accuracy with  $x^2 - y^2$
- Struggles with  $\sin\left(\pi\sqrt{x^2 + y^2}\right)$  due to the lack of differentiability at  $(0,0)$



# Bézout Matrix Polynomials

We create a matrix polynomial  $B(x)$  from our Chebyshevs

- $B(x)$  is a matrix with polynomial entries
- $\det(B(x)) = 0 \iff$  common roots along the specific  $x$  value
- Solving  $\det(B(x)) = 0$  is a **matrix polynomial eigenvalue problem**





# 1-D Rootfinding and Local Refinement

- We now have the possible  $x$  values for where there are common roots
- Employ a well known 1-D rootfinding method again using Chebyshev Polynomials
- **Issue:** Poor conditioning of matrix polynomial problem  $\implies$  need for local refinement
- Polish the roots with Newton's method

# Results:

# Conclusions and Future Work

## Key Contributions:

- Made significant progress in replicating the 2-D rootfinding algorithm in chebfun2 with some modifications in C++
- Developed ideas and have a strategy for extending the algorithm to non-rectangular domains

## Future Work:

- Further develop ideas on non-rectangular domains and subdivision strategies
- Introduce GPU/parallel computing to the rootfinding process

## Other Contributions while at NIST:

- Idea for using Gram-Schmidt process to create orthogonal terms for developing equations of state

# Acknowledgements

- Dr. Ian Bell for the mentorship and guidance
- Dr. Bradley Alpert for the lively discussions and brainstorming
- The SURF program

References here