Orthogonal Polynomials

Cheng-Hao Fu, Youngwoo Kwon, Sai Zheng, Anthony Della Pella, Shoucheng Yu, Ahmad Barhoumi

Laboratory of Geometry at Michigan

LOG(M)

Introduction

Goal

We are looking for a way to effectively visualize zeros for Orthogonal Polynomials in the complex plane

Definition (Orthogonal Polynomial). *Orthogonal Polynomial are polynomials Q_n that are perpendicular to other polynomials P as defined by a specific inner product*

$$\langle P, Q_n \rangle = \int_{-1}^{1} P(x)Q_n(x)w(x)dx \quad (k = 0, 1, ..., n-1)$$

where w(x) is a given function. To normalize, we will assume that $Q(x)$ is monic.

Definition (Zeros of Orthogonal Polynomials). *A zero of an orthogonal polynomial $Q(x)$ is simply a value of x for which $Q(x) = 0$.*

Background and Methods

Visualizing the zeros

Checking Accuracy

We want to make sure our result maintain accuracy for higher degree n and less well-behaved zeros, visualizing the zeros can help us verify our algorithm's correctness for zeros that we more or less know the behavior of.

For this part, we will make heavy use of the following theorem:

Given $w(x) \geq 0$ for $x \in [a, b]$, and polynomials Q_n satisfying

$$\int_a^b x^k Q_n(x)w(x)dx = 0 \quad (k = 0, 1, ..., n-1)$$

It follows that Q_n has n simple zeros lying in $[a, b]$.

This result allows us to verify the approximate magnitude of n and $w(x)$ for which our code is reliable.

Calculating our Q(x)

Using linearity of integration, we can write the conditions for our $Q(x)$ from the definition as follows:

$$\int_{-1}^{1} x^k Q_n(x)w(x)dx = 0$$

for all $k = 0, \cdots, n-1$.
We can then define:

$$\mu_k = \int_{-1}^{1} x^k w(x)dx$$

We also write $Q(x)$ as the following:

$$Q_n = x^n + a_{n-1}x^{n-1} + ... + a_0$$

This will allow us to rewrite the definition of orthogonal polynomials into the following system of equations:

$$\mu_{2n-1} + a_{n-1}\mu_{2n-2} + \cdots a_0\mu = 0$$
$$\vdots$$
$$\mu_n + a_{n-1}\mu_{n-1} + \cdots a_0\mu_0 = 0$$

By calculating all of the μ_k values beforehand using standard integration, we can rely on a computer to solve the resulting system of equations and find the correct $Q_n(x)$.

Hankel Matrix

In our code, we make use of a structure called a Hankel matrix. Essentially, it's a matrix that matches the form of the linear system we are solving, where the diagonals from the top right to the bottom left are all populated with the same elements. This type of matrix has some special properties, but we use it in our research to make generating matrices easier, as it can be defined by the first column and last row.

$$\begin{bmatrix} \mu_{2n-2} & \mu_{2n-3} & \cdots & \mu_{n-1} \\ \mu_{2n-3} & \mu_{2n-4} & \cdots & \mu_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ \mu_{n-1} & \mu_{n-2} & \cdots & \mu_0 \end{bmatrix}$$

Results

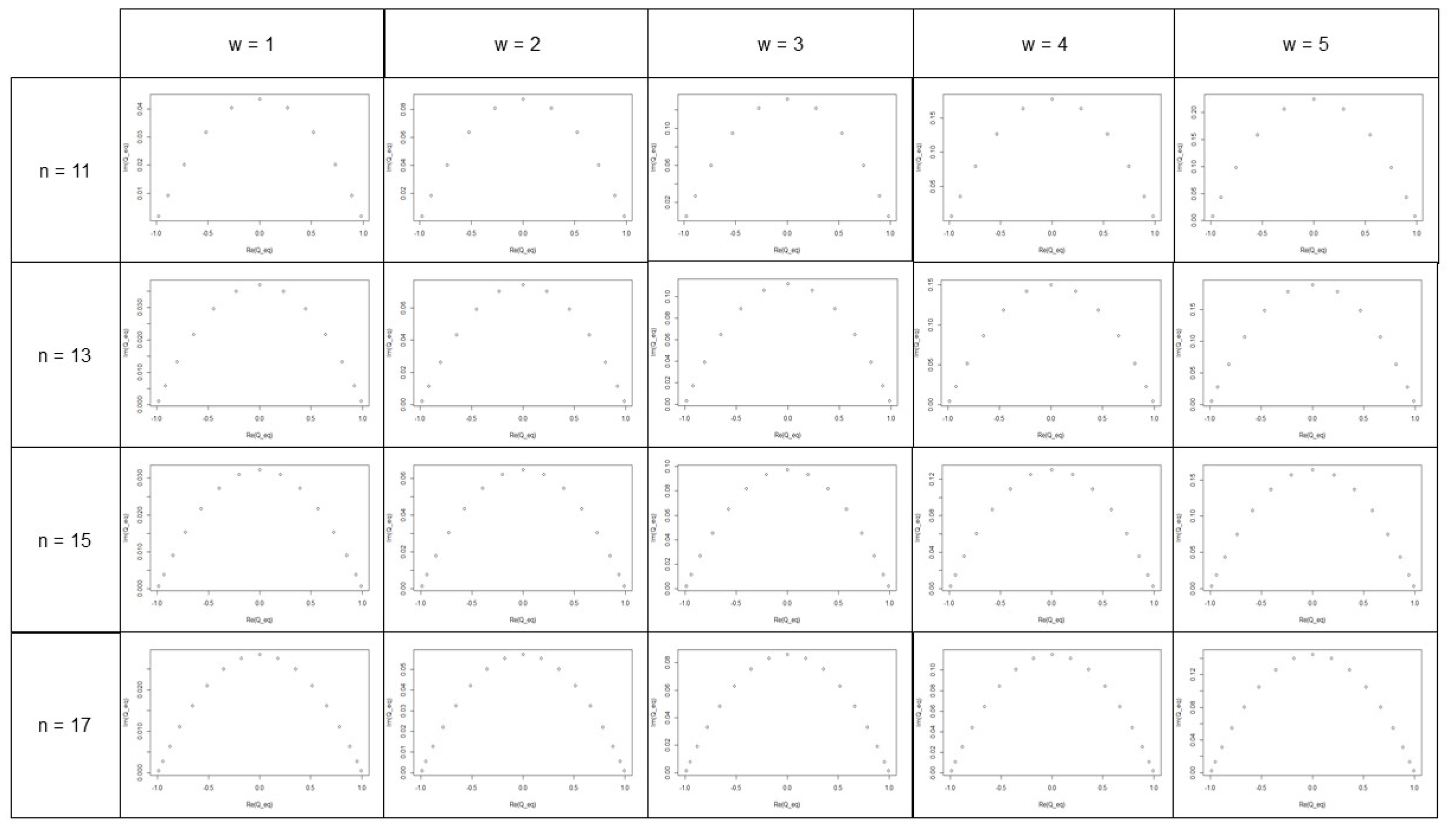
We used R programming language to find the zeros of the Kissing Polynomial, the orthogonal polynomial with $w(x) = e^{i\omega x}$.

General Relationship

We could see that zeros drew a parabolic graph with real values between -1 to 1.

The imaginary part of the zeros increases as n decreases, and ω increases.

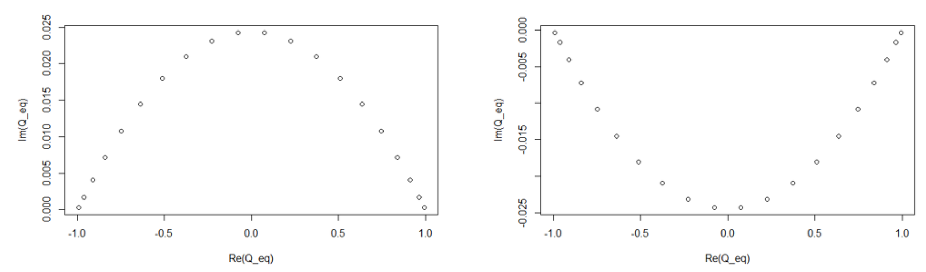
Figure 1: *n = 11, 13, 15, 17 w = 1, 2, 3, 4, 5*



Negative ω

If ω is negative, the graph is upside down.

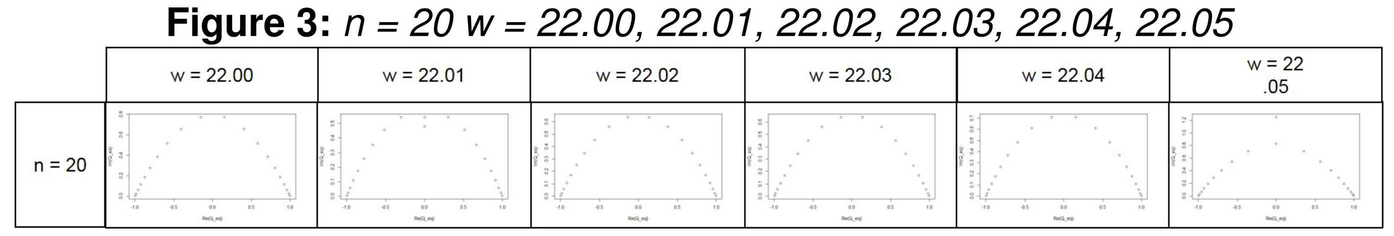
Figure 2: *n = 20, w = 1, -1*



Threshold ω

When ω keeps increase, there is a threshold where the graph can maintain the parabola shape. After that ω, the part of the zeros showed unusual behavior.

Figure 3: *n = 20 w = 22.00, 22.01, 22.02, 22.03, 22.04, 22.05*



Applications

Continued Fraction

Suppose we want to estimate $f(z) = \int_{-1}^{1} \frac{w(x)}{(z-x)}dx$, we can write $f(z)$ as a continued fraction. In fact $f(z)$ is equivalent to $\frac{P_n(z)}{Q_n(z)}$, where Q_n satisfies $\int_{-1}^{1} P(x)Q_n(x)w(x)dx = 0$

Gaussian Quadrature Rule

If we want to estimate $\int_a^b f(x)w(x) = a_0 f(x_0) + a_1 f(x_1) + ... + a_n f(x_n)$, the best x_is to estimate are the zeros of the orthogonal polynomial $Q_{n+1}(z)$ satisfy $\int_a^b Q_{n+1}(x)x^k w(x)dx = 0$

Future Directions

We aren't able to generate accurate results from both Hankel Matrix Method using Integration by Parts and Recurrence relationship in Matlab.
One possible reason is that we lose precision when we try to calculate the moments. First, we need to introduce the term - Condition Number.

Definition (Condition Number). *A condition number for a matrix measures how sensitive the answer is to perturbations in the input data during the solution process.*

Gautschi introduced an algorithm called modified Chebyshev algorithm. It allows us to transform ordinary moments into modified moments. Suppose we modified data with Bernstein polynomials, we can decrease our condition number from exponential in n to about 4^n. This means we will still have a relatively large condition number, hence lose d digits when our moments are perturbed by one unit in the last decimal.
In order to reduce the condition number significantly, one way is to modify our moments using Lagrange polynomials, which would give us a condition number $< 2n$, which is much smaller than something in exponential forms. This would require future work.

References

[1] Estelle Basor, Yang Chen and Torsten Ehrhardt. *PainlevÃ© V and time dependent Jacobi polynomials* 2009; arXiv:0905.2620. DOI: 10.1088/1751-8113/43/1/015204.

[2] Gabor Szego. *Orthogonal Polynomials.* American Mathematical Society Colloquium Publications, Vol. 23. Revised ed. American Mathematical Society, Providence, R.I., 1959.

[3] Gautschi Walter. *Orthogonal Polynomials : Computation and Approximation.* OUP Oxford, Oxford Science Publications. 2004.

VIEW THE PDF