Test Report on Interpolation & Polynomial Approximation

Name:张弛
ID: 117020910105

1 Introduction

The *Weierstrass* approximation theorem states that every continual function *f(x)* defined on an interval *[a, b]* can be uniformly approximated as closely as desired by a polynomial function $P_n(\mathrm{x})$ of sufficiently large degree ≤ n, i.e.

$$\lim_{n\to\infty}(\max_{a\le x\le b}\left|f(\mathrm{x})-\mathrm{P}_n(\mathrm{x})\right|)=0 \tag{1}$$

It is natural to construct approximating polynomials by interpolating at evenly-spaced nodes. However, Runge phenomenon demonstrates that interpolation can easily lead to divergent approximations. This test gives examples of how well different interpolation (or approximation) methods approximate Runge function by plotting their graphs respectively in Matlab.

$$f(\mathrm{x})=\frac{1}{1+25x^2} \qquad x\in[-1,1] \tag{2}$$

2 Solution

2.1 Lagrange Polynomial Interpolation

The approximating polynomial is derived from an interval of [-1, 1] which is evenly divided into *n* parts (n = 10, 20). Their graphs are as follows.

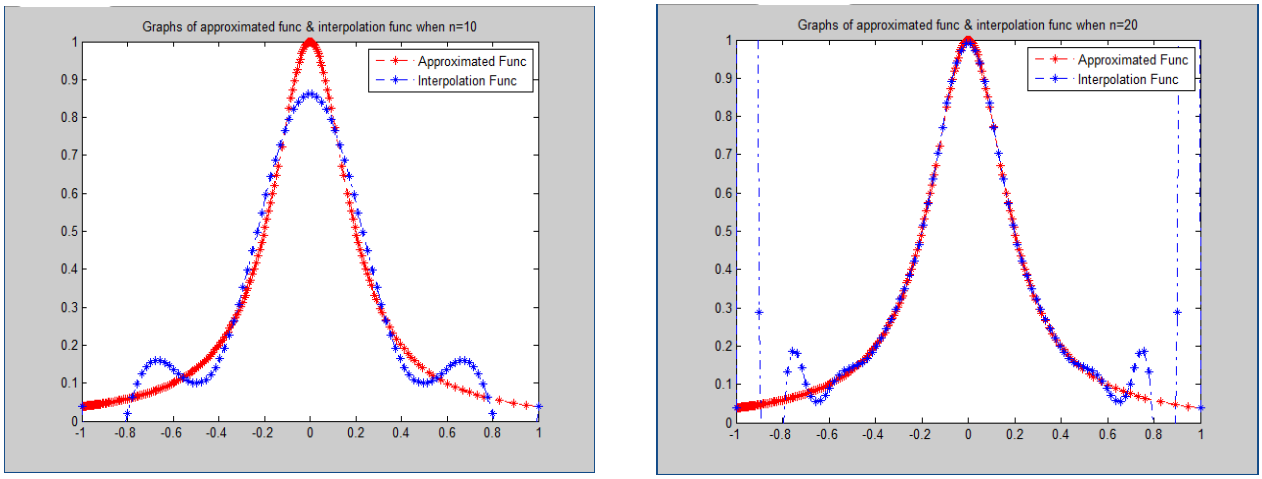


Figure1: Graphs of approximated func & interpolation func when n=10 and 20

The graph shows that our interpolation polynomial does not fit well to Runge function when n = 10. If we increase the number of nodes (n =20), however, the resulting polynomial fits beautifully to the central part of Runge function while oscillates toward the end of the interval. The oscillation becomes fiercer when n = 30, but the approximation to the center turns even better.

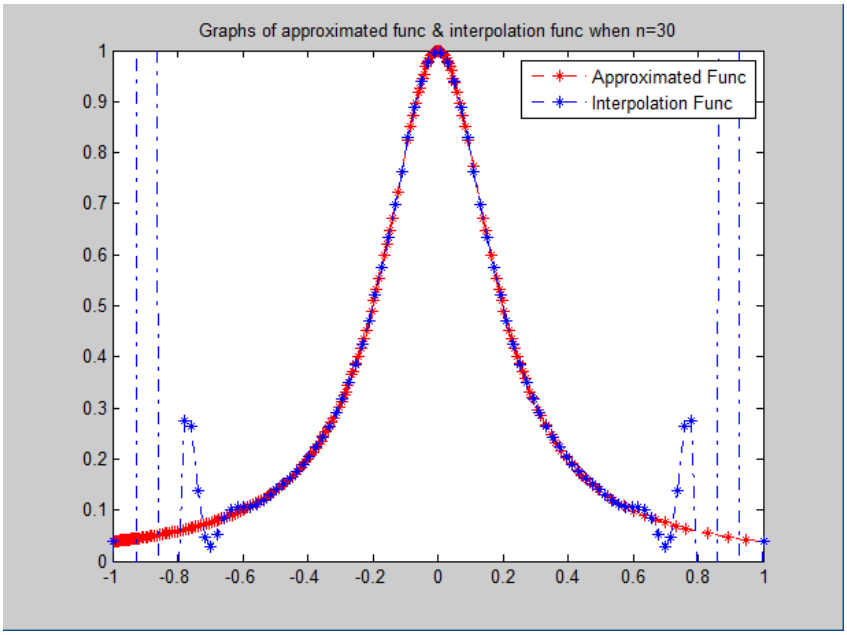


Figure2: Graphs of approximated func & interpolation func when n=30

In fact, the error between generating function and the interpolating polynomial is given by

$$f(\mathrm{x})-\mathrm{P}_n(\mathrm{x})=\frac{f^{(\mathrm{n}+1)}(\xi(\mathrm{x}))}{(\mathrm{n}+1)!}\prod_{i=1}^{n+1}(\mathrm{x}-\mathrm{x}_i) \tag{3}$$

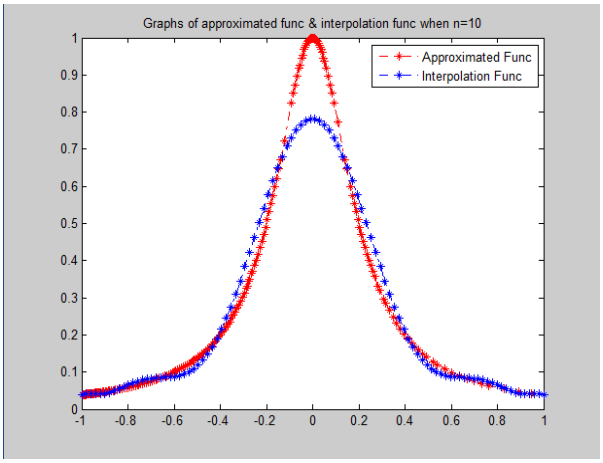
Thus,

$$\max_{-1\le x\le 1}\left|f(\mathrm{x})-P_n(\mathrm{x})\right|\le\max_{-1\le x\le 1}\frac{\left|f^{(\mathrm{n}+1)}(\mathrm{x})\right|}{(\mathrm{n}+1)!}\max_{-1\le x\le 1}\prod_{i=1}^{n}\left|x-x_i\right| \tag{4}$$

In the case of interpolation at equidistant points of Runge function, each of the two multipliers in the upper bound of approximation error reaches infinity when *n* increases. Specifically, for a certain point x_0, the nth derivative of *f(x)* and product of $(x-x_i)$ grows larger and larger with *n*.

Naturally, we can **interpolate non-equidistant points** to mitigate Runge phenomenon.

Let $x_k=\cos\frac{2k+1}{2(\mathrm{n}+1)}\pi,\ k=0,1,2,\cdots,\mathrm{n}$ (n = 10, 20) be the interpolating points, and the graphs of interpolating polynomial are depicted as follow.



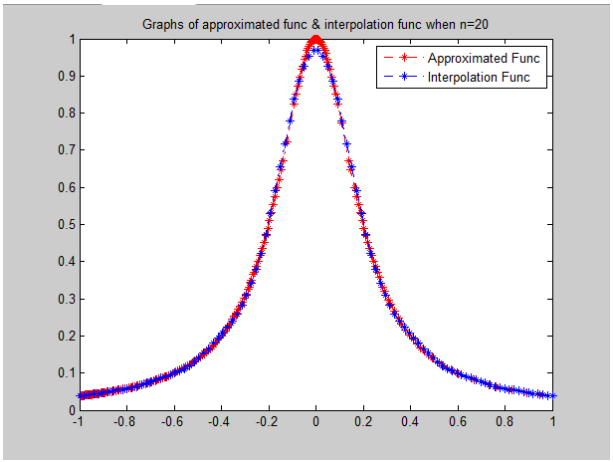


Figure3: Graphs of approximated func & non equidistant points interpolated func when n=10 and 20

Runge phenomenon has been mitigated to a large extent! Furthermore, when n increases to 30, the resulting interpolation fits perfectly to Runge function.

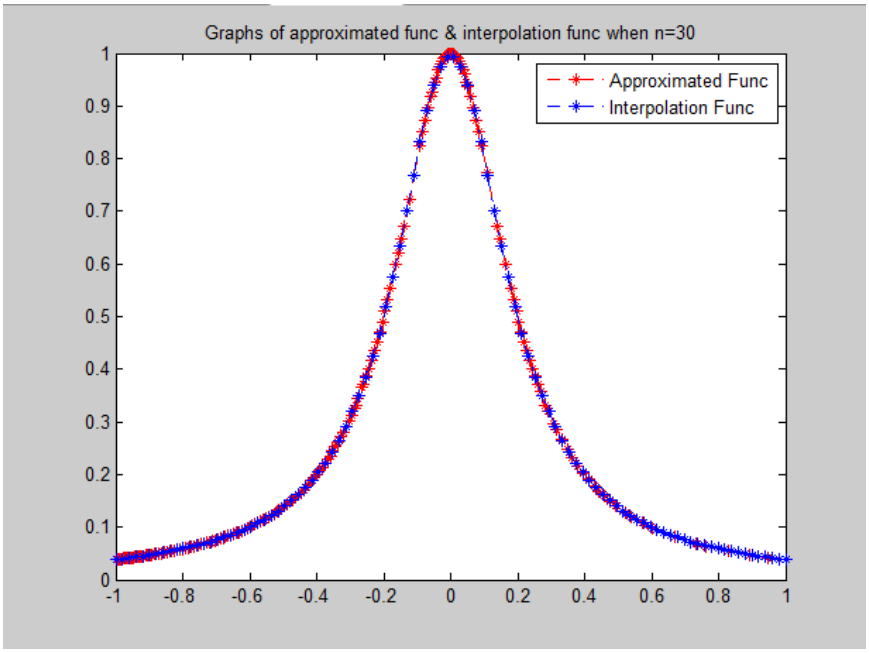


Figure4: Graphs of approximated func & non equidistant points interpolated func when n=30

2.2 Piecewise Linear Interpolation

On the other hand, we manage to **eliminate the nth derivative in the error function** (one of the two multipliers which are growing to infinity with n) by adopting linear interpolation in isometric interval within *[-1,1]*.

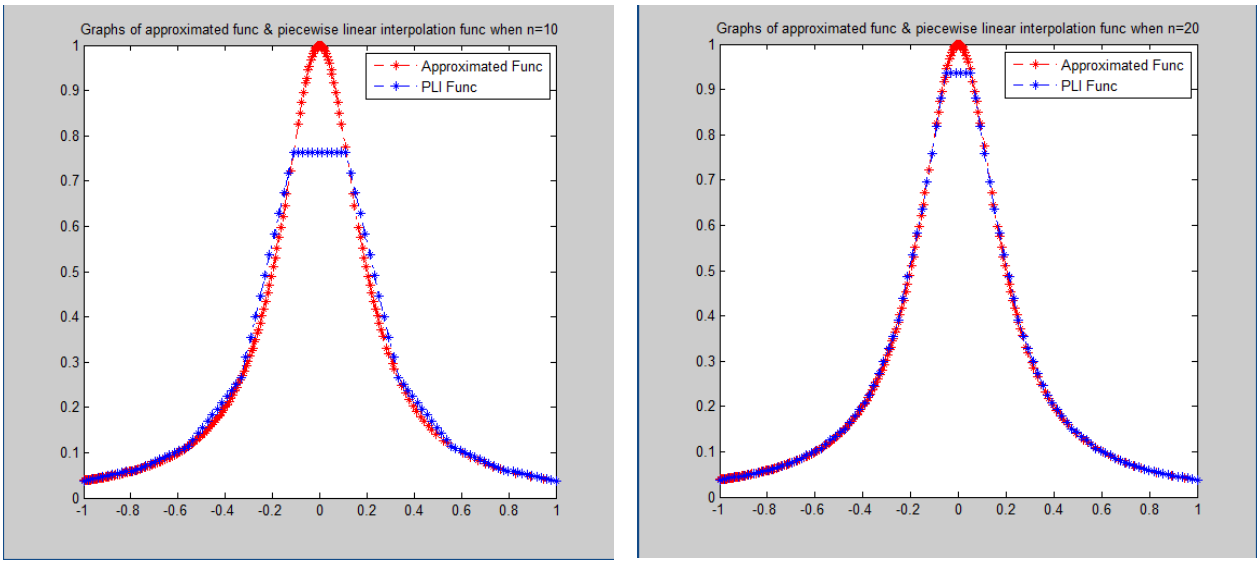


Figure5: Graphs of approximated func & piecewise linear interpolation func when n=10 and 20

The PLI approximates well only to part (which can be counted as linear) of the generating function, but the highly nonlinear part hasn't been well interpolated. Even if we divide the interval into more parts (n=30), the highly nonlinear central part of Runge function still cannot be perfectly fitted.

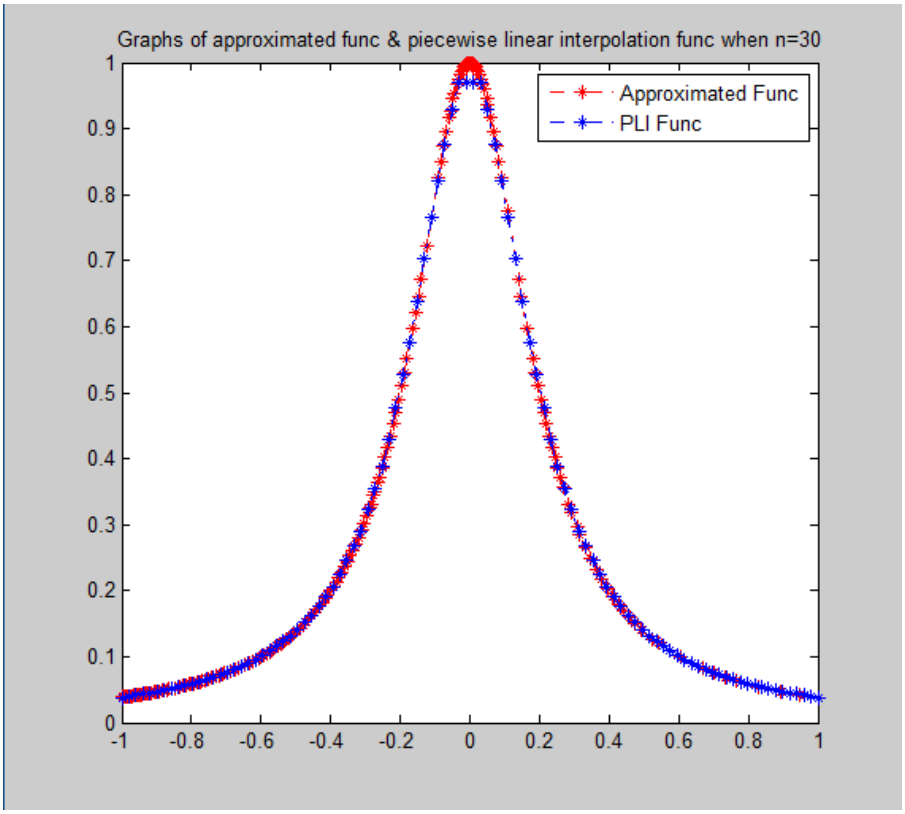


Figure6: Graphs of approximated func & piecewise linear interpolation func when n=30

This can be mitigated by **densifying divisions of the interval where Runge function shows its strong nonlinearity** such as *[0.4, 0.6] and [-0.08, 0.08]*.

2.3 Least Square Approximation with orthogonal Polynomials

In this section, we manage to compute least square orthogonal polynomials of degree 3 and 5. For one thing, least squares approximation may best fit a collection a data. For another, orthogonal polynomials can avoid round off errors difficulties when solving normal equations.

The set of polynomial functions $\{\phi_0,\phi_1,\ldots,\phi_n\}$ defined in the following way is orthogonal on [a, b], with respect to the weight function ω .

$$\phi_0(\mathrm{x})\equiv 1,\quad \phi_1(\mathrm{x})=\mathrm{x}-\mathrm{B}_1,\quad \textit{for each } x \textit{ in } [\mathrm{a},\mathrm{b}] \tag{5}$$

where

$$B_1=\frac{\sum_{i=0}^{m} x_i\omega(\mathrm{x}_i)\left[\phi_0(\mathrm{x}_i)\right]^2}{\sum_{i=0}^{m}\omega(\mathrm{x}_i)\left[\phi_0(\mathrm{x}_i)\right]^2}, \tag{6}$$

and when $k\geq 2$,

$$\phi_k(\mathrm{x})=(\mathrm{x}-B_k)\phi_{k-1}(\mathrm{x})-C_k\phi_{k-2}(\mathrm{x}),\quad \text{for each } x \textit{ in } [\mathrm{a},\mathrm{b}] \tag{7}$$

where

$$B_k=\frac{\sum_{i=0}^{m} x_i\omega(\mathrm{x}_i)\left[\phi_{k-1}(\mathrm{x}_i)\right]^2}{\sum_{i=0}^{m}\omega(\mathrm{x}_i)\left[\phi_{k-1}(\mathrm{x}_i)\right]^2},\quad k=2,3,\cdots,\mathrm{n} \tag{8}$$

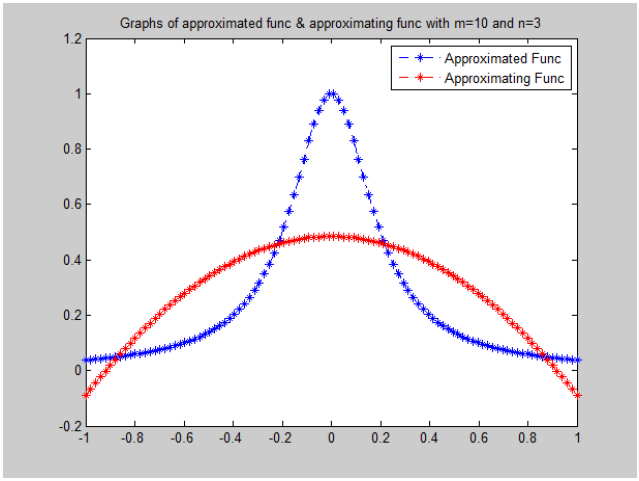
and

$$C_k=\frac{\sum_{i=0}^{m} x_i\omega(\mathrm{x}_i)\phi_{k-1}(\mathrm{x}_i)\phi_{k-2}(\mathrm{x}_i)}{\sum_{i=0}^{m}\omega(\mathrm{x}_i)\left[\phi_{k-2}(\mathrm{x}_i)\right]^2},\quad k=2,3,\cdots,n \tag{9}$$

Then the solution to normal equation is

$$a_k^*=\frac{\sum_{i=0}^{m}\omega(\mathrm{x}_i)\,\mathrm{f}(\mathrm{x}_i)\phi_k(\mathrm{x}_i)}{\sum_{i=0}^{m}\omega(\mathrm{x}_i)[\phi_k(\mathrm{x}_i)]^2},\quad k=0,1,2,\cdots,n \tag{10}$$

Though normal equation may get ill-conditioned while n grows, recruiting orthogonal polynomials can solve it directly by obtaining division of two summations which consist of weight function, collected data and approximating polynomials.



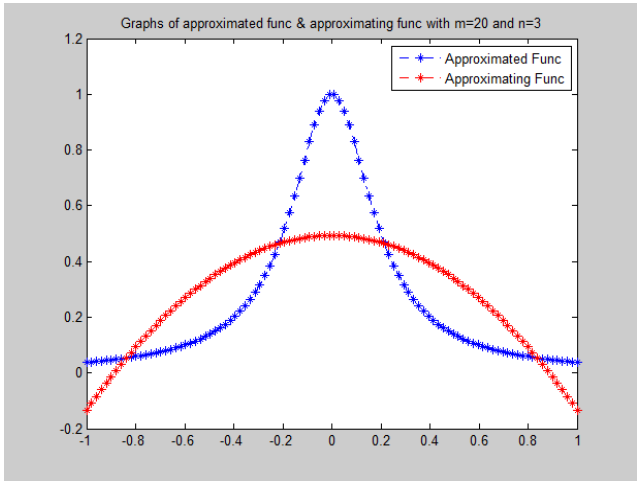
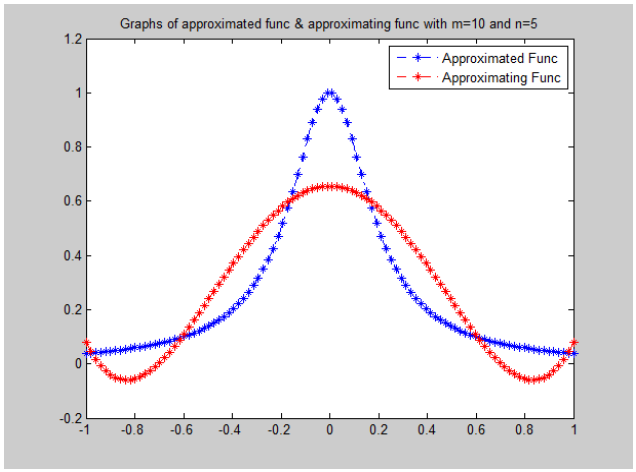


Figure7: Graphs of approximated func & approximating func with m=10, 20 and n=3

A degree of 3 is too small for the polynomial to approximate the Runge function, and it cannot be improved even if we adopt more points. This will be modified when we **elevate the degree of polynomial,** for example, to 5.



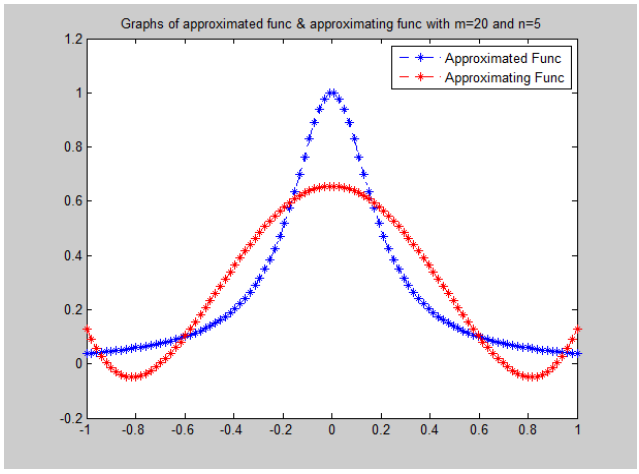


Figure8: Graphs of approximated func & approximating func with m=10, 20 and n=5

Our result polynomial fits better with higher degree, but it does not seem better when we augment the size of collection of data $\{(x_i, y_i)\}_{i=1}^{m}$.

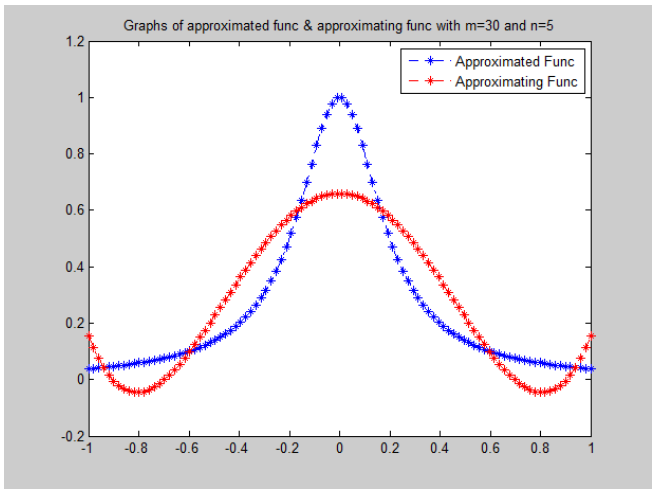


Figure9: Graphs of approximated func & approximating func with m=30 and n=5

However, is augmentation of number of points actually of no use? Here is a graph of polynomial of 10 degrees and its approximated function.

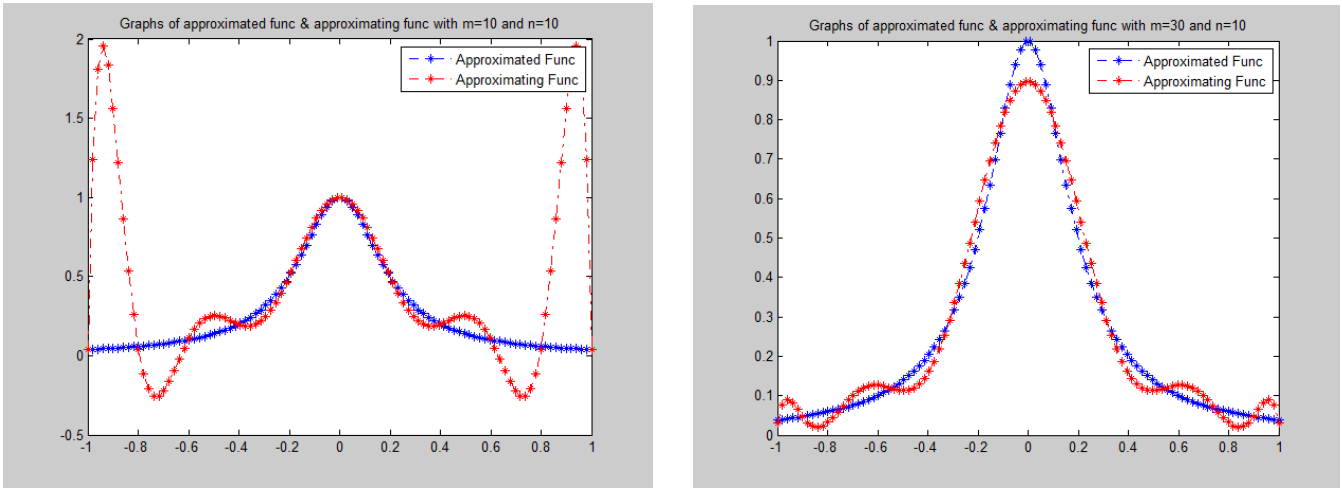


Figure10: Graphs of approximated func & approximating func with m=10, 30 and n=10

Runge phenomenon appears, though the resulting function fits well to the center part of Runge function. When we **increase our points**, Runge phenomenon diminishes! This is so interesting a phenomenon!

It is natural to **increase both number of collection of data and polynomial degree** and it turns out exactly as what we expected – least square approximation fits best the generating function with appropriate m and n.

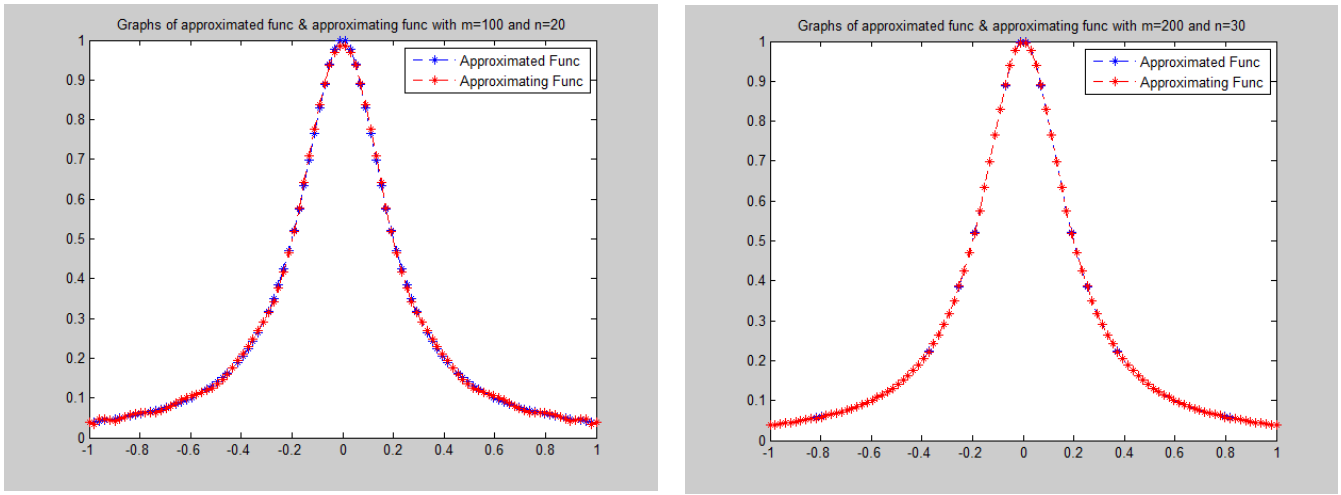


Figure11: Graphs of approximated func & approximating func with m=100, n=20 and m=200, n=30

3 Conclusions

Interpolating equidistant points can lead to Runge phenomenon, especially when n grows bigger, owing to two factors – the nth derivative of f(x) (denoted factor_1) and continual product of (x-xi) (denoted factor_2).

On one hand, we interpolate non equidistant points like $x_k = \cos\frac{2k+1}{2(n+1)}\pi,\ k = 0,1,2,\cdots,n$ (n = 10, 20) to eliminate factor_2 and it turns out satisfying when n increases adequately to a big number such as 30. On the other, we adopt piecewise linear interpolation to eliminate factor_1. Runge function will only be well approximated when dense intervals are created especially where it shows strong nonlinearity.

Least Square Approximation fits best to the generating function and we utilize orthogonal polynomials to avoid round off errors caused by ill-conditioned matrix of normal equation. As a result, large size of collection of data will eliminate Runge phenomenon and a high polynomial degree can guarantee a better approximation. Both factors are essential to achieve a well-performed result polynomial.

4 References

- [1] John D. Cook Runge Phenomenon [DB/OL] https://www.johndcook.com/blog/2017/11/18/runge-phenomena/
- [2] Burden R L, Faires J D. Numerical analysis (7th)[J]. Prindle Weber and Schmidt, Boston, 2001.

5 Appendix

Please refer to my website https://github.com/zhangchi0923/Numerical-Analysis-Experiments for source codes and pics.