Numerical Analysis / Numerical Methods

Spring 2023 Lectures 3-4 Review

Review of interpolation and approximation

PURPOSES OF INTERPOLATION

1. Replace a set of data points

$$\{(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)\}$$

with a function given analytically. Usually this is a polynomial function. In other words, **Purpose 1** consists in finding a polynomial of degree n that passes every given data point.

PURPOSES OF INTERPOLATION

1. Replace a set of data points

$$\{(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)\}$$

   with a function given analytically. Usually this is a polynomial function. In other words, **Purpose 1** consists in finding a polynomial of degree n that passes every given data point.

2. Approximate functions (at least continuous) with simpler ones, usually polynomials or ‘piecewise polynomials’. Here interpolation is used to form polynomials that accurately approximate continuous functions, and next question is how accurately we can do it?

Polynomial interpolation

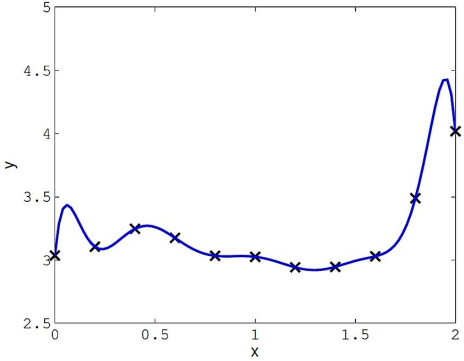
Once data set with $n+1$ points $\{(x_i, y_i)\}_{i=0}^{n}$ is given, there is a UNIQUE polynomial $P_n(x)$ of degree at most n that passes them:

$$P_n(x_i) = y_i, \quad i = 0, 1, \dots n$$

Polynomial interpolation

Once data set with $n+1$ points $\{(x_i, y_i)\}_{i=0}^{n}$ is given, there is a UNIQUE polynomial $P_n(x)$ of degree at most n that passes them:

$$P_n(x_i) = y_i, \quad i = 0, 1, \ldots n$$



Polynomial of degree 10 passing 11 data points

Polynomial interpolation

This UNIQUE polynomial can be obtained by (see Lecture 3 Reading):

1. Lagrange formula:

$$P_n(x) = \sum_{i=0}^{n} y_i L_i(x), \quad \text{where } L_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}.$$

2. Newton difference formula

$$P_n(x) = P_{n-1}(x) + \underbrace{f[x_0, x_1, x_2, \ldots, x_n]}_{\text{Divided difference}}(x - x_0)(x - x_1)\cdots(x - x_{n-1}).$$

Polynomial interpolation

This UNIQUE polynomial can be obtained by (see Lecture 3 Reading):

1. Lagrange formula:

$$P_n(x) = \sum_{i=0}^{n} y_i L_i(x), \quad \text{where } L_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}.$$

2. Newton difference formula

$$P_n(x) = P_{n-1}(x) + \underbrace{f[x_0, x_1, x_2, \ldots, x_n]}_{\text{Divided difference}}(x - x_0)(x - x_1)\cdots(x - x_{n-1}).$$

Never use Lagrange formula for practical reasons!

So, we have essentially solved the polynomial interpolation problem of discrete data perfectly.

Approximation Purpose

Polynomial approximation problem

Given a function $f(x)$ on $[a, b]$, usually at least continuous, find a polynomial $P(x)$ of certain degree that accurately approximates it.

Approximation Purpose

Polynomial approximation problem

Given a function $f(x)$ on $[a, b]$, usually at least continuous, find a polynomial $P(x)$ of certain degree that accurately approximates it.

We might use Taylor polynomials, but the error increases as we move toward endpoints and it will be of the same sign (see Taylor polynomial discussion in Lecture 1. Part 1)

Approximation Purpose

Polynomial approximation problem

Given a function $f(x)$ on $[a, b]$, usually at least continuous, find a polynomial $P(x)$ of certain degree that accurately approximates it.

We might use Taylor polynomials, but the error increases as we move toward endpoints and it will be of the same sign (see Taylor polynomial discussion in Lecture 1. Part 1)

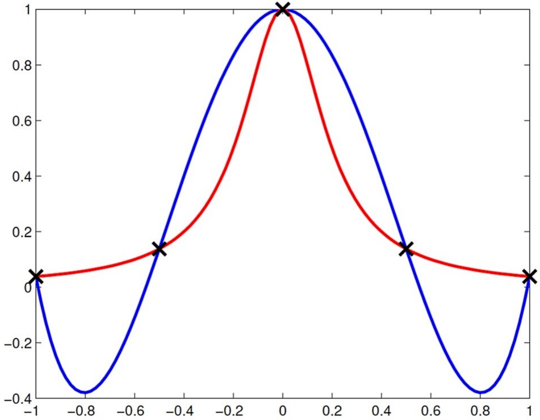
Another approach is by interpolation:

Consider the partition of $[a, b]$: $\quad a \leqslant x_0 < x_1 < x_2 < \cdots < x_n \leqslant b$

and the data points $\quad \{(x_0, f(x_0)), (x_1, f(x_1)), \ldots, (x_n, f(x_n))\}$.

Construct the interpolation polynomial passing these data points to get $P_n(x)$.

Interpolation for Function Approximation



Function $f(x) = \frac{1}{1+25x^2}$ (red) and polynomial $P_4(x)$ (blue) passing 5 data points (black)

Polynomial approximation clearly depends on the interpolation partition considered.

Interpolation for Function Approximation

Approximation accuracy is given by error $e(x) = f(x) - P_n(x)$.

Interpolation for Function Approximation

Approximation accuracy is given by error $e(x) = f(x) - P_n(x)$.
One measure of the error is its **infinity norm**:

$$\|e(x)\|_\infty = \|f(x) - P_n(x)\|_\infty = \max_{x\in[a,b]} \left|f(x) - P_n(x)\right|$$

Interpolation for Function Approximation

Approximation accuracy is given by error $e(x) = f(x) - P_n(x)$.
One measure of the error is its **infinity norm**:

$$\|e(x)\|_\infty = \|f(x) - P_n(x)\|_\infty = \max_{x \in [a,b]} \left| f(x) - P_n(x) \right|$$

Error for polynomial interpolation is given by:

$$f(x) - P_n(x) = \frac{(x - x_0)(x - x_1)\ldots(x - x_n)}{(n+1)!} f^{(n+1)}(\xi) = \frac{\Psi_n(x)}{(n+1)!} f^{(n+1)}(\xi),$$

where ξ is some point on $[a, b]$.

Interpolation for Function Approximation

Approximation accuracy is given by error $e(x) = f(x) - P_n(x)$.
One measure of the error is its **infinity norm**:

$$\|e(x)\|_\infty = \|f(x) - P_n(x)\|_\infty = \max_{x \in [a,b]} \left| f(x) - P_n(x) \right|$$

Error for polynomial interpolation is given by:

$$f(x) - P_n(x) = \frac{(x - x_0)(x - x_1) \dots (x - x_n)}{(n+1)!} f^{(n+1)}(\xi) = \frac{\Psi_n(x)}{(n+1)!} f^{(n+1)}(\xi),$$

where ξ is some point on $[a, b]$.

$$\left| f(x) - P_n(x) \right| \leqslant \frac{\left| \Psi_n(x) \right|}{(n+1)!} \cdot \max_{x \in [a,b]} \left| f^{(n+1)}(x) \right|.$$

Behavior (distribution and magnitude) of error depends on the shape of function $\Psi_n(x)$ (see Lecture 3 Reading)

Interpolation for Function Approximation

Generally, if we have more interpolation points, and hence the degree of the interpolation polynomial is higher, then the error will be smaller:

$$\|e(x)\|_\infty = \|f(x) - P_n(x)\|_\infty \to 0 \text{ as } n \to \infty.$$

Interpolation for Function Approximation

Generally, if we have more interpolation points, and hence the degree of the interpolation polynomial is higher, then the error will be smaller:

$$\|e(x)\|_\infty = \|f(x) - P_n(x)\|_\infty \to 0 \text{ as } n \to \infty.$$

But ... interpolation points should be chosen appropriately.

Interpolation for Function Approximation

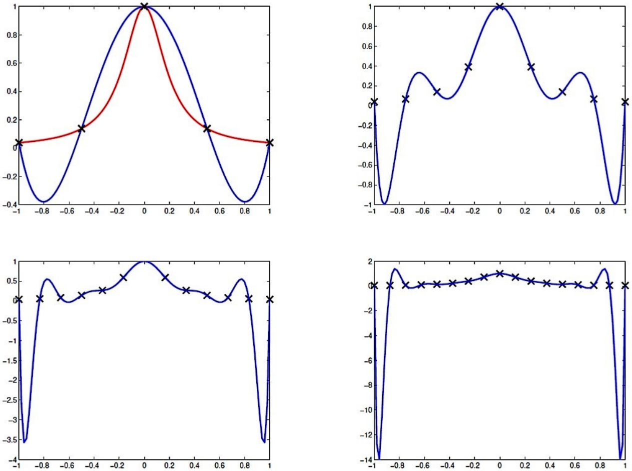
Generally, if we have more interpolation points, and hence the degree of the interpolation polynomial is higher, then the error will be smaller:

$$\|e(x)\|_\infty = \|f(x) - P_n(x)\|_\infty \to 0 \text{ as } n \to \infty.$$

But ... interpolation points should be chosen appropriately.

If the interpolation partition is uniform (equally spaced),
then pathological Runge's example might happen (see Lecture 3 Reading).

Another Runge's Example



$f(x) = \frac{1}{1+25x^2}$ (red) and its interpolants on evenly spaced points P_4, P_8, P_{12} and P_{16}

Approximation Problem

Can build different approximating polynomials of the same degree n for the same $f(x)$:

Approximation Problem

Can build different approximating polynomials of the same degree n for the same $f(x)$:

- Taylor polynomial of degree n.

Approximation Problem

Can build different approximating polynomials of the same degree n for the same $f(x)$:

- Taylor polynomial of degree n.
- Interpolation polynomials (based on various $n + 1$ data points).

Approximation Problem

Can build different approximating polynomials of the same degree n for the same $f(x)$:

- Taylor polynomial of degree n.
- Interpolation polynomials (based on various $n+1$ data points).

Approximation Problem

Can build different approximating polynomials of the same degree n for the same $f(x)$:

- Taylor polynomial of degree n.
- Interpolation polynomials (based on various $n+1$ data points).

Question

Do we have the best approximation among all polynomials of degree n for a given function $f(x)$?

Approximation Problem

Can build different approximating polynomials of the same degree n for the same $f(x)$:

- Taylor polynomial of degree n.
- Interpolation polynomials (based on various $n+1$ data points).

Question

Do we have the best approximation among all polynomials of degree n for a given function $f(x)$?

Best means the smallest error measure:

$$\rho_n(f) = \min_{P_n} \|f(x) - P_n(x)\|_\infty = \min_{P_n} \max_{x\in[a,b]} |f(x) - P_n(x)|.$$

Approximation Problem

Can build different approximating polynomials of the same degree n for the same $f(x)$:

- Taylor polynomial of degree n.
- Interpolation polynomials (based on various $n+1$ data points).

Question

Do we have the best approximation among all polynomials of degree n for a given function $f(x)$?

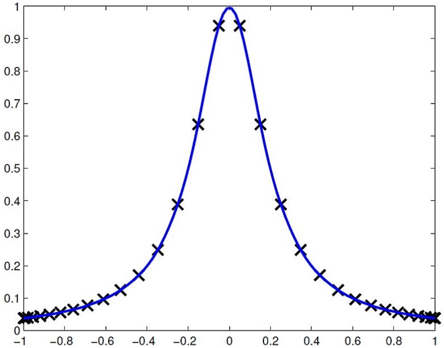
Best means the smallest error measure:

$$\rho_n(f) = \min_{P_n} \|f(x) - P_n(x)\|_\infty = \min_{P_n} \max_{x\in[a,b]} |f(x) - P_n(x)|.$$

Answer is YES!
(Best approximation, Lecture 4 Reading), but practical implementation is cumbersome.

Near-Minimax Approximation

A practical alternative for best approximation is the near-minimax approximation (known as Chebyshev approximation) based on Chebyshev polynomials (Lecture 4).



Chebyshev approximation of the function $f(x) = \frac{1}{1+25x^2}$

Notice that, using Chebyshev points, we overcame the pathology of Runge's Example.

Lebesgue constant

In other words, interpolation points chosen for approximation have big effect on how accurately the interpolation polynomial approximates $f(x)$.

Lebesgue constant

In other words, interpolation points chosen for approximation have big effect on how accurately the interpolation polynomial approximates $f(x)$.

Chebyshev interpolation points were chosen in order to minimize the interpolation error formula for $|f(x) - P_n(x)|$.

,

Lebesgue constant

In other words, interpolation points chosen for approximation have big effect on how accurately the interpolation polynomial approximates $f(x)$.

Chebyshev interpolation points were chosen in order to minimize the interpolation error formula for $|f(x) - P_n(x)|$.

'This formula depends on function $f(x)$ and interpolation partition.

Lebesgue constant

In other words, interpolation points chosen for approximation have big effect on how accurately the interpolation polynomial approximates $f(x)$.

Chebyshev interpolation points were chosen in order to minimize the interpolation error formula for $|f(x) - P_n(x)|$.

'This formula depends on function $f(x)$ and interpolation partition.

On the other hand, we would prefer to have a measure of interpolation accuracy that is independent of f.

Lebesgue constant

In other words, interpolation points chosen for approximation have big effect on how accurately the interpolation polynomial approximates $f(x)$.

Chebyshev interpolation points were chosen in order to minimize the interpolation error formula for $|f(x) - P_n(x)|$.

'This formula depends on function $f(x)$ and interpolation partition.

On the other hand, we would prefer to have a measure of interpolation accuracy that is independent of f.

Something that would measure the quality of interpolation points. This is provided by so-called **Lebesgue constant**.

Lebesgue constant

Definition

Let $\mathcal{P}$ denote a set of interpolation points:

$$\mathcal{P} = \{x_0, x_1, \ldots, x_n\} \subset [a, b].$$

The Lebesgue constant of $\mathcal{P}$ is defined as,

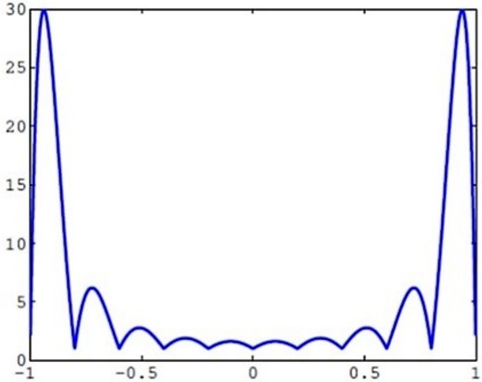
$$\Lambda_n(\mathcal{P}) = \max_{x \in [a,b]} \sum_{k=0}^{n} |L_k(x)|,$$

where $L_k(x)$ are the Lagrange basis functions associated with interpolation set $\mathcal{P}$.

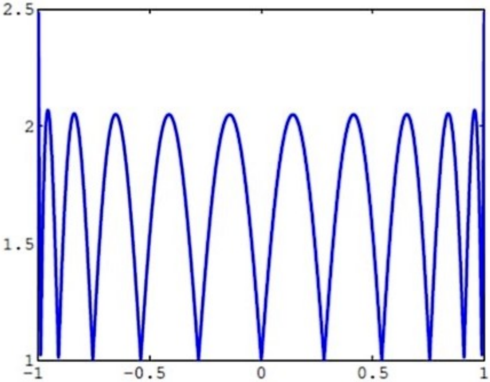
Small Lebesgue constant means that our interpolation can't be much worse that the best polynomial approximation!

Lebesgue constant

Plot of $\sum_{k=0}^{10} |L_k(x)|$ for $\mathcal{P}_{unif}$ and $\mathcal{P}_{cheb}$ with 11 data points in $[-1, 1]$.



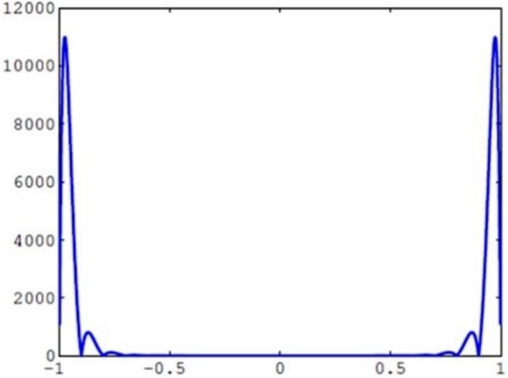
$\Lambda_{10}(\mathcal{P}_{unif}) \approx 29.9$



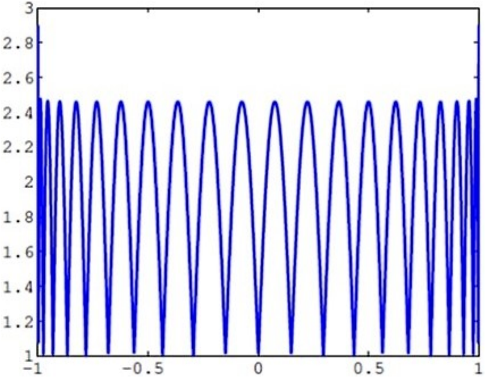
$\Lambda_{10}(\mathcal{P}_{cheb}) \approx 2.49$

Lebesgue constant

Plot of $\sum_{k=0}^{20} |L_k(x)|$ for $\mathcal{P}_{unif}$ and $\mathcal{P}_{cheb}$ with 21 data points in $[-1, 1]$.



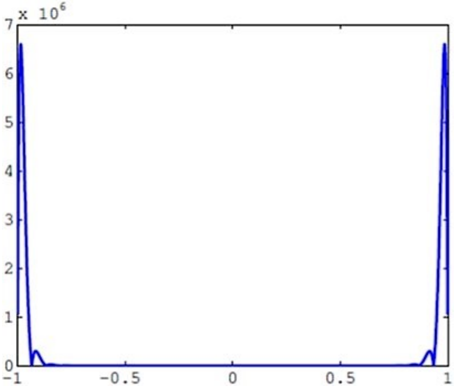
$\Lambda_{20}(\mathcal{P}_{unif}) \approx 10\,987$

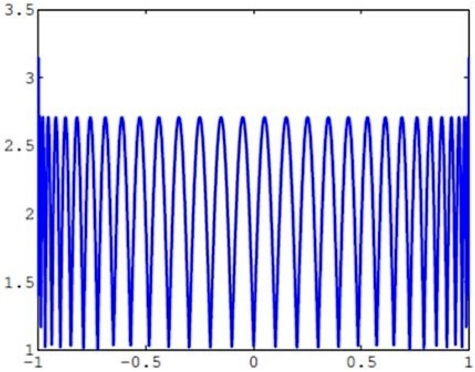


$\Lambda_{20}(\mathcal{P}_{cheb}) \approx 2.9$

Lebesgue constant

Plot of $\sum_{k=0}^{30} |L_k(x)|$ for $\mathcal{P}_{unif}$ and $\mathcal{P}_{cheb}$ with 31 data points in $[-1, 1]$.





$\Lambda_{30}(\mathcal{P}_{unif}) \approx 6\,600\,000$

$\Lambda_{30}(\mathcal{P}_{cheb}) \approx 3.15$

Lebesgue constant

The explosive growth of $\Lambda_n(\mathcal{P}_{unif})$ is an explanation for Runge example pathology.

Lebesgue constant

The explosive growth of $\Lambda_n(\mathcal{P}_{unif})$ is an explanation for Runge example pathology.

Also, it has been shown that as $n \to \infty$,

$$\Lambda_n(\mathcal{P}_{unif}) \approx \frac{2^n}{e\, n \log n}, \qquad \text{BAD!}$$

Lebesgue constant

The explosive growth of $\Lambda_n(\mathcal{P}_{unif})$ is an explanation for Runge example pathology.

Also, it has been shown that as $n \to \infty$,

$$\Lambda_n(\mathcal{P}_{unif}) \approx \frac{2^n}{e\, n \log n}, \qquad \text{BAD!}$$

whereas

$$\Lambda_n(\mathcal{P}_{cheb}) < \frac{2}{\pi} \log(n+1) + 1. \qquad \text{GOOD!}$$

Conclusions

1. **Polynomial interpolation purpose 1 (fitting discrete data)**

Conclusions

1. **Polynomial interpolation purpose 1 (fitting discrete data)**
   - There is a unique polynomial $P_n(x)$ that fits the data.

Conclusions

1. **Polynomial interpolation purpose 1 (fitting discrete data)**
   - There is a unique polynomial $P_n(x)$ that fits the data.
   - Should use Newton divided difference formula.

Conclusions

1. **Polynomial interpolation purpose 1 (fitting discrete data)**
   - There is a unique polynomial $P_n(x)$ that fits the data.
   - Should use Newton divided difference formula.
   - Avoid equally spaced (uniform) partition of data points.

Conclusions

1. **Polynomial interpolation purpose 1 (fitting discrete data)**
   - There is a unique polynomial $P_n(x)$ that fits the data.
   - Should use Newton divided difference formula.
   - Avoid equally spaced (uniform) partition of data points.
2. **Polynomial interpolation purpose 2 (approximating functions)**

Conclusions

1. **Polynomial interpolation purpose 1 (fitting discrete data)**
   - There is a unique polynomial $P_n(x)$ that fits the data.
   - Should use Newton divided difference formula.
   - Avoid equally spaced (uniform) partition of data points.
2. **Polynomial interpolation purpose 2 (approximating functions)**
   - For a given set of interpolation points, use point 1 methodology to build the approximation $P_n(x)$.

Conclusions

1. **Polynomial interpolation purpose 1 (fitting discrete data)**
   - There is a unique polynomial $P_n(x)$ that fits the data.
   - Should use Newton divided difference formula.
   - Avoid equally spaced (uniform) partition of data points.
2. **Polynomial interpolation purpose 2 (approximating functions)**
   - For a given set of interpolation points, use point 1 methodology to build the approximation $P_n(x)$.
   - Interpolation points play an important role on the size of error $\|f(x) - P_n(x)\|_\infty$ (keep in mind Runge's example).

Piecewise polynomial interpolation

Problem

Is it possible, given a set of data points, to build a function (piecewise polynomial) s.t. the "shape" of the data is preserved?

Piecewise polynomial interpolation

Problem

Is it possible, given a set of data points, to build a function (piecewise polynomial) s.t. the "shape" of the data is preserved?

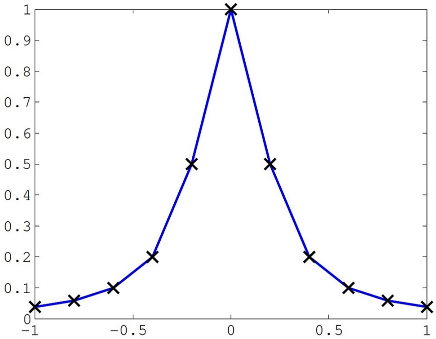
Answer is YES: Cubic spline functions (See Lecture 3 Reading).

Piecewise polynomial interpolation

Problem

Is it possible, given a set of data points, to build a function (piecewise polynomial) s.t. the "shape" of the data is preserved?

Answer is YES: Cubic spline functions (See Lecture 3 Reading).

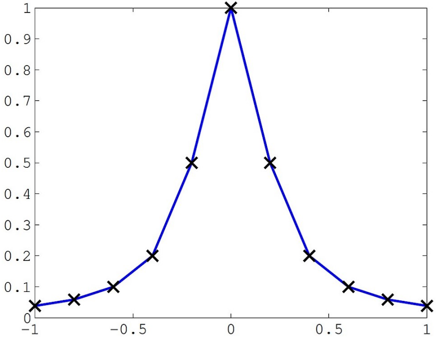


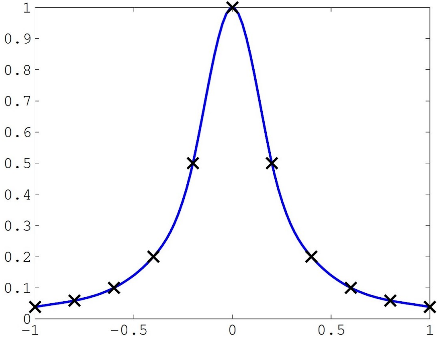
Piecewise polynomial interpolation

Problem

Is it possible, given a set of data points, to build a function (piecewise polynomial) s.t. the “shape” of the data is preserved?

Answer is YES: Cubic spline functions (See Lecture 3 Reading).





Numerical Analysis for Calculus

In the next lectures, we will discuss the development and application of numerical methods to problems of Calculus:

- Integration;

Numerical Analysis for Calculus

In the next lectures, we will discuss the development and application of numerical methods to problems of Calculus:

- Integration;
- Differentiation;

Numerical Analysis for Calculus

In the next lectures, we will discuss the development and application of numerical methods to problems of Calculus:

- Integration;
- Differentiation;
- Solving ODE (ordinary differential equations);

Numerical Analysis for Calculus

In the next lectures, we will discuss the development and application of numerical methods to problems of Calculus:

- Integration;
- Differentiation;
- Solving ODE (ordinary differential equations);
- Optimization.

Numerical Analysis for Calculus

In the next lectures, we will discuss the development and application of numerical methods to problems of Calculus:

- Integration;
- Differentiation;
- Solving ODE (ordinary differential equations);
- Optimization.
- Least square approximation.

Numerical Analysis for Calculus

In the next lectures, we will discuss the development and application of numerical methods to problems of Calculus:

- Integration;
- Differentiation;
- Solving ODE (ordinary differential equations);
- Optimization.
- Least square approximation.
- Data fitting.