MAT128A: Numerical Analysis
Lecture Two: Finite Precision Arithmetic

September 28, 2018

Floating point arithmetic

Computers use finite strings of binary digits to represent real numbers.

Before we discuss the IEEE double precision binary format, which is a standard available on most current computers, we will discuss a decimal floating point number format.

The principles are the same, and it is easier for most people to think about roundoff errors and other numerical issues in decimal.

We will then describe the IEEE double precision binary format in detail.

A decimal floating point format

The form of a number in our floating point decimal format is

$$x = (-1)^s d.dddddd \times 10^{ee-49}$$

where:

- s is either 0 or 1
- $d.dddddd$ represents a string of 7 decimal digits which does not start with 0;
- ee is a string of 2 decimal digits

We call $d.dddddd$ the **mantissa**,

$$ee - 49$$

the **exponent**, and s the **sign bit**.

The form of a number in our floating point decimal format is

$$x = (-1)^s d.dddddd \times 10^{ee-49}$$

The expression $ee - 49$ is known as a biased exponent. It is used for technical reasons — mostly, to make certain operations faster.

The exponent can range from -48 to 49. The values the values $ee = 00$ and $ee = 99$ have special meanings (we'll talk more about this when we discuss IEEE double precision binary format).

In the interests of clarity and simplicity, we will generally write down floating point numbers in the form

$$1.234567 \times 10^{33},$$

rather than

$$1.234567 \times 10^{82-49}$$

and simply keep in mind that the exponent can vary from -48 to 49.

A decimal floating point format

Given a real number x, we denote by $\mathrm{fl}(x)$ the number of the form

$$\pm d.dddddd \times 10^{ee-49}$$

closest to x. In cases in which there are two such numbers, we round down (there are more complicated schemes which are better, but for our purposes, this suffices).

Rounding error

$\mathrm{fl}(x) = x(1+\delta)$ where $|\delta| < u$ with $u = \frac{1}{2}10^{-6}$.

The number $u = \frac{1}{2}10^{-6}$ is called the **unit roundoff.**

A decimal floating point format

Rounding error

$\mathrm{fl}(x) = x(1+\delta)$ where $|\delta| < u$ with $u = \frac{1}{2}10^{-6}$.

Important observation: This bound on the error in our representation of a real number x depends on the magnitude of x.

For instance:

$$|\pi - \mathrm{fl}(\pi)| = |3.1415926535879\ldots - 3.1415927| \approx 2.4 \times 10^{-8}$$

But

$$|\mathrm{fl}(\exp(10)) - \exp(10)| = |22026.4657948067\ldots - 22026.47| \approx 4.2 \times 10^{-3}$$

A decimal floating point format

So we don't have bounds on the **absolute error**

$$|\mathrm{fl}(x) - x|$$

in our representation of the real number x. Instead, we have a bound on the **relative error**

$$\frac{|\mathrm{fl}(x) - x|}{|x|} \leq \frac{1}{2} 10^{-6}.$$

This is exactly what we should expect from approximating a real number using its leading digits.

A decimal floating point format

Adding two numbers:

$$
\begin{aligned}
& \quad 3.141593 \times 10^{00} \\
& + 1.001100 \times 10^{03} \\
& = 0.003141593 \times 10^{03} \\
& + 1.001100000 \times 10^{03} \\
& = 0.0031415 \times 10^{03} \\
& + 1.0011000 \times 10^{03} \\
& = 1.0042415 \times 10^{03} \\
& \approx 1.004242 \times 10^{03}
\end{aligned}
$$

Here, we used an extra digit while performing the addition operation. Most computers do likewise — this extra digit is called a "guard digit."

A decimal floating point format

We will use the notation fl(x+y) to denote the result of performing adding x to y using our floating point number system, and likewise for other arithmetic operations.

Standard model for numerical arithmetic

$\mathrm{fl}(x+y) = (x+y)(1+\delta)$ where $|\delta| < u$

$\mathrm{fl}(x-y) = (x-y)(1+\delta)$ where $|\delta| < u$

$\mathrm{fl}(x*y) = (x*y)(1+\delta)$ where $|\delta| < u$

$\mathrm{fl}(x/y) = (x/y)(1+\delta)$ where $|\delta| < u$

Important observation: these are bounds on the relative, but not absolute accuracy.

Complications which arise from finite precision arithmetic

Each arithmetic operation gives us relative accuracy on the order of the unit roundoff u.

If we conduct a series of arithmetic operations, does this mean the result will always agree with the result obtained via exact arithmetic operations with relative accuracy on the order of u?

Complications which arise from finite precision arithmetic

Each arithmetic operation gives us relative accuracy on the order of the unit roundoff u.

If we conduct a series of arithmetic operations, does this mean the result will always agree with the result obtained via exact arithmetic operations with relative accuracy on the order of u?

No. It absolutely, positively, most certainly does not.

Cancellation errors

What happens when we perform the operations

$$\pi + 10000.01 - 10000.0 = \pi + 00000.01 \approx 3.15159265358979\cdots$$

using our floating point number system?

$$\begin{aligned} & 00003.141593 \\ + \; & 10000.01 \\ = \; & 10003.15 \\ - \; & 10000.00 \\ = \; & 00003.15 \end{aligned}$$

The relative error is pretty bad:

$$\frac{|\pi + 0.1 - 3.15|}{|\pi + 0.1|} \approx .028$$

Cancellation errors

This is called a **cancellation error**.

This is also what goes wrong in the evaluation of the monomial expansion

$$
\begin{aligned}
p(x) = x^{20} &- 210x^{19} + 20615x^{18} - 1256850x^{17} + 53327946x^{16} - 1672280820x^{15} \\
&+ 40171771630x^{14} - 756111184500x^{13} + 11310276995381x^{12} \\
&- 135585182899530x^{11} + 1307535010540395x^{10} - 10142299865511450x^{9} \\
&+ 63030812099294896x^{8} - 311333643161390640x^{7} + 1206647803780373360x^{6} \\
&- 3599979517947607200x^{5} + 8037811822645051776x^{4} \\
&- 12870931245150988800x^{3} + 13803759753640704000x^{2} \\
&- 8752948036761600000x + 2432902008176640000
\end{aligned}
$$

from Lecture 1.

Cancellation errors

We need to be particularly careful to avoid magnifying cancellation errors through multiplication or division as in the next example.

Suppose we wish to evaluate

$$\frac{1-\cos(x)}{x^2}$$

for $x = 1.000000 \times 10^{-3}$. A very high accuracy approximation of this quantity is

$$0.49999995833333472222197420635196208.$$

What happens when we use our decimal floating point system to perform these operations?

We will assume that the cosine function can be evaluated with relative accuracy on the order of unit roundoff, just like basic arithmetic functions.

Cancellation errors

We approximate $\cos(x)$ to obtain

$$\cos(x) \approx 9.999996 \times 10^{00}$$

and we get

$$x^2 \approx 1.000000^{-6}.$$

So the approximation we get of the quantity

$$\frac{1 - \cos(x)}{x^2}$$

is equal to

$$\frac{4 \times 10^{-7}}{10^{-6}} = 4.0000000 \times 10^{-1}.$$

So we are way off from the high accuracy approximation

$$0.49999995833333472222219742063519620 8.$$

This is because the division by 10^{-6} magnified the error we incurred when evaluating $1 - \cos(x)$.

What could we have done to avoid this?

We can often remove problematic cancellation using mathematical analysis.

For instance, we can use a Taylor expansions of $\cos(x)$ to obtain

$$\begin{aligned}\frac{1-\cos(x)}{x^2} &\approx \frac{1}{x^2}\left(1-\left(1-\frac{x^2}{2}+\frac{x^4}{4!}-\frac{x^6}{6!}+\frac{x^8}{8!}\right)\right)\\ &= \left(\frac{1}{2}-\frac{x^2}{4!}+\frac{x^4}{6!}-\frac{x^6}{8!}\right).\end{aligned}$$

We no longer have a problematic subtraction and the error in this expression is on the order of x^8.

Cancellation errors

The naive use of the quadratic formula can also lead to severe cancellation errors.

As you no doubt recall, the two complex-valued roots of $ax^2 + bx + c = 0$ are

$$z = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

When $4|ac|$ is small compared to b^2, the error in the computation of

$$-b \pm \sqrt{b^2 - 4ac}$$

is large, and then it can be magnified by the division by a.

Cancellation errors

The roots of

$$10^{-3}x^2 + 10^7 x + 3 = 0$$

are

$$x_1 = \frac{-10^7 + \sqrt{10^{14} - 12 \cdot 10^{-3}}}{2 \times 10^{-3}} \approx -3 \times 10^{-7}$$

and

$$x_2 = \frac{-10^7 - \sqrt{10^{14} - 12 \cdot 10^{-3}}}{2 \times 10^{-3}} \approx -1.0 \times 10^{10}.$$

There is no difficult computing x_2 numerically, but the approximation of x_1 obtained using our decimal floating point system (and with the double precision arithmetic system used on most computers) is 0. So we do not even get a single correct digit.

Cancellation errors

In this case, we could instead use the formula

$$\frac{-b+\sqrt{b^2-4ac}}{2a} = \frac{\left(-b+\sqrt{b^2-4ac}\right)\left(-b-\sqrt{b^2-4ac}\right)}{2a\left(-b-\sqrt{b^2-4ac}\right)}$$
$$= \frac{-2c}{b+\sqrt{b^2-4ac}},$$

which is perfectly stable.

So here is a case in which algebraic manipulation can be used to avoid numerical cancellation.

Heuristics for avoiding roundoff errors

There is no foolproof method for dealing with cancellation errors. However, most such errors arise from performing operations on quantities of vastly different scales.

Heuristic rule

Try to limit the range of the quantities which arise in your code.

Overflow and underflow

If an arithmetic operation results in a quantity of such large magnitude that it cannot be represented using our format, then **overflow** is said to have occurred. In this case, one of the special values $\pm\infty$ is usually returned to the user and a flag is set. For example, if we tried to perform the operation

$$10^{30} \times 10^{36}$$

using our arithmetic system, the result would be ∞ since 10^{66} is too large to represent using our format.

If the arithmetic operation results in a number whose magnitude is so small that it cannot be represented using our number system, then **underflow** is said to have occurred. As an example,

$$10^{-30} \times 10^{-36}$$

would result in underflow. The value 0 is returned in such cases and a flag in the processor which indicates an underflow exception is set.

Underflow and overflow are comparatively easy to avoid.

IEEE double precision floating point numbers

The IEEE double precision binary format is probably the most widely used floating point number system. The form of a normalized double precision IEEE floating point number is

$$\pm 1.bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb \times 2^{eeeeeeeeeee-1023}$$

where

- the mantissa is a 52 digit binary string and
- *eeeeeeeeee* is an 11 digit binary string

The exponent ranges from -1022 to 1023. The values $eeee = 0$ and $eeee = 2047$ have special meanings. Real numbers are represented

In order to extend the range of numbers which can be represented and to avoid certain problems which arise when subtracting numbers, the IEEE double precision format also also for *subnormal* numbers of the form

$$\pm 0.bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb \times 2^{eeeeeeeeeee-1023}.$$

Each double precision number is represented using 64 bits, which is 8 bytes.

IEEE double precision floating point numbers

IEEE double precision format gives roughly 16 decimal digits of precision. In fact, the unit roundoff is

$$u = 2^{-52} \approx 2.22044604925031 \times 10^{-16}.$$

The largest number which can be represented is

$$1.1111111111111111111111111111111111111111111111111111 \times 2^{1023} \approx 1.8 \times 10^{308}.$$

The smallest positive subnormal number which can be represented is roughly

$$4.9 \times 10^{-324}$$

while the smallest positive normalized number is roughly

$$2.22 \times 10^{-308}.$$

Infinity and NaN

Some arithmetic operations are invalid. For instance, since the floating point units do not support complex values, the expression

$$\sqrt{-1}$$

is meaningless. The value NaN (which stands for not a number) is returned in these cases. NaN is signaled by having bit in the exponent be 1 — as long as the bits in the mantissa are not all 0.

The quantities $\pm\infty$, which arise when overflow occurs or when evaluating expression such as

$$\frac{1}{0} = \infty$$

are signaled by having all the exponent bits be set to 1 and all of the mantissa bits set to 0. The sign bit selects between $\pm\infty$.

Don't implement your own floating point format

The IEEE standards are very well engineered. A great deal of thought was put into how arithmetic operations should be conducted, how 0, ∞ and NaN should be represented, and so on.

There are several infamous examples of compilers and systems whose deviations from the standard (sometimes in apparently minor ways) lead to extremely poor results.

The distribution of floating point numbers

The smallest positive double precision number greater than 0 is roughly

$$4.9 \times 10^{-324},$$

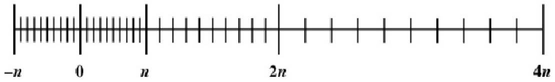
while the smallest double precision number greater than 1 is roughly

$$1 + 2^{-52} \approx 1 + 2.22 \times 10^{-16}.$$

Obviously, double precision numbers are much denser near 0 than near 1.

In fact, they are distributed logarithmically. There are the same number of double precision numbers in each interval of the form

$$\left(2^k, 2^{k+1}\right).$$



MAT128A: Numerical Analysis
Lecture Three: Condition Numbers

October 1, 2018

An auspicious example

Last time, we saw that the naive evaluation of the function

$$f(x) = \frac{1 - \cos(x)}{x^2}$$

for x near 0 leads to numerical cancellation and a loss of accuracy, but that this problem can be easily overcome.

One mechanism for doing so is to approximate $f(x)$ using a Taylor expansion:

$$f(x) \approx \frac{1}{2} - \frac{x^2}{24} + \frac{x^4}{720} - \frac{x^6}{40320} + \frac{x^8}{3628800} - \frac{x^{10}}{479001600} + \cdots$$

An inauspicious example

Let's look at another example: evaluating the function

$$\cos(x)$$

when x is large.

Suppose we wish to evaluate cosine at the argument $x = 10^7\sqrt{2}$. Using the computer algebra system Mathematica, we find that to around 15 digits of precision

$$cos(10^7\sqrt{2}) \approx 0.251079412844212.$$

But when we evaluate the same quantity using double precision arithmetic, we get:

$$cos(10^7\sqrt{2}) \approx 0.251079414230471.$$

The approximation obtained via double precision arithmetic isn't terrible, but we did lose around 6 digits of precision:

$$|0.25107941284421212 - 0.251079414230471| \approx 1.4 \times 10^{-9}$$

An inauspicious example

What happened?

In order to evaluate $\cos(x)$, the computer first finds the value of x modulo 2π. That is, it calculates $0 \leq y < 2\pi$ such that

$$x = 2\pi k + y \quad \text{with} \quad k \text{ an integer.}$$

Since cosine is periodic with period 2π,

$$\cos(x) = \cos(y)$$

and the computer next uses this identity to calculate $cos(x)$.

There is no significant loss of precision when evaluating $\cos(y)$. For small arguments, cosine can be evaluated with essentially machine precision accuracy.

An inauspicious example

The loss of precision comes from computing the argument modulo 2π. This involves subtracting a large multiple of 2π from x, which leads to a cancellation error.

Indeed, if this computation is performed using double precision arithmetic, we get

$$10^7\sqrt{2} \approx 2250790 \times 2\pi + 4.96618421189487,$$

whereas

$$10^7\sqrt{2} \approx 2250790 \times 2\pi + 4.96618420904161$$

is an approximation accurate to 15 digits.

The computer then calculates the value of $\cos(4.96618421189487)$ accurately, but the damage has already been done at this point.

An inauspicious example

You would be forgiven for thinking there might be a way to compute $\cos(x)$ without calculating $\mathrm{mod}(x, 2\pi)$), and that this could provide a means to evaluate $\cos(x)$ to higher accuracy than we compute $\mathrm{mod}(x, 2\pi)$.

Alas, there is no way to compute $\cos(x)$ without also being able to accurately compute the modulus of x. If

$$x = 2\pi k + y$$

and arccos is defined in the usual way, then

$$\mathrm{mod}(x, 2\pi) = y = \arccos(\cos(x)).$$

It turns out that arccos can be evaluated with relative accuracy on the order of 15 digits, so this means that the accuracy with which we can evaluate

$$\mathrm{mod}\ (x, 2\pi)$$

is intrinsically tied to accuracy with which we can evaluate

$$\cos(x).$$

An inauspicious example

This is a much more serious problem than in the first example, where we were able to easily obtain a method for accurately approximating the function f.

Given the double precision representation of a real number x of large magnitude, there is simply no way to evaluate $\text{mod}(x, 2\pi)$ to high accuracy. The information is simply not there — we would require knowledge of more than 15 digits of x to evaluate $\text{mod}(x, 2\pi)$ with 15 digits of accuracy.

The condition number of a function

In order to distinguish between cases in which we can (at least in theory) correct a roundoff or other numerical error and cases in which we cannot, we will introduce a notion called the "condition number of the function f at the point x."

The condition number of evaluation of f at the point x is a measure of the ratio of the relative change in a function $f(x)$ to the relative change in x.

Part of the intuition for trying to measure this ratio is that we will also have a relative error in the argument of x which is at least on the order of machine precision, so the best relative error in the evaluation of $f(x)$ we can ever hope to achieve will be on the order of

$$(\text{ratio of relative error in f to relative error in x}) \times (\text{machine precision})$$

The condition number of a function

If we perturb x by a quantity h of small magnitude, then the relative change in $f(x)$ is

$$\frac{f(x+h)-f(x)}{f(x)}$$

and the relative change in x is

$$\frac{x+h-x}{x}=\frac{h}{x},$$

so we wish to consider the ratio

$$\left|\frac{f(x+h)-f(x)}{f(x)}\Big/\frac{h}{x}\right|=\left|\frac{f(x+h)-f(x)}{h}\times\frac{x}{f(x)}\right|$$

for values of h of small magnitude.

The condition number of a function

It would be nice to be able to bound the ratio

$$\left|\frac{f(x+h)-f(x)}{f(x)} \Big/ \frac{h}{x}\right| = \left|\frac{f(x+h)-f(x)}{h} \times \frac{x}{f(x)}\right|$$

for all x and small h. This is usually too difficult, so we instead define the condition number $\kappa_f(x)$ of evaluation of f at the point x by taking limit of this ratio as $h \to 0$. This gives us a way to prove bounds which will hold for all sufficient small h given smoothness/continuity assumptions on f and f'.

Condition number

The condition number of the function f at the point x is

$$\kappa_f(x) = \lim_{h\to 0}\left|\frac{f(x+h)-f(x)}{h} \times \frac{x}{f(x)}\right| = \left|\,x\,\frac{f'(x)}{f(x)}\right|$$

The condition number of a function

Condition number

The condition number of the function f at the point x is

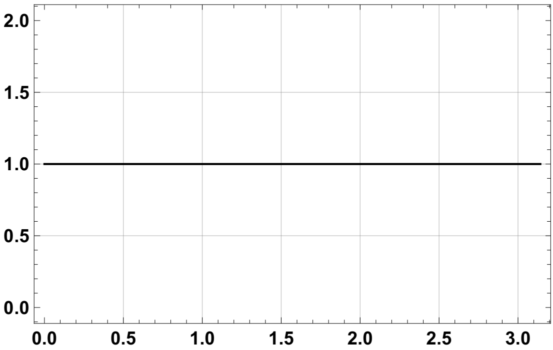
$$\kappa_f(x) = \lim_{h \to 0} \left| \frac{f(x+h) - f(x)}{h} \times \frac{x}{f(x)} \right| = \left| x \, \frac{f'(x)}{f(x)} \right|$$

Interpretation

If $\kappa_f(x) = 10^k$ then we expect to lose around k decimal digits of precision when evaluating f at the point x.

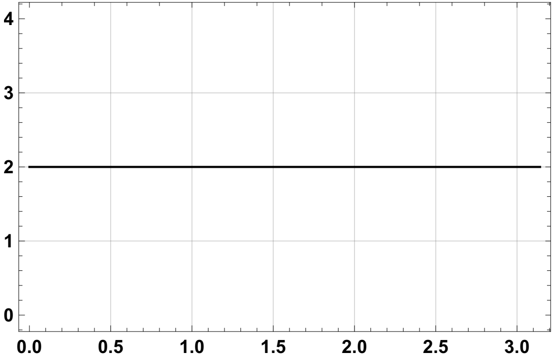
The condition numbers of certain functions

$$f(x) = x, \qquad \kappa(x) = 1$$



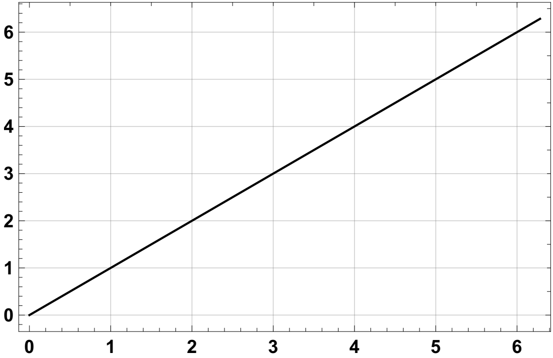
The condition numbers of certain functions

$$f(x) = x^2, \qquad \kappa(x) = 2$$



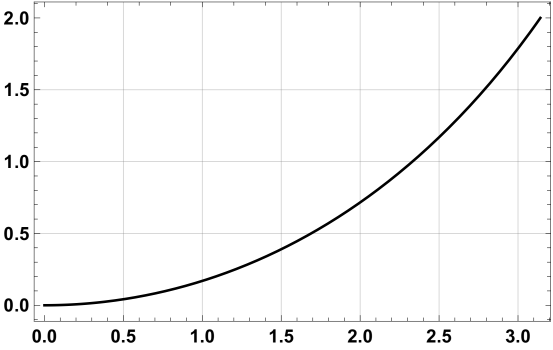
The condition numbers of certain functions

$$f(x) = \exp(x), \qquad \kappa(x) = |x|$$



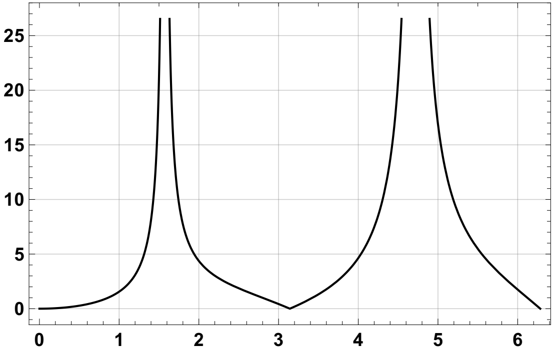
The condition numbers of certain functions

$$f(x) = \frac{1 - \cos(x)}{x^2}, \qquad \kappa(x) = x \cot\left(\frac{x}{2}\right) - 2$$



The condition numbers of certain functions

$$f(x) = \cos(x), \qquad \kappa(x) = |x\tan(x)|$$



The condition number of $f(x)$ when $f(x) = 0$

The condition number of evaluation of $\cos(x)$ at $x = \pi/2$ is ∞.

Indeed, the condition number $\kappa(x)$ of evaluation of any value f is infinite at any point $x \neq 0$ for which $f(x) = 0$ and $f'(x) \neq 0$ since

$$\kappa(x) = \left| x \frac{f'(x)}{f(x)} \right|.$$

Is this some artifact of our definition, or do we lose relative precision in practice when we evaluate $\cos(x)$ near $\pi/2$?

The condition number of $f(x)$ when $f(x) = 0$

Relative accuracy is lost when evaluating $\cos(x)$ near $x = \pi/2$ (although, high absolute accuracy can be obtained).

$$f(x) = \cos(x), \qquad \kappa(x) = |x \tan(x)|$$

x	$\kappa(x)$	computed value of $\cos(x)$	relative error
$\frac{\pi}{2} + 10^{-7}$	1.57×10^{7}	$-9.99999999971542 \times 10^{-08}$	2.85×10^{-11}
$\frac{\pi}{2} + 10^{-9}$	1.57×10^{9}	$-1.00000002150803 \times 10^{-09}$	2.15×10^{-08}
$\frac{\pi}{2} + 10^{-11}$	1.57×10^{11}	$-9.99993959506375 \times 10^{-12}$	6.04×10^{-06}
$\frac{\pi}{2} + 10^{-16}$	1.57×10^{16}	$6.12323399573677 \times 10^{-17}$	1.61×10^{00}

The condition number of $f(x)$ when $f(x) = 0$

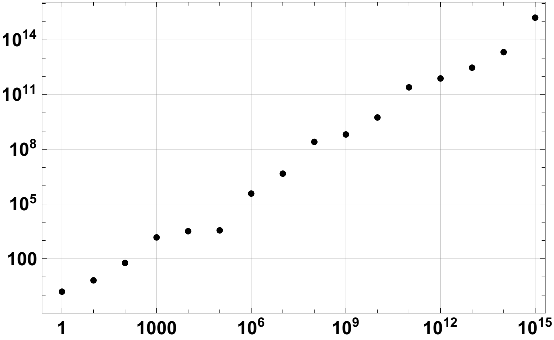
We do not necessarily lose relative accuracy if $f'(x) = 0$ or if $x = 0$. For instance, the evaluation of $\sin(x)$ near $x = 0$ is not problematic even though $\sin(0) = 0$.

$$f(x) = \sin(x), \qquad \kappa(x) = |x \cot(x)|$$

x	$\kappa(x)$	computed value of $\cos(x)$	relative error
10^{-7}	0.999999999999997	$9.99999999999998 \times 10^{-08}$	1.32×10^{-16}
10^{-9}	0.999999999999997	$1.00000000000000 \times 10^{-09}$	1.67×10^{-19}
10^{-11}	0.999999999999997	$1.00000000000000 \times 10^{-11}$	1.67×10^{-23}
10^{-13}	0.999999999999997	$1.00000000000000 \times 10^{-13}$	1.67×10^{-27}

The condition numbers of certain functions

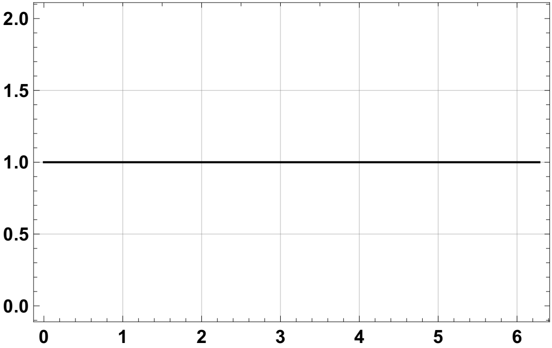
$$f(x) = \cos(x), \qquad \kappa(x) = |x \ \tan(x)|$$



The condition number of evaluation of $\cos(x)$ for various values of x.

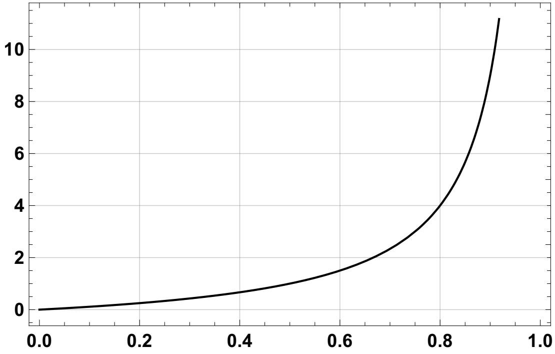
The condition numbers of certain singular functions

$$f(x) = \frac{1}{x}, \qquad \kappa(x) = 1$$



The condition numbers of certain singular functions

$$f(x) = \frac{1}{x-1}, \qquad \kappa(x) = \left|\frac{x}{x-1}\right|$$



Condition numbers and singularities

The last two examples are quite interesting — why does the condition number of

$$\frac{1}{x-1}$$

blowup as $x \to 1$, but that of

$$\frac{1}{x}$$

is constant near 0? Does this reflect what actually happens when we evaluate these functions using double precision arithmetic? Why does this happen?

Condition numbers and singularities

The last two examples are quite interesting — why does the condition number of

$$\frac{1}{x-1}$$

blowup as $x \to 1$, but that of

$$\frac{1}{x}$$

is constant near 0? Does this reflect what actually happens when we evaluate these functions using double precision arithmetic? Why does this happen?

y	δ	relative error
0	10^{-7}	$0.0000 \times 10^{+00}$
0.01	10^{-7}	5.9389×10^{-12}
0.1	10^{-7}	2.8756×10^{-11}
1	10^{-7}	5.8387×10^{-10}
10	10^{-7}	6.0775×10^{-09}
100	10^{-7}	5.9368×10^{-08}
1000	10^{-7}	3.4359×10^{-07}

The relative error in the evaluation of the quantity

$$\frac{1}{(y+\delta)-y}$$

Condition numbers and singularities

The last two examples are quite interesting — why does the condition number of

$$\frac{1}{x-1}$$

blowup as $x \to 1$, but that of

$$\frac{1}{x}$$

is constant near 0? Does this reflect what actually happens when we evaluate these functions using double precision arithmetic? Why does this happen?

The distribution of the double precision numbers!

Condition numbers and singularities

y	δ	relative error
0	10^{-7}	$0.0000 \times 10^{+00}$
1	10^{-7}	5.8387×10^{-10}
10	10^{-7}	6.0775×10^{-09}
100	10^{-7}	5.9368×10^{-08}
1000	10^{-7}	3.4359×10^{-07}

The relative error in the evaluation of the quantity

$$\frac{1}{(y+\delta)-y}$$

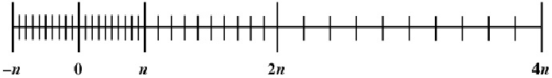
The absolute error in the computed value of $1000+\delta$ is larger than the absolute error in the computed value of $100+\delta$, which is larger than the absolute error in the computed value of $10+\delta$, and so on. This means that the cancellation error in $(y+\delta)-y$ is larger for larger y.

These absolute values are larger because the double precision numbers are less dense near 1000 than near 100, and so the distance between the closest approximation to $1000+\delta$ and its true value is greater than the distance between the closest approximation of $100+\delta$ and its true value (and so on).

Condition numbers and singularities

The double precision numbers are distributed logarithmically. That means there are the same number of double precision numbers in each interval of the form

$$\left(2^k, 2^{k+1}\right) \quad k = \ldots, -10, -9, \ldots, 0, 1, 2, 3, 4, \ldots.$$



The factor x in the condition number

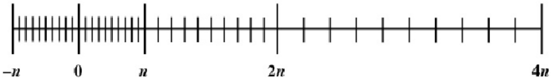
$$\left| x \, \frac{f'(x)}{f(x)} \right|$$

reflects this fact.

Condition numbers and singularities

The double precision numbers are distributed logarithmically. That means there are the same number of double precision numbers in each interval of the form

$$\left(2^k, 2^{k+1}\right) \quad k = \ldots, -10, -9, \ldots, 0, 1, 2, 3, 4, \ldots.$$



Important conclusion

Whenever possible, put singularities at 0.

Summary

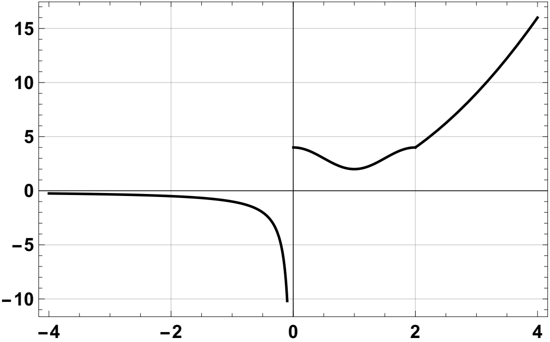
- The condition number of evaluation of f at x is
$$\kappa_f(x) = \left| x \frac{f'(x)}{f(x)} \right|.$$
- If $\kappa_f(x) = 10^k$, then we expect to lose about k digits of relative precision when evaluating f near x.
- The condition number of evaluation of rapidly oscillating functions is large for intrinsic reasons.
- The condition number of evaluation of a singular function can usually be made small by placing the singularity near 0. This exploits the logarithmic distribution of the double precision numbers.
- More generally, there is more "breathing room" near 0 because the double precision numbers are more dense there. All delicate operations should be conducted near 0 when possible.

MAT128A: Numerical Analysis
Lecture Four: Introduction to Fourier Series

October 3, 2018

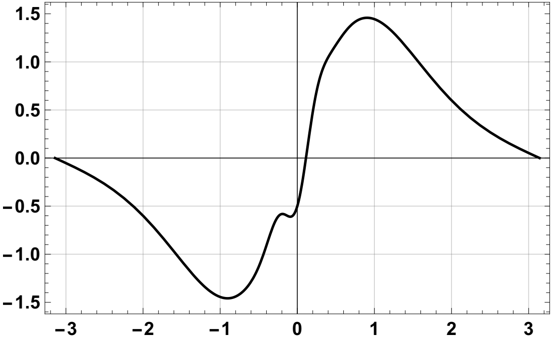
Outline of topics for the next few weeks

We are working toward a numerically viable method for the representation of piecewise smooth functions given on intervals.



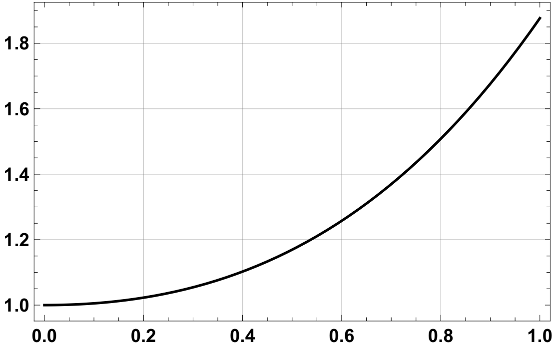
Outline of topics for the next few weeks

We will begin by considering methods for representing smooth, periodic functions on the interval $[-\pi, \pi]$.



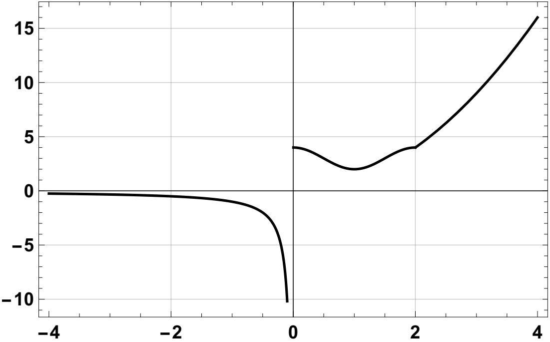
Outline of topics for the next few weeks

These methods will immediately yield an approach to representing nonperiodic, smooth functions on intervals.



Outline of topics for the next few weeks

We will then extend those methods for nonperiodic functions to handle the piecewise smooth case.



Fourier Series

In the 1820s, Joseph Fourier claimed that any function given on the interval $(-\pi, \pi)$ could be expanded in a series of the form

$$f(t) = \sum_{n=-\infty}^{\infty} a_n \exp(int).$$

He was studying partial differential equations and certain ordinary differential equations which arise from them. His primary motivation for decomposing f in this form was that

$$f'(t) = \sum_{n=-\infty}^{\infty} ina_n \exp(int),$$

assuming that the above series expansion of f, which is called a Fourier series, can be differentiated term-by-term.

Fourier's claim was met with much skepticism, some of it warranted.

Fourier Series

It is not actually the case that all functions f admit series expansions of the form

$$f(t) = \sum_{n=-\infty}^{\infty} a_n \exp(int) \tag{1}$$

which converge to f at every point in $(-\pi, \pi)$. Indeed, there are continuous functions for which this isn't true.

However, very general classes of functions can be represented by series expansions of the form (1) if one is willing to be flexible about what is meant by "converge."

Moreover, under fairly mild conditions on f, the series (1) converges to f on the interval $[-\pi, \pi]$. We will prove a result of this type shortly and focus on functions which meet these conditions.

The orthogonality of the exponential functions

Before we prove a convergence result, we will consider how we might go about computing the coefficients in an expansion of the form

$$f(t) = \sum_{n=-\infty}^{\infty} a_n \exp(int).$$

The underlying observation is that the exponential functions are orthogonal with respect to a certain inner product.

Orthonormal bases in $\mathbb{R}^n$

You should be familiar with orthonormal bases of vectors for $\mathbb{R}^n$ from your linear algebra class.

The inner product of the vectors

$$v = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} \text{ and } w = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix}$$

is

$$(v, w) = v_1 w_1 + v_2 w_2 + \cdots + v_n w_n.$$

Orthonormal bases in $\mathbb{R}^n$

We say that a set of vector $\{v_1, v_2, \ldots, v_n\}$ is an orthonormal basis for $\mathbb{R}^n$ provided:

$$(v_i, v_j) = \begin{cases} 1 & \text{if } \ i = j \\ 0 & \text{otherwise.} \end{cases}$$

If $v \in \mathbb{R}^n$, then, since $\{v_1, \ldots, v_n\}$ is a basis, there are coefficients $a_1, \ldots, a_n$ such that

$$v = \sum_{j=1}^{n} a_j v_j.$$

If we take the inner product of v with v_i we get:

$$\begin{aligned} (v, v_i) &= \left(\sum_{j=1}^{n} a_j v_j, v_i \right) \\ &= \sum_{j=1}^{n} a_j \left(v_j, v_i \right) = a_i. \end{aligned}$$

Orthonormal bases in $\mathbb{R}^n$

Expansions in orthonormal bases of vectors in $\mathbb{R}^n$

If $\{v_1, v_2, \ldots, v_n\}$ is an orthonormal basis in $\mathbb{R}^m$ and v is an arbitrary vector in $\mathbb{R}^n$, then

$$v = \sum_{i=1}^{n} a_i v_i,$$

where

$$a_i = (v, v_i)$$

for each $i = 1, 2, \ldots, n$.

Orthogonality of the exponential functions

Something similar is true for the exponential functions. If n and m are integers, then

$$\int_{-\pi}^{\pi} \exp(int) \exp(imt)\ dt = \begin{cases} 2\pi & \text{if } \ n = -m \\ 0 & \text{if } \ n \neq -m. \end{cases}$$

This is because we have

$$\begin{aligned} \int_{-\pi}^{\pi} \exp(int) \exp(imt)\ dt &= \int_{-\pi}^{\pi} \exp(i(n+m)t)\ dt \\ &= \frac{1}{(n+m)} \int_{-(n+m)\pi}^{(n+m)\pi} \exp(iu)\ du \\ &= \frac{-i}{(n+m)} \exp(iu)\Big|_{u=-\pi}^{u=\pi} = 0 \end{aligned}$$

when $n \neq -m$, and

$$\int_{-\pi}^{\pi} \exp(int) \exp(-int)\ dt = \int_{-\pi}^{\pi} 1\ dt = 2\pi.$$

Orthogonality of the exponential functions

Something similar is true for the exponential functions. If n and m are integers, then

$$\int_{-\pi}^{\pi} \exp(int)\exp(imt)\ dt = \begin{cases} 2\pi & \text{if } \ n = -m \\ 0 & \text{if } \ n \neq -m. \end{cases}$$

Theorem (Computation of Fourier Coefficients)

If f is integrable,

$$f(t) = \sum_{n=-\infty}^{\infty} a_n \exp(int),$$

and the series can be integrated term-by-term, then

$$a_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(t)\exp(-int)\ dt.$$

Formal definition of the Fourier Series

Suppose that f is a continuous function, and that, for each integer n, a_n is defined via

$$a_n = \int_{-\pi}^{\pi} f(t) \exp(-int)\ dt.$$

We note that the assumption that f is continuous is sufficient to ensure that these integrals exist and are finite. We call

$$\sum_{n=-\infty}^{\infty} a_n \exp(int)$$

the **Fourier series** for f, and a_n the n^{th} **Fourier coefficient** of f.

Review of Modes of convergence

The N^{th} partial sum for the series

$$\sum_{n=-\infty}^{\infty} a_n \exp(int) \tag{2}$$

is

$$S_N[f](t) = \sum_{n=-N}^{N} a_n \exp(int),$$

and we say that (2) converges to f at the point t provided that for all $\epsilon > 0$, there exists M such that

$$|S_N[f](t) - f(t)| < \epsilon$$

for all $N > M$.

Review of Modes of convergence

We say that the Fourier series

$$\sum_{n=-\infty}^{\infty} a_n \exp(int)$$

converges uniformly on $[-\pi, \pi]$ provided for each $\epsilon > 0$, there exists M such that

$$|S_N[f](t) - f(t)| < \epsilon \tag{3}$$

for all $N > M$ and all $t \in [-\pi, \pi]$.

This differs from the notion of pointwise convergence in that (3) does not depend on t. That is, we must be able to select the same N for all values of t.

Review of Modes of convergence

We say that the Fourier series

$$\sum_{n=-\infty}^{\infty} a_n \exp(int)$$

converges absolutely if

$$\sum_{n=-\infty}^{\infty} |a_n| < \infty.$$

We note that

$$\sum_{n=-\infty}^{\infty} |a_n \exp(int)| = \sum_{n=-\infty}^{\infty} |a_n|$$

since $|\exp(int)| = 1$.

Absolute convergence implies uniform convergence.

Uniform convergence of Fourier series

In the next lecture, we will work on proving the following theorem on the convergence of Fourier series:

Theorem

Suppose that $f : [-\pi, \pi] \to \mathbb{C}$ is continuously differentiable and periodic, and that, for each integer n, a_n is defined via the formula

$$a_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(t) \exp(-int)\ dt.$$

Then the series

$$\sum_{n=-\infty}^{\infty} a_n \exp(int)$$

converges uniformly and absolutely to $f(t)$ on $[-\pi, \pi]$.

MAT128A: Numerical Analysis
Lecture Five: Pointwise Convergence of Fourier Series

October 8, 2018

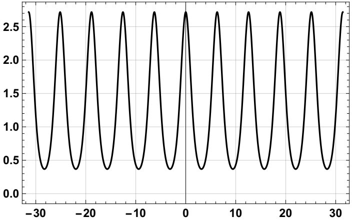
Continuously differentiable periodic functions

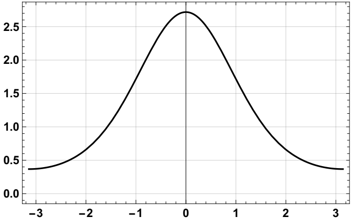
We say that a function $f : \mathbb{R} \to \mathbb{C}$ is 2π-periodic if

$$f(x + 2\pi) = f(x) \quad \text{for all} \quad x \in \mathbb{R}.$$

If this is the case, then the values of f on the interval $[-\pi, \pi)$ determine its values on all of $\mathbb{R}$. So we might as well identify 2π-periodic functions given on $\mathbb{R}$ with the set of functions given on $[-\pi, \pi) \to \mathbb{C}$.

It is actually more convenient to consider functions defined on the closed interval $[-\pi, \pi]$ instead of functions defined on $[-\pi, \pi)$, and so we will identify the 2π periodic functions with the set of all function $f : [-\pi, \pi] \to \mathbb{C}$ such that $f(-\pi) = f(\pi)$.





Continuously differentiable periodic functions

We say that the 2π-periodic function $f : [-\pi, \pi] \to \mathbb{C}$ is **continuous** provided

$$\lim_{h \to 0} f(x+h) = f(x)$$

for all $x \in (-\pi, \pi)$,

$$\lim_{h \to 0^+} f(-\pi + h) = f(-\pi),$$

and

$$\lim_{h \to 0^-} f(\pi + h) = f(\pi).$$

We say that the 2π-periodic function $f : [-\pi, \pi] \to \mathbb{C}$ is **continuously differentiable** if

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

exists for all $x \in (-\pi, \pi)$ and it extends to a continuous, periodic function on the interval $[-\pi, \pi]$.

Pointwise convergence for continuously differentiable functions

Theorem (Pointwise convergence for continuously differentiable functions)

Suppose that $f : [-\pi, \pi] \to \mathbb{C}$ is continuously differentiable and 2π-periodic, and that, for each integer n, a_n is defined via the formula

$$a_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(t) \exp(-int) \, dt.$$

Then for each $t \in [-\pi, \pi]$, the series

$$\sum_{n=-\infty}^{\infty} a_n \exp(int)$$

converges pointwise to $f(t)$.

Lemma (Bessel's Inequality)

If $f : [-\pi, \pi] \to \mathbb{C}$ *is a continuous* 2π*-periodic function and, for each integer* n*,* a_n *is defined by*

$$a_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(t) \exp(-int) \, dt,$$

then

$$\sum_{n=-\infty}^{\infty} |a_n|^2 \leq \frac{1}{2\pi} \int_{-\pi}^{\pi} |f(t)|^2 \, dt.$$

In fact, we will later see that

$$\sum_{n=-\infty}^{\infty} |a_n|^2 = \frac{1}{2\pi} \int_{-\pi}^{\pi} |f(t)|^2 \, dt.$$

Proof: Since $|z|^2 = z\overline{z}$,

$$\left|f(t) - \sum_{n=-N}^{N} a_n \exp(int)\right|^2 = \left(f(t) - \sum_{n=-N}^{N} a_n \exp(int)\right)\overline{\left(f(t) - \sum_{n=-N}^{N} a_n \exp(int)\right)}$$

$$= \left(f(t) - \sum_{n=-N}^{N} a_n \exp(int)\right)\left(\overline{f(t)} - \sum_{n=-N}^{N} \overline{a_n} \exp(-int)\right)$$

$$= |f(t)|^2 - \sum_{n=-N}^{N} a_n \overline{f(t)} \exp(int) - \sum_{n=-N}^{N} \overline{a_n} f(t) \exp(-int)$$

$$+ \sum_{n,m=-N}^{N} a_n \overline{a_m} \exp(i(n-m)t).$$

Now we divide both sides by 2π and integrate from $-\pi$ to π and use the facts that

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} f(t)\exp(-int)\ dt = a_n \quad \text{and} \quad \frac{1}{2\pi}\int_{-\pi}^{\pi} \exp(i(n-m)t)\ dt = \begin{cases} 0 & \text{if } n \neq m \\ 1 & \text{if } n = m. \end{cases}$$

By doing this we obtain:

$$\frac{1}{2\pi}\int_{-\pi}^{\pi}\left|f(t)-\sum_{n=-N}^{N}a_n\exp(int)\right|^2\,dt=\frac{1}{2\pi}\int_{-\pi}^{\pi}|f(t)|^2\;dt-\sum_{n=-N}^{N}a_n\overline{a_n}-\sum_{n=-N}^{N}\overline{a_n}a_n+\sum_{n=-N}^{N}a_n\overline{a_n}$$

$$=\frac{1}{2\pi}\int_{-\pi}^{\pi}|f(t)|^2\;dt-\sum_{n=-N}^{N}|a_n|^2\,.$$

The quantity on the left-hand side of this inequality is surely greater than or equal to 0, so

$$0\leq\frac{1}{2\pi}\int_{-\pi}^{\pi}|f(t)|^2\;dt-\sum_{n=-N}^{N}|a_n|^2\,.$$

This immediately implies

$$\sum_{n=-N}^{N}|a_n|^2\leq\frac{1}{2\pi}\int_{-\pi}^{\pi}|f(t)|^2\;dt,$$

and the conclusion of the lemma follows by taking a limit as $N\to\infty$. ■

Corollary

If $f : [-\pi, \pi] \to \mathbb{C}$ is a continuous 2π-periodic function and $\{a_n\}$ is the sequence of Fourier coefficients of f, then

$$\lim_{n \to \pm\infty} |a_n| = 0.$$

Proof: Since

$$\sum_{n=-\infty}^{\infty} |a_n|^2$$

converges, we must have

$$\lim_{n \to \pm\infty} |a_n|^2 = 0.$$

The conclusion of the theorem follows immediately. ■

The Dirichlet kernel

The n^{th} partial sum for the Fourier series of f is

$$S_N[f](t) = \sum_{n=-N}^{N} a_n \exp(int) \tag{1}$$

where

$$a_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(s) \exp(-ins)\ ds. \tag{2}$$

By inserting (2) into (1), we obtain

$$\begin{aligned} S_N[f](t) &= \sum_{n=-N}^{N} \frac{1}{2\pi} \left(\int_{-\pi}^{\pi} f(s) \exp(-ins)\ ds \right) \exp(int) \\ &= \frac{1}{2\pi} \int_{-\pi}^{\pi} f(s) \sum_{n=-N}^{N} \exp(in(t-s))\ ds \\ &= \frac{1}{2\pi} \int_{-\pi}^{\pi} f(s) \sum_{n=-N}^{N} \exp(in(s-t))\ ds \\ &= \int_{-\pi}^{\pi} f(t+s) \left(\frac{1}{2\pi} \sum_{n=-N}^{N} \exp(ins) \right) ds \end{aligned}$$

The Dirichlet kernel

Now we let

$$D_N(t) = \frac{1}{2\pi} \sum_{n=-N}^{N} \exp(int)$$

so that

$$\begin{aligned} S_N[f](t) &= \int_{-\pi}^{\pi} f(t+s) \left(\frac{1}{2\pi} \sum_{n=-N}^{N} \exp(ins) \right) ds \\ &= \int_{-\pi}^{\pi} f(t+s)\, D_N(s)\, ds. \end{aligned}$$

We call $D_N(s)$ the Dirichlet kernel.

Lemma (Properties of the Dirichlet kernel)

It is the case that

$$D_N(t) = \frac{1}{2\pi} \frac{\exp(i(N+1)t) - \exp(-iNt)}{\exp(it) - 1} \quad \textit{for all} \quad t \neq 0,$$

$$D_N(0) = \frac{2N+1}{2\pi},$$

and

$$\int_{-\pi}^{\pi} D_N(t)\ dt = 1.$$

Proof: We observe that

$$\begin{aligned} D_N(t) &= \frac{1}{2\pi} \exp(-iNt) \left(1 + \exp(it) + \exp(2it) + \cdots + \exp(2Nit)\right) \\ &= \frac{1}{2\pi} \exp(-iNt) \sum_{n=0}^{2N} (\exp(it))^n . \end{aligned}$$

Now we recall that for all $r \neq 0$,

$$\sum_{n=0}^{N} r^n = \frac{1 - r^{N+1}}{1 - r}.$$

With this formula, we obtain

$$D_N(t) = \frac{1}{2\pi} \exp(-iNt) \frac{1 - \exp(i(2N+1)t)}{1 - \exp(it)},$$

from which the first conclusion of the lemma follows easily.

The second conclusion follows easily from the definition

$$D_N(t) = \frac{1}{2\pi} \sum_{n=-N}^{N} \exp(int),$$

and the third follows by combining the definition with the observation that

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} \exp(int)\ dt = \begin{cases} 1 & \text{if } n = 0 \\ 0 & \text{otherwise.} \end{cases}$$

■

It might seem that $D_N(t)$ is discontinuous at 0, but you can verify using l'Hôpital's rule that

$$\lim_{t \to 0} \frac{1}{2\pi} \frac{\exp(i(N+1)t) - \exp(-iNt)}{\exp(it) - 1} = \frac{2N+1}{2\pi},$$

which shows that $D_N(t)$ is continuous at 0.

Proof of the theorem:

We observe that

$$
\begin{aligned}
S_N\left[f\right](t) - f(t) &= \int_{-\pi}^{\pi} f(t+s) D_N(s)\ ds - \int_{-\pi}^{\pi} f(t) D_N(s)\ ds \\
&= \int_{-\pi}^{\pi} \left(f(t+s) - f(t)\right) D_N(s)\ ds \\
&= \frac{1}{2\pi} \int_{-\pi}^{\pi} \left(f(t+s) - f(t)\right) \frac{\exp(i(N+1)s) - \exp(-iNs)}{\exp(is) - 1}\ ds \\
&= \frac{1}{2\pi} \int_{-\pi}^{\pi} \frac{f(t+s) - f(t)}{\exp(is) - 1} \quad \left(\exp(i(N+1)s) - \exp(-iNs)\right)\ ds.
\end{aligned}
$$

We define

$$
g(s) = \frac{f(t+s) - f(t)}{\exp(is) - 1},
$$

so that

$$
S_N\left[f\right](t) - f(t) = \frac{1}{2\pi} \int_{-\pi}^{\pi} g(s)\ \left(\exp(i(N+1)s) - \exp(-iNs)\right)\ ds. \tag{3}
$$

Since f is differentiable, we can apply l'Hôptial's rule to see that the function

$$g(s) = \frac{f(t+s) - f(t)}{\exp(is) - 1}$$

is continuous at 0. Indeed,

$$\lim_{s \to 0} \frac{f(t+s) - f(t)}{\exp(is) - 1} = \lim_{s \to 0} \frac{f'(t+s)}{i \exp(is)} = \frac{f'(t)}{i}$$

In particular, if we let b_n the the n^{th} Fourier coefficient of g — that is,

$$b_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} g(t) \exp(-int)\ dt$$

— then $|b_n| \to 0$ as $n \to \pm\infty$.

But (3) can be rewritten as

$$\begin{aligned} S_N[f](t) - f(t) &= \frac{1}{2\pi}\int_{-\pi}^{\pi} g(s)\ \left(\exp(i(N+1)s) - \exp(-iNs)\right)\ ds \\ &= b_{-N-1} - b_N. \end{aligned}$$

It follows that

$$|S_N[f](t) - f(t)| \leq |b_{-N-1}| + |b_N| \to 0 \ \text{ as } \ N \to \infty,$$

from which we conclude that

$$\sum_{n=-\infty}^{\infty} a_n \exp(int)$$

converges to $f(t)$. ■.

MAT128A: Numerical Analysis
Lecture Six: Uniform Convergence of Fourier Series and the Decay of Fourier Coefficients

October 8, 2018

Magnitude of Fourier Coefficients

We will very briefly set aside the technical issue of the convergence of Fourier series and discuss the rate of decay of the Fourier coefficients instead.

It will turn out that our investigations into the rate of decay of the Fourier coefficients will furnish a proof of the uniform convergence of Fourier series for continuously differentiable functions, but in the first instance our interest in estimating the rate of decay of the Fourier coefficients stems from our desire to bound(or at least approximate) the error in a truncated Fourier series:

Truncated Fourier Series

Assuming the Fourier series of f converges to f absolutely,

$$\left| f(t) - \sum_{|n| \leq N} a_n \exp(int) \right| = \left| \sum_{|n| > N} a_n \exp(int) \right| \leq \sum_{|n| > N} |a_n|$$

Magnitude of Fourier Coefficients

Here is our central observation: the Fourier coefficients of smooth functions decay rapidly. Indeed, the smoother then function, the faster the rate of decay.

We can see this using integration by parts. Suppose that f is continuously differentiable 2π-periodic function. Then:

$$\begin{aligned}\frac{1}{2\pi}\int_{-\pi}^{\pi} f(t)\exp(-int)\ dt &= -\frac{1}{2\pi}f(t)\frac{\exp(-int)}{in}\bigg|_{-\pi}^{\pi} + \frac{1}{2\pi}\int_{-\pi}^{\pi} f'(t)\frac{\exp(-int)}{in}\ dt\\ &= \frac{-i}{2\pi n}\int_{-\pi}^{\pi} f'(t)\exp(-int)\ dt.\end{aligned}$$

Since f' is continuous, we know from the Riemann-Lebesgue Lemma that

$$\lim_{n\to\infty}\left|\int_{-\pi}^{\pi} f'(t)\exp(-int)\ dt\right| = 0.$$

We say that a 2π-periodic function f is in $C^1(\mathbb{T})$ if it is continuously differentiable.

The Fourier Coefficients of C^1 Functions

If f is $C^1(\mathbb{T})$ and $\{a_n\}$ are the Fourier coefficients of f, then

$$\lim_{n \to \pm\infty} |na_n| = 0.$$

In other words, $|a_n|$ decays faster than the sequence $\frac{1}{n}$.

The "big O" notation $f(n) = \mathcal{O}(g(n))$ means

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = C < \infty,$$

while the "little o" notation $f(n) = o(g(n))$ means

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$$

The Fourier Coefficients of C^1 Functions

If f is $C^1(\mathbb{T})$ and $\{a_n\}$ are the Fourier coefficients of f, then

$$|a_n| = o\left(\frac{1}{n}\right).$$

Magnitude of Fourier Coefficients

If f is twice continuously differentiable, then

$$\begin{aligned}
\frac{1}{2\pi}\int_{-\pi}^{\pi} f(t)\exp(-int)\ dt &= -\frac{1}{2\pi} f(t)\frac{\exp(-int)}{in}\Big|_{-\pi}^{\pi} + \frac{1}{2\pi}\int_{-\pi}^{\pi} f'(t)\frac{\exp(-int)}{in}\ dt \\
&= \frac{-i}{2\pi n}\int_{-\pi}^{\pi} f'(t)\exp(-int)\ dt \\
&= \frac{-i}{2\pi n} f'(t)\frac{\exp(-int)}{-in}\Big|_{-\pi}^{\pi} + \frac{1}{2\pi n^2}\int_{-\pi}^{\pi} f''(t)\exp(-int)\ dt \\
&= \frac{1}{2\pi n^2}\int_{-\pi}^{\pi} f''(t)\exp(-int)\ dt.
\end{aligned}$$

Again, we know that

$$\left|\int_{-\pi}^{\pi} f''(t)\exp(-int)\ dt\right| \to 0 \text{ as } \ n\to\infty.$$

Magnitude of Fourier Coefficients

We will say that a 2π-periodic function is $C^k(\mathbb{T})$ if it has k derivatives and the k^{th} derivative is continuous.

Decay of Fourier Coefficients of C^k Functions

If f is a $C^k(\mathbb{T})$ function and $\{a_n\}$ is the sequence of Fourier coefficients of f, then

$$\lim_{n\to\pm\infty}\left|n^k a_n\right| = 0.$$

In other words,

$$|a_n| = o\left(\frac{1}{n^k}\right).$$

Magnitude of Fourier Coefficients

The preceding analysis can be extended to functions which are in C^{k-1} and have piecewise smooth k^{th} order derivatives. We will not prove it, but the Fourier coefficients of such a function behave as

$$|a_n| = o\left(\frac{1}{n^k}\right)$$

Magnitude of Fourier Coefficients

We will say that a 2π-periodic function is $C^{\infty}(\mathbb{T})$ if it has derivatives of all order.

Decay of Fourier Coefficients of C^{∞} Functions

If f is a $C^{\infty}(\mathbb{T})$ function and $\{a_n\}$ is the sequence of Fourier coefficients of f, then

$$\lim_{n\to\infty} n^k |a_n| = 0$$

for all nonnegative integers k.

Magnitude of Fourier Coefficients

We say that f is analytic at a point z_0 in the complex plane if it has a convergent Taylor series expansion there; that is, if it can be represented as

$$f(z) = \sum_{n=0}^{\infty} a_n (z - z_0)^n$$

for all z sufficiently close to z_0.

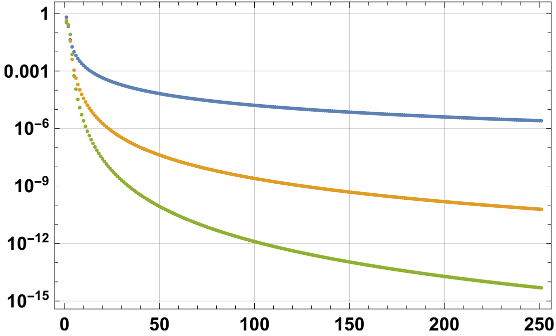
Decay of Fourier Coefficients of Functions Analytic in a Strip

If f is a 2π-periodic function which is analytic in a strip of radius $r > 0$ centered around the real axis, then

$$|a_n| = \mathcal{O}\left(\exp(-rn)\right).$$

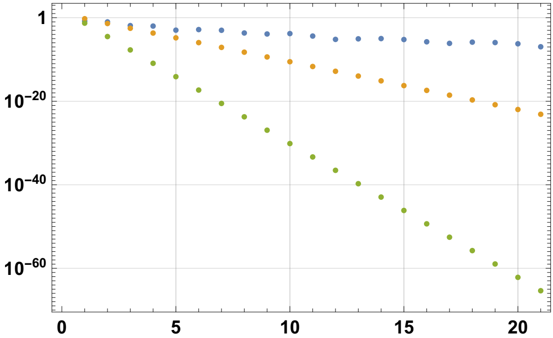
Magnitude of Fourier Coefficients

$$f(t) = |\sin(t)|, \quad g(t) = |\sin(t)|^3, \quad h(t) = |\sin(t)|^5$$



Magnitude of Fourier Coefficients

$$f(t) = \exp\left(\frac{1}{\cos(t)^2 - 1}\right) \qquad g(t) = \frac{1}{\cos(t) + 2} \qquad h(t) = \frac{1}{\cos(t) + 20}$$



Uniform Convergence for C^1 functions

Cauchy's Inequality

If $\{a_n\}_{n\in I}$ and $\{b_n\}_{n\in I}$ are sequences of complex numbers, then

$$\sum_{n\in I} \left|a_n\overline{b_n}\right| \leq \sqrt{\sum_{n\in I} |a_n|^2} \cdot \sqrt{\sum_{n\in I} |b_n|^2}.$$

Uniform Convergence for C^1 functions

If f is a continuously differentiable 2π-periodic function, then its Fourier series converges to $f(t)$ uniformly and absolutely on $[-\pi, \pi]$.

Proof: We will denote by $\{a_n\}$ the Fourier coefficients of f and by $\{b_n\}$ those of f'. It is easy to see that

$$b_n = -ina_n.$$

Cauchy's inequality implies that

$$\sum_{n\neq 0} |a_n| = \sum_{n\neq 0} \frac{n}{n} |\, a_n| = \sum_{n\neq 0} \frac{1}{n} |b_n| \leq \left(\sum_{n\neq 0} \frac{1}{n^2} \right)^{\frac{1}{2}} \cdot \left(\sum_{n\neq 0} |b_n|^2 \right)^{\frac{1}{2}} .$$

Now

$$\sum_{n\neq 0} |b_n|^2 < \infty$$

by Bessel's inequality, and

$$\sum_{n\neq 0} \frac{1}{n^2} = \frac{\pi^2}{3} < \infty.$$

■

MAT128A: Numerical Analysis
Lecture Seven: The Trapezoidal Rule

October 10, 2018

The Trapezoidal Rule

We call the quadrature rule

$$\int_a^b f(x)\ dx \approx \frac{b-a}{2n}\left(f(x_0) + 2f(x_1) + 2f(x_2) + \ldots + 2f(x_{n-1}) + f(x_n)\right),$$

where the nodes $a = x_0 < x_1 < \cdots < x_n = b$ are given by

$$x_j = a + \frac{b-a}{n}j,$$

the $(n+1)$-point trapezoidal rule on the interval $[a, b]$ (or just the $(n+1)$-point trapezoidal rule when it is clear what interval we are working on).

The Trapezoidal Rule

The trapezoidal rule is not a very good way to integrate most functions. The following table gives the relative error in

$$\int_0^1 \exp(x)\ dx$$

for various values of n.

n	relative error
10	$0.001431662930269 \times 10^{+0}$
100	$0.000014318991372 \times 10^{+0}$
1000	$1.431901499850846 \times 10^{-7}$
10000	$1.431901523477221 \times 10^{-9}$
100000	$1.431901523713485 \times 10^{-11}$

Later on, we will show that the error in the trapezoidal rule is $\mathcal{O}\left(\frac{1}{n^2}\right)$ when it is used to integrate generate smooth functions.

The Trapezoidal Rule

But the trapezoidal rule is a spectacularly good way to integrate smooth, periodic functions. The following table gives the relative errors in the integral

$$\int_{-\pi}^{\pi} \exp(\cos(t))\, dt$$

which arise when the trapezoidal rule of various lengths are used to approximate it.

n	relative error
4	$0.03439691880919236 \times 10^{+00}$
5	$0.00341130316644266 \times 10^{+00}$
6	$0.00028260086127189 \times 10^{+00}$
7	$0.00002009636897756 \times 10^{+00}$
8	$1.25168893154474934 \times 10^{-6}$
9	$3.45945653506638244 \times 10^{-9}$
12	$6.52918602772248367 \times 10^{-12}$
15	$2.97881172300385740 \times 10^{-16}$

The errors here are on the order of $\exp(-rn)$ for some constant $r > 0$.

What's going on? Why is the trapezoidal rule so effective for smooth, periodic functions?

The Trapezoidal Rule

We say that a quadrature rule

$$\int_a^b f(x)\ dx \approx \sum_{j=1}^{n} f(x_j) w_j$$

is **exact** for the functions $f_1, f_2, \ldots, f_m$ if

$$\int_a^b f_i(x)\ dx = \sum_{j=1}^{n} f_i(x_j) w_j \quad \text{for all} \quad i = 1, 2, \ldots, m.$$

That is, a quadrature rule is exact for a collection of functions if it correctly evlautes their integrals.

The Trapezoidal Rule

The $(n+1)$-point trapezoidal rule on the intervals $[-\pi, \pi]$ is

$$\int_{-\pi}^{\pi} f(t)\, dt \approx \frac{\pi}{n}\left(f(x_0) + 2f(x_1) + \ldots + 2f(x_{n-1}) + f(x_n)\right) \tag{1}$$

where

$$x_j = -\pi + \frac{2\pi}{n} j.$$

If f is 2π-periodic, then $f(x_0) = f(x_n)$ and (1) can be rewritten as

$$\begin{aligned} \int_{-\pi}^{\pi} f(t)\, dt &\approx \frac{\pi}{n}\left(2f(x_0) + 2f(x_1) + \ldots + 2f(x_{n-1})\right) \\ &= \frac{2\pi}{n}\left(f(x_0) + f(x_1) + \ldots + f(x_{n-1})\right) \\ &= \frac{2\pi}{n} \sum_{j=0}^{n-1} f\left(-\pi + \frac{2\pi}{n} j\right) \end{aligned}$$

The n-point Periodic Trapezoidal Rule

We will refer to the quadrature rule

$$\int_{-\pi}^{\pi} f(t)\, dt \approx \frac{2\pi}{n} \sum_{j=0}^{n-1} f\left(-\pi + \frac{2\pi}{n} j\right), \tag{2}$$

as the n-point periodic trapezoidal rule. It is exact for the functions

$$\exp(-ikt), \quad k = -n+1, -n+2, \ldots, -1, 0, 1, 2, \ldots, n-1.$$

Indeed, when (2) is used to approximate the integral

$$\int_{-\pi}^{\pi} \exp(ikt)\, dt,$$

the result is

$$\begin{cases} (-1)^{|k|}\, 2\pi & \text{if } k = m \cdot n \text{ for some nonzero integer } m \\ 2\pi & \text{if } k = 0 \\ 0 & \text{otherwise.} \end{cases}$$

Proof: This is part of homework assignment 3. ■

The Trapezoidal Rule

Corollary

If

$$f(t) = \sum_{n=-\infty}^{\infty} a_n \exp(int).$$

then

$$\int_{-\pi}^{\pi} f(t)\; dt = 2\pi a_0,$$

whereas the approximation of the integral obtained via the m-point periodic trapezoidal rule is

$$2\pi a_0 + 2\pi \sum_{k=1}^{\infty} (-1)^{km} \left(a_{km} + a_{-km}\right).$$

In particular, the error in the trapezoidal will be small if the Fourier coefficients of f decay rapidly!

The Trapezoidal Rule

Corollary

If f is a C^k function, then error in the approximation of

$$\int_{-\pi}^{\pi} f(t)\, dt$$

obtained via the n-point periodic trapezoidal rule is $O\left(\frac{1}{n^k}\right)$.

Corollary

If f is a 2π-periodic function which is analytic in a strip containing the real line, then the error in the approximation of

$$\int_{-\pi}^{\pi} f(t)\, dt$$

obtained via the n-point periodic trapezoidal rule is $O(\exp(-rn))$ with $r > 0$ a constant.

The first of these is a *pessimistic* estimate, meaning that it is not the best possible result.

The Trapezoidal Rule and The Exponential Functions

The functions

$$\{\exp(int) : n \in \mathbb{Z}\}$$

are orthonormal with respect to the inner product

$$(f, g) = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(t)\overline{g(t)}\, dt.$$

That is,

$$(\exp(int), \exp(imt)) = \begin{cases} 1 & \text{if } n = m \\ 0 & \text{otherwise.} \end{cases}$$

The Trapezoidal Rule and The Exponential Functions

The periodic trapezoidal rule of length $(2N+1)$ integrates the exponential functions

$$\exp(-i2Nt), \ldots, \exp(-it), 1, \exp(it), \ldots, \exp(i2Nt).$$

This menas that it integrates all products of the form

$$\exp(int)\exp(imt)$$

with $-N \leq n, m \leq N$.

In other words, it accurately discretizes the restriction of inner product we just defined to the finite-dimensional space

$$\{\exp(int) : -N \leq n \leq N\}.$$

The Trapezoidal Rule and The Exponential Functions

That is to say: if $t_0, t_1, \ldots, t_{2N}, w_0, w_1, \ldots, w_{2N}$ are the nodes and weights of the $(2N+1)$-point periodic trapezoidal rule then

$$(\exp(int), \exp(imt)) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \exp(int)\exp(-imt)\ dt = \frac{1}{2\pi}\sum_{j=0}^{2N} \exp(int_j)\exp(-imt_j) w_j$$

for all $-N \le n, m \le N$.

The Trapezoidal Rule and The Exponential Functions

If we assign to each 2π-periodic function f the $\mathbb{C}^{2N+1}$ vector

$$[f] = \begin{pmatrix} f(t_0)\sqrt{\dfrac{w_0}{2\pi}} \\ f(t_1)\sqrt{\dfrac{w_1}{2\pi}} \\ \vdots \\ f(t_{2N})\sqrt{\dfrac{w_{2N}}{2\pi}} \end{pmatrix},$$

then

$$[f] \cdot [g] = \frac{1}{2\pi} \sum_{j=0}^{2N+1} f(t_j)\overline{g(t_j)} w_j \approx \frac{1}{2\pi} \int f(t)\overline{g(t)}\, dt. \tag{3}$$

If f and g are in the span of

$$E_N = \{\exp(int) : -N \leq n \leq N\},$$

then (3) is exact!! This is what we mean by the trapezoidal rule discretizes the inner product we defined on the space E_N.

MAT128A: Numerical Analysis
Lecture Eight: Numerical Computation of Fourier Coefficients

October 12, 2018

Computation of Fourier Coefficients

Let $t_0, t_1, \ldots, t_{2N}, w_0, w_1, \ldots, w_{2N}$ be the nodes and weights of the $(2N+1)$-point periodic trapezoidal rule.

Last time, we saw that the $(2N+1)$-point periodic trapezoidal quadrature rule accurately integrates all products of the form

$$\exp(int)\exp(imt) \quad \text{with} \quad -N \le n, m \le N.$$

This makes it the perfect tool to compute the Fourier coefficients of a function f which admits a finite Fourier series expansion of the form

$$f(t) = \sum_{n=-N}^{N} a_n \exp(int).$$

Computation of Fourier Coefficients

More explicitly, if

$$f(t) = \sum_{n=-N}^{N} a_n \exp(int).$$

and $-N \leq m \leq N$, then

$$a_m = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(t) \exp(-imt)\ dt = \frac{1}{2\pi} \sum_{j=0}^{2N} f(t_j) \exp(-imt) w_j.$$

The approximation of this integral obtained by the trapezoidal rule is exact because

$$\sum_{n=-N}^{N} a_n \exp(int) \exp(-imt) = \sum_{n=-N}^{N} a_n \exp(i(n-m)t),$$

which is in the span of

$$\{\exp(int) : -2N \leq n \leq 2N\}.$$

Computation of Fourier Coefficients

Using the notation

$$[f] = \begin{pmatrix} f(t_0)\sqrt{\dfrac{w_0}{2\pi}} \\ f(t_1)\sqrt{\dfrac{w_1}{2\pi}} \\ \vdots \\ f(t_{2N})\sqrt{\dfrac{w_{2N}}{2\pi}} \end{pmatrix},$$

from last time, we can rewrite the expression

$$a_m = \frac{1}{2\pi}\sum_{j=0}^{2N} f(t_j)\exp(-imt)w_j.$$

as

$$a_m = [f] \cdot [\exp(imt)].$$

Computation of Fourier Coefficients

Note the parallels in the two statements:

$$a_m = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(t) \exp(-imt)\ dt$$

and

$$a_m = [f] \cdot [\exp(imt)] .$$

The coefficient a_m is obtained as the inner product of the function f with $\exp(imt)$ or as the inner product of the vector $[f]$ with the vector $[\exp(imt)]$.

Computation of Fourier Coefficients

What happens when f has a Fourier expansion of the form

$$f(t) = \sum_{n=-\infty}^{\infty} a_n \exp(int)$$

and we use the $(2N+1)$-point periodic trapezoidal rule to approximate the integral

$$a_m = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(t) \exp(-imt)\ dt?$$

For technical reasons, we will assume the Fourier series of f converges absolutely. Moreover, we will denote by $\widetilde{a_m}$ the approximation of a_m obtained in this way and develop a bound on the error of

$$\left| f(t) - \sum_{n=-N}^{N} \widetilde{a}_n \exp(int) \right|$$

in terms of the Fourier coefficients of f.

Computation of Fourier Coefficients

We start by writing

$$\left| f(t) - \sum_{n=-N}^{N} \tilde{a}_n \exp(int) \right| \leq \left| f(t) - \sum_{n=-N}^{N} a_n \exp(int) \right| + \left| \sum_{n=-N}^{N} \tilde{a}_n \exp(int) - \sum_{n=-N}^{N} a_n \exp(int) \right|$$

and noting that it is easy to estimate the first sum in terms of the Fourier coefficients of f:

$$\left| f(t) - \sum_{n=-N}^{N} a_n \exp(int) \right| = \left| \sum_{|n|>N} a_n \exp(int) \right| \leq \sum_{|n|>N} |a_n|.$$

Computation of Fourier Coefficients

Next, we observe that

$$a_m = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(t)\exp(-imt)\ dt = \frac{1}{2\pi}\sum_{n=-\infty}^{\infty} a_n \int_{-\pi}^{\pi}\exp(i(n-m)t)\ dt$$
$$= \frac{1}{2\pi}\sum_{n=-\infty}^{\infty} a_{n+m}\int_{-\pi}^{\pi}\exp(int)\ dt$$

and note that according to a problem from Homework 3, when the $(2N+1)$-point trapezoidal rule is used to evaluate the integral of $\exp(int)$, the result is

$$\begin{cases} (-1)^{|n|}\ 2\pi & \text{if } n = m\cdot(2N+1) \text{ for some nonzero integer } m \\ 2\pi & \text{if } k = 0 \\ 0 & \text{otherwise.} \end{cases}$$

It follows that

$$\widetilde{a}_n = a_n + \sum_{j=1}^{\infty}(-1)^{(2N+1)j}\left(a_{(2N+1)j+n} + a_{-(2N+1)j+n}\right).$$

Computation of Fourier Coefficients

Using the fact that

$$\tilde{a}_n = a_n + \sum_{j=1}^{\infty} (-1)^{(2N+1)j} \left(a_{(2N+1)j+n} + a_{-(2N+1)j+n} \right),$$

we can bound the sum

$$\left| \sum_{n=-N}^{N} \tilde{a}_n \exp(int) - \sum_{n=-N}^{N} a_n \exp(int) \right| = \left| \sum_{n=-N}^{N} \exp(int) \left(\tilde{a}_n - a_n \right) \right|.$$

In particular,

$$\begin{aligned}
\left|\sum_{n=-N}^{N} (\tilde{a}_n - a_n)\exp(int)\right| &= \left|\sum_{n=-N}^{N} \exp(int) \sum_{j=1}^{\infty} (-1)^{(2N+1)j} \left(a_{(2N+1)j+n} + a_{-(2N+1)j+n}\right)\right| \\
&\leq \sum_{n=-N}^{N} \sum_{j=1}^{\infty} \left(\left|a_{(2N+1)j+n}\right| + \left|a_{-(2N+1)j+n}\right|\right) \\
&\leq \sum_{n=-N}^{N} \sum_{j=-\infty}^{\infty} \left|a_{(2N+1)j+n}\right| \\
&= \sum_{|n|>N} \left|a_n\right|.
\end{aligned}$$

Note that as j varies from $-\infty$ to ∞ and n varies from $-N$ to N, the expression

$$(2N+1)j + n$$

assumes the value of each integer greater than N in magnitude exactly once.

Computation of Fourier Coefficients

Aliasing Error

Suppose that f is a 2π periodic function which is continuously differentiable, that $\{a_n\}$ are the Fourier coefficients of f, and that $\{\tilde{a}_n\}$ are the approximations of these coefficients obtained via the $(2N+1)$-point trapezoidal rule. Then

$$\left| f(t) - \sum_{n=-N}^{N} a_n \exp(int) \right| \leq \sum_{|n|>N} |a_n|$$

while

$$\left| f(t) - \sum_{n=-N}^{N} \tilde{a}_n \exp(int) \right| \leq 2 \sum_{|n|>N} |a_n|.$$

In other words, when we approximate the Fourier coefficients of f using the trapezoidal rule of appropriate length, we get an error which is only twice as large as the error we get if we use the exact values of the coefficients.

Are Fourier Series Numerically Viable?

We saw that expansions in monomials tend to suffer from numerical difficulties.

Now that we have a method for evaluating the Fourier coefficients of a periodic function f, it is natural to ask a few questions:

Is the numerical evaluation of Fourier coefficients numerically stable?

Once the coefficients are known, is the numerical evaluation of a Fourier series numerically stable?

Are Fourier Series Numerically Viable?

The answer to both of these questions is a resounding yes, as long as we define stability in terms of the Euclidean norm.

The Euclidean norm of a vector $x = (x_1, \ldots, x_n)$ in $\mathbb{C}^n$ is

$$\|x\| = \sqrt{\sum_{j=1}^{n} |x_j|^2}$$

The norm of the matrix A is

$$\|A\| = \max_{x \neq 0} \frac{\|Ax\|}{\|x\|},$$

so that

$$\|Ax\| \leq \|A\| \|x\|$$

for all $x \in \mathbb{C}^n$.

The Condition Number of a Matrix

if A is invertible, then we define its condition number to be

$$\kappa_A = \|A\| \|A^{-1}\|$$

If x is a vector in $\mathbb{C}^n$ and $\widehat{Ax}$ is the approximation of Ax obtained with double precision arithmetic, then we expect that

$$\frac{\|Ax - \widehat{Ax}\|}{\|Ax\|} \approx \kappa_A(x)\epsilon.$$

In other words, κ_A measures the loss of relative accuracy when we apply A to x. Note though that we are measuring precision with respect to the Euclidean norm.

The Condition Number of a Matrix

We say that a matrix A is Hermitian if

$$AA^* = I,$$

where A^* is the conjugate transpose of A. That is, if

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{pmatrix}$$

then

$$A^* = \begin{pmatrix} \overline{a_{11}} & \overline{a_{21}} & \overline{a_{31}} & \cdots & \overline{a_{n1}} \\ \overline{a_{12}} & \overline{a_{22}} & \overline{a_{32}} & \cdots & \overline{a_{n2}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \overline{a_{1n}} & \overline{a_{2n}} & \overline{a_{3n}} & \cdots & \overline{a_{nn}} \end{pmatrix}.$$

The Condition Number of a Matrix

Theorem

The condition number of a Hermitian matrix A is 1.

Proof: You can verify that

$$(Ax, y) = (x, A^* y).$$

It follows that

$$\|Ax\|^2 = (Ax, Ax) = (x, A^* Ax) = (x, x) = \|x\|^2.$$

So

$$\frac{\|Ax\|}{\|x\|} = 1$$

for any $x = 1$, which establishes that $\|A\| = 1$. If A is Hermitian, then so is A^*, so $\|A^*\| = 1$. The theorem easily follows from this.

■

The Computation of Fourier Coefficient is Perfectly Stable

Now, for each $n = -N, -N+1, \ldots, N$, the n^{th} approximate Fourier coefficient of f is

$$\widetilde{a_n} = [\exp(int)] \cdot [f].$$

If we let A denote the matrix

$$\begin{pmatrix} [\exp(-iNt)] \\ [\exp(-i(N-1)t] \\ \vdots \\ [\exp(i(N)t] \end{pmatrix},$$

whose rows are the discretized exponential functions, then the approximate Fourier coefficients are given by

$$\begin{pmatrix} \widetilde{a_{-N}} \\ \widetilde{a_{-N+1}} \\ \vdots \\ \widetilde{a_0} \\ \vdots \\ \widetilde{a_N} \end{pmatrix} = A[f].$$

The Computation of Fourier Coefficients is Stable

But the rows of the matrix

$$A = \begin{pmatrix} [\exp(-iNt)] & [\exp(-i(N-1)t] & \cdots & [\exp(i(N)t] \end{pmatrix}$$

are orthonormal since

$$[\exp(int] \cdot [\exp(imt)] = \frac{1}{2\pi}\int_{-\pi}^{\pi} \exp(int)\exp(imt)\ dt$$

(note that this is because we used the $(2N+1)$-point periodic trapezoidal rule to discretize the exponential functions).

It follows that A is a Hermitian matrix. So the numerical computation of the approximate Fourier coefficients is stable, in the sense of Euclidean norms. We note that we will not incur as significant relative error when we scale by quadrature weights.

Computation of Fourier Coefficients

The computation of the approximate Fourier coefficients $\{\tilde{a}_n : n = -N, \ldots, N\}$ of the function f can be performed by applying a Hermitian matrix to the vector

$$\begin{pmatrix} f\left(-\pi + \frac{0}{2N+1}\right)\sqrt{\frac{1}{2N+1}} \\ f\left(-\pi + \frac{1}{2N+1}\right)\sqrt{\frac{1}{2N+1}} \\ f\left(-\pi + \frac{2}{2N+1}\right)\sqrt{\frac{1}{2N+1}} \\ \vdots \\ f\left(-\pi + \frac{2N}{2N+1}\right)\sqrt{\frac{1}{2N+1}} \\ f\left(-\pi + \frac{2N}{2N+1}\right)\sqrt{\frac{1}{2N+1}} \end{pmatrix}$$

Evaluation of Fourier Series at Appropriate Quadrature Nodes is Stable

Suppose that

$$x_1, \ldots, x_m, w_1 \ldots, w_m$$

are the nodes and weights of a quadrature rule which integrates the exponential functions

$$\{\exp(int) : n = -2N, \ldots, -1, 0, 1, \ldots, 2N\}.$$

Then the matrix E

$$E = \begin{pmatrix} E_{1,-N} & E_{1,-N+1} & \cdots & E_{1,N} \\ E_{2,-N} & E_{2,-N+1} & \cdots & E_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ E_{m,-N} & E_{m,-N+1} & \cdots & E_{m,N} \end{pmatrix},$$

with entries

$$E_{j,k} = \sqrt{\frac{w_k}{2N+1}} \exp\left(ikx_j\right)$$

is also Hermitian.

Evaluation of Fourier Series at Appropriate Quadrature Nodes is Stable

Moreover, when we apply E to the vector of approximate Fourier coefficients

$$\begin{pmatrix} \widetilde{a_{-N}} \\ \widetilde{a_{-N+1}} \\ \vdots \\ \widetilde{a_0} \\ \widetilde{a_1} \\ \vdots \\ \widetilde{a_N} \end{pmatrix},$$

we obtain the scaled values of the approximation

$$f(t) = \sum_{n=-N}^{N} \widetilde{a_n} \exp(int)$$

at the nodes $x_1, \ldots, x_m$; that is, the output is the vector

$$\begin{pmatrix} f(x_1)\sqrt{w_1} \\ f(x_2)\sqrt{w_2} \\ \vdots \\ f(x_m)\sqrt{w_m} \end{pmatrix}.$$

One Thing More

We know that evaluating exponential functions of large arguments numerically results in large relative errors. Usually, this isn't a problem since we typically don't use Fourier series of such large orders that the effect is significant.

However, it is useful to know that these roundoff errors can be avoided in this case, albeit via a slow procedure.

One Thing More

Suppose we wish to evaluate $\cos(nt)$ and $\sin(nt)$ for $-\pi < t < \pi$ and n a large positive integer.

A large cancellation error will result if we compute the product nt in double precision arithmetic and then evaluate cosine and sine. But in this case there is a way around this problem.

We will use the identities

$$\cos(a+b) = \cos(a)\cos(b) - \sin(a)\sin(b)$$

and

$$\sin(a+b) = \cos(a)\sin(b) + \sin(a)\cos(b)$$

One Thing More

In particular, we can compute $\cos(nt)$ and $\sin(nt)$using the values of $\cos((n-1)t)$ and $\sin((n-1)t)$ via the relations

$$\begin{aligned} \cos(nt) &= \cos((n-1)t)\cos(t) - \sin((n-1)t)\sin(t) \\ \sin(nt) &= \cos((n-1)t)\sin(t) + \sin((n-1))t)\cos(t) \end{aligned} \tag{1}$$

We first compute the values of $\cos(t)$ and $\sin(t)$ and then use those values and (1) to compute $\cos(2t)$ and $\sin(2t)$. Now we can compute $\cos(3t)$ and $\sin(3t)$ using (1). and so on until we reach the values of $\sin(nt)$ and $\cos(nt)$.

Relations of the form (1) are called recurrence relations and these recurrence relations give us a stable (albeit slow) way of computing $\cos(nt)$ and $\sin(nt)$ when n is a large integer and t is in the interval $[-\pi, \pi]$.

MAT128A: Numerical Analysis
Lecture Ten: Calculation of Chebyshev Expansions, Part I

October 17, 2018

Chebyshev expansions

Suppose that f is a continously differentiable function on the interval $[-1, 1]$. The Chebyshev expansion of f is essentially the same as the cosine expansion of

$$g(t) = f(\cos(t)),$$

which is an even periodic function on $[-\pi, \pi]$.

The cosine expansion of g is

$$g(t) = \sum_{n=0}^{\infty}{}' a_n \cos(nt)$$

where

$$a_n = \frac{2}{\pi} \int_0^{\pi} g(t) \cos(nt)\ dt$$

and the prime sign next to the summation means that the first term in the series is scaled by $\frac{1}{2}$.

Chebyshev expansions

If we make the change of variables $t = \arccos(x)$, we obtain

$$f(x) = \sum_{n=0}^{\infty}{}' a_n \cos(n \arccos(x))$$

where

$$a_n = \frac{2}{\pi} \int_{-1}^{1} f(x) \cos(n \arccos(x)) \frac{dx}{\sqrt{1-x^2}}.$$

As we saw last time, the function $\cos(n \arccos(x))$ is a polynomial of degree n. We call it the Chebyshev polynomial of degree n and denote it by T_n.

Chebyshev expansions

Using the notation T_n for Chebyshev polynomials, we see that the Chebyshev expansion of f is

$$f(x) = {\sum_{n=0}^{\infty}}' a_n T_n(x),$$

where

$$a_n = \frac{2}{\pi} \int_{-1}^{1} f(x) T_n(x) \frac{dx}{\sqrt{1-x^2}}.$$

Chebyshev expansions

Chebyshev polynomials are a collection of **orthogonal polynomials**.

Using the fact that

$$\cos(nt) = \frac{\exp(int) + \exp(int)}{2},$$

you can easily verify that

$$\frac{2}{\pi}\int_0^\pi \cos(nt)\cos(mt)\ dt = \begin{cases} 2 & \text{if} \quad n = m = 0 \\ 1 & \text{if} \quad n = m \neq 0 \\ 0 & \text{if} \quad n \neq m. \end{cases}$$

Chebyshev expansions

Introducing the new variable $x = \cos(t)$ gives us

$$\frac{2}{\pi}\int_0^{\pi} \cos(nt)\cos(mt)\ dt = \frac{2}{\pi}\int_{-1}^{1} \cos(n\arccos(x))\cos(m\arccos(x))\ \frac{dx}{\sqrt{1-x^2}}$$

$$= \frac{2}{\pi}\int_{-1}^{1} T_n(x)T_m(x)\ \frac{dx}{\sqrt{1-x^2}}.$$

In particular,

$$\frac{2}{\pi}\int_{-1}^{1} T_n(x)T_m(x)\ \frac{dx}{\sqrt{1-x^2}} = \begin{cases} 2 & \text{if } \ n=m=0 \\ 1 & \text{if } \ n=m\neq 0 \\ 0 & \text{if } \ n\neq m \end{cases}$$

and we say that the set $\{T_n\}$ of Chebyshev polynomials is orthogonal with respect to the weight function $w(x) = \dfrac{1}{\sqrt{1-x^2}}$.

Two Quadrature Rules for Calculating Cosine Expansions

Just as we did for exponential functions, we will try to construct quadrature rules which discretize this inner product. That will allow us to compute Chebyshev expansions by exploting orthonormality.

We are actually going to derive **two** different quadrature rules for computing the coefficients in cosine expansions. Each of these will give us a method for computing Chebyshev expansions.

These quadratures correspond to two different sets of points which are commonly used as discretization points for Chebyshev expansions.

Two Quadrature Rules for Calculating Cosine Expansions

The set of points

$$\left\{\cos\left(\frac{\pi j}{n-1}\right) : j = 0, 2, \ldots, n-1\right\}$$

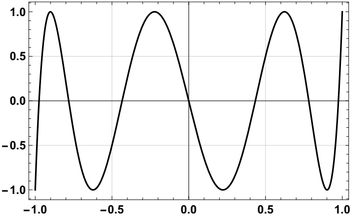
is called the *n*-**point Chebyshev extrema grid**. It is so named because the local maximums and minimums of the function $T_{n-1}(x)$ occur at these points. Note that the endpoints -1 and 1 of the interval $[-1, 1]$ are included in this set.

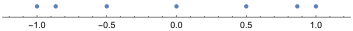
The set of points

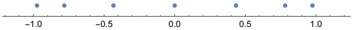
$$\left\{\cos\left(\frac{j+\frac{1}{2}}{n}\pi\right) : j = 0, 2, \ldots, n-1\right\}$$

is called the *n*-**point Chebyshev root grid**. It is so named because these the roots of the function $T_n(x)$ occur at these points. Note that the endpoints -1 and 1 of the interval $[-1, 1]$ are **not** included in this set.

Two Quadrature Rules for Calculating Cosine Expansions

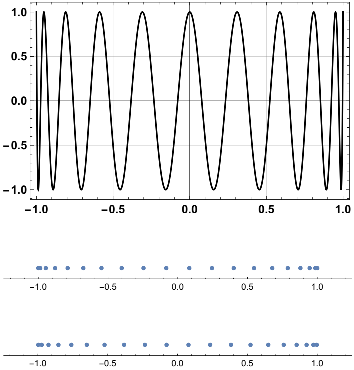






Two Quadrature Rules for Calculating Cosine Expansions

Observation: the nodes of both grids cluster close to the endpoints ± 1 of the intervals.



A Rule for the Extrema Grid

The first of our two rules is constructed using the periodic trapezoidal rule.

If g is a linear combination of the functions

$$\{\exp(int) : -2N-1 \leq n \leq 2N-1\},$$

then

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} g(t)\, dt = \frac{1}{2N}\sum_{j=0}^{2N-1} g\left(-\pi + \frac{2\pi}{2N}j\right)$$

A Rule for the Extrema Grid

If g is also even, then

$$
\begin{aligned}
\frac{1}{2\pi}\int_{-\pi}^{\pi} g(t)\,dt &= \frac{1}{2N}\sum_{j=0}^{2N-1} g\left(-\pi+\frac{2\pi}{2N}j\right) \\
&= \frac{1}{2N}\left(g\left(-\pi\right)+\sum_{j=1}^{N-1} g\left(-\pi+\frac{2\pi}{2N}j\right)\right. \\
&\left.+g(0)+\sum_{j=N+1}^{2N-1} g\left(-\pi+\frac{2\pi}{2N}j\right)\right) \\
&= \frac{1}{2N}\left(g\left(-\pi\right)+g(0)+2\sum_{j=1}^{N-1} g\left(-\pi+\frac{2\pi}{2N}(j+N)\right)\right) \\
&= \frac{1}{2N}\left(g\left(-\pi\right)+g(0)+2\sum_{j=1}^{N-1} g\left(\frac{\pi}{N}j\right)\right) \\
&= \frac{1}{N}\left(\frac{1}{2}g\left(-\pi\right)+\frac{1}{2}g(0)+\sum_{j=1}^{N-1} g\left(\frac{\pi}{N}j\right)\right).
\end{aligned}
$$

A Rule for the Extrema Grid

It follows that

$$
\begin{aligned}
\frac{2}{\pi}\int_0^{\pi} g(t)\ dt &= \frac{1}{\pi}\int_{-\pi}^{\pi} g(t)\ dt \\
&= \frac{2}{N}\left(\frac{1}{2}g\left(-\pi\right) + \frac{1}{2}g(0) + \sum_{j=1}^{N-1} g\left(\frac{\pi}{N}j\right)\right).
\end{aligned}
$$

A Rule for the Extrema Grid

If g is in the span of the functions

$$\{\cos(nt) : 0 \leq n \leq 2N - 1\},$$

then

$$\frac{2}{\pi}\int_0^{\pi} g(t)\ dt = \frac{2}{N}\left(\frac{1}{2}g(0) + \sum_{j=1}^{N-1} g\left(\frac{\pi}{N}j\right) + \frac{1}{2}g(-\pi)\right)$$
$$= \frac{2}{N}\sum_{j=0}^{N}{}'' g\left(\frac{\pi}{N}j\right).$$

Here, we are using the "double prime" next to the summation symbol to indicate that its first and last terms in the sum are scaled by $\frac{1}{2}$.

A Rule for the Extrema Grid

It is obvious that when we take the product of two exponential functions of orders between $-N$ and N, the result is an exponential function of order between $-2N$ and $2N$. Something similar is true for cosines.

Lemma

If n and m are nonnegative integers, then

$$\cos(nt)\cos(mt) = \frac{1}{2}\cos((n+m)t) + \frac{1}{2}\cos((n-m)t).$$

In particular, $\cos(nt)\cos(mt)$ is in the span of the set

$$\{1, \cos(t), \cos(2t), \ldots, \cos(2Nt)\}.$$

A Rule for the Extrema Grid

Proof: We observe that

$$\begin{aligned}
\cos(nt)\cos(mt) &= \frac{\exp(int)+\exp(-int)}{2}\,\frac{\exp(imt)+\exp(-imt)}{2} \\
&= \frac{1}{4}\left(\exp(i(n+m)t)+\exp(-i(n+m)t)\right) \\
&+ \frac{1}{4}\left(\exp(i(n-m)t)+\exp(-i(n-m)t)\right) \\
&= \frac{1}{2}\cos((n+m)t)+\frac{1}{2}\cos((n-m)t).
\end{aligned}$$

■

A Rule for the Extrema Grid

Theorem

If

$$g(t) = \sum_{n=0}^{N-1}{}' a_n \cos(nt),$$

then

$$\begin{aligned} a_m &= \frac{2}{\pi} \int_0^{\pi} g(t) \cos(mt) dt \\ &= \frac{2}{N} \sum_{j=0}^{N}{}'' g\left(\frac{\pi}{N} j\right) \cos\left(\frac{\pi}{N} mj\right) \end{aligned}$$

for all $m = 0, 1, \ldots, N-1$.

A Rule for the Extrema Grid

Proof: The integrand in is in the span of the functions

$$\{\cos(nt)\cos(mt) : 0 \leq n, m \leq N-1\}.$$

By the lemma, this coincides with the span of the set

$$\{\cos(nt) : 0 \leq n \leq 2N-2\}.$$

The conclusion of the theorem follows from our earlier observation that the quadrature rule of length $N+1$ integrates anything in the span of the set

$$\{\cos(nt) : 0 \leq n \leq 2N-1\}.$$

∎

A Rule for the Extrema Grid

The preceding theorem is a little bit unsatisfying since the quadrature rule is of length $N+1$, while the expansion of f has only N terms.

This is different from what happened when we used the trapezoidal rule to compute Fourier series. There, we found that a $(2N+1)$-point rule was sufficient to evaluate the coefficients of a $(2N+1)$-term series.

This really isn't too much of a problem, though. It is immediate that that

$$\frac{2}{N}\sum_{j=0}^{N}{}'' cos\left(\frac{\pi}{N}Nj\right)\cos\left(\frac{\pi}{N}Nj\right) = \frac{2}{N}\sum_{j=0}^{N}{}'' cos\left(\frac{\pi}{j}\right)^2 = 2.$$

This is incorrect in that the integral should be 1, but this formula will enable us to compute the N^{th} coefficient, even though our quadrature rule isn't quite right.

A Rule for the Extrema Grid

Theorem

If g is in the span of the set

$$\{\cos(nt) : n = 0, 1, \ldots, N\},$$

then the cosine expansion of g is

$$g(t) = \sum_{n=0}^{N}{}'' b_n \cos(nt),$$

where

$$b_m = \frac{2}{N} \sum_{j=0}^{N}{}'' g\left(\frac{\pi}{N} j\right) \cos\left(\frac{\pi}{N} mj\right)$$

for all $m = 0, 1, \ldots, N$.

In particular,

$$\frac{2}{N}\sum_{j=0}^{N}{}'' \cos\left(\frac{\pi}{N}nj\right)\cos\left(\frac{\pi}{N}mj\right) = \begin{cases} 2 & \text{if} \;\; n = m = 0; \\ 2 & \text{if} \;\; n = m = N; \\ 1 & \text{if} \;\; n = m \;\; \text{and} \;\; 0 < n, m < N; \\ 0 & \text{if} \;\; n \neq m \;\; \text{and} \;\; 0 < n, m < N, \end{cases}$$

whereas

$$\frac{2}{\pi}\int_0^{\pi} \cos(nt)\cos(mt)\; dt = \begin{cases} 2 & \text{if} \;\; n = m = 0; \\ 1 & \text{if} \;\; n = m = N; \\ 1 & \text{if} \;\; n = m \;\; \text{and} \;\; 0 < n, m < N; \\ 0 & \text{if} \;\; n \neq m \;\; \text{and} \;\; 0 < n, m < N \end{cases}$$

A Rule for the Extrema Grid

By introducing the new variable $x = \cos(t)$, we obtain the corresponding result for Chebyshev expansions.

Theorem

If f is a polynomial of degree N, then

$$f(x) = {\sum_{n=0}^{N}}'' b_n T_n(x),$$

where

$$b_m = \frac{2}{N} {\sum_{j=0}^{N}}'' f\left(\cos\left(\frac{\pi}{N} j\right)\right) T_m\left(\cos\left(\frac{\pi}{N} j\right)\right)$$

for all $m = 0, 1, \ldots, N$.

MAT128A: Numerical Analysis
Lecture Nine: Chebyshev Expansions

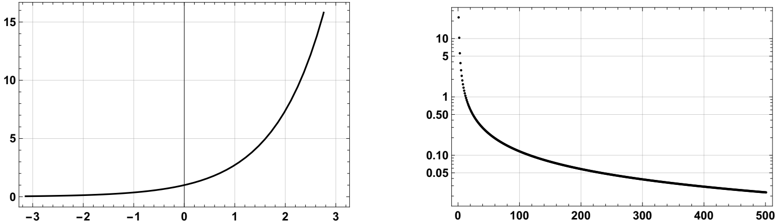
October 15, 2018

Fourier Expansions of Nonperiodic Functions Are Inefficient

Fourier series are an effective mechanism for representing smooth *periodic* functions.

They are not effective mechanism for representing nonperiodic functions.

$$f(x) = \exp(x)$$



From Nonperiodic to Periodic

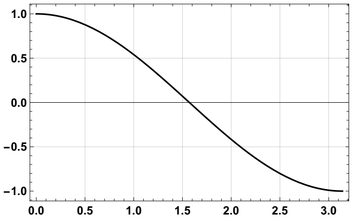
We will represent periodic functions $f : [-1,1] \to \mathbb{C}$ via a Fourier series for the periodic function

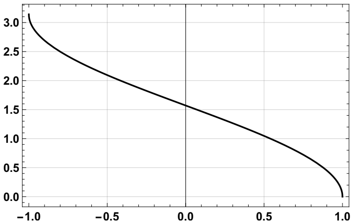
$$g(t) = f(\cos(t)).$$

Another way of thinking about this is that, given $f(x)$, we are introducing the new variable t which is related to x via

$$x = \cos(t) \quad \text{and} \quad t = \arccos(x).$$

Note that cosine maps the interval $[0, \pi]$ onto the interval $[-1, 1]$. By arccosine, we mean the inverse of this map.

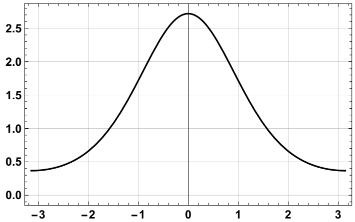


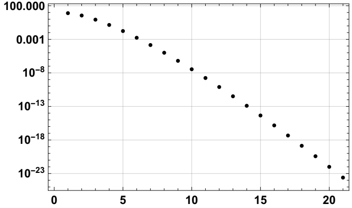


Fourier Expansions of Nonperiodic Functions Are Inefficient

To repesent a nonperiodic function f on $[-1,1]$, we will form a Fourier series of the function $f(\cos(t))$.

$$f(x) = \exp(\cos(x))$$





Cosine Expansions of Even Functions

Suppose that $f : [-\pi, \pi] \to \mathbb{C}$ is a continuously differentiable 2π-periodic function. Then

$$f(t) = \sum_{n=-\infty}^{\infty} a_n \exp(int) \tag{1}$$

where

$$a_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(t) \exp(-int)\ dt. \tag{2}$$

If we make the additional assumption that f is even — that is,

$$f(-t) = f(t) \ \text{ for all } \quad 0 < t < \pi$$

— then we can simplify (1) and (2).

Cosine Expansions of Even Functions

If f is even, then

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} f(t)\sin(nt)\ dt = 0 \quad \text{for all} \quad n \in \mathbb{Z},$$

and

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} f(t)\cos(nt)\ dt = \frac{1}{\pi}\int_{0}^{\pi} f(t)\cos(nt)\ dt.$$

for all $n = 0, 1, 2, \ldots$. It follows that

$$f(t) = \sum_{-\infty}^{\infty} a_n \exp(int)$$

with

$$a_n = \frac{1}{\pi}\int_{0}^{\pi} f(t)\cos(nt) \quad \text{for all} \quad n \in \mathbb{Z}.$$

Cosine Expansions of Even Functions

But, since cosine is even,

$$a_n = a_{-n} \quad \text{for all} \quad n > 1$$

and we can rewrite the Fourier series for f as

$$\begin{aligned} f(t) &= a_0 + \sum_{n=1}^{\infty} a_n \left(\exp(int) + \exp(-int)\right) \\ &= a_0 + \sum_{n=1}^{\infty} 2a_n cos(nt). \end{aligned}$$

Cosine Expansions of Even Functions

We conclude that:

If f is a 2π-periodic continuously differentiable function which is even, then

$$f(t) = \sum_{n=0}^{\infty} b_n \cos(nt),$$

where

$$b_0 = \frac{1}{\pi} \int_0^{\pi} f(t)\ dt$$

and

$$b_n = \frac{2}{\pi} \int_0^{\pi} f(t) \cos(nt)\ dt$$

for all integers $n \geq 1$.

Sine Expansions of Odd Functions

An analogous argument gives us the following:

If f is a 2π-periodic continuously differentiable function which is odd — that is,

$$f(t) = -f(-t) \quad \text{for all} \;\; 0 \le t \le \pi$$

— then

$$f(t) = \sum_{n=1}^{\infty} b_n \sin(nt)$$

with

$$b_n = \frac{2}{\pi} \int_0^{\pi} f(t) \sin(nt)\; dt.$$

From Fourier To Chebyshev

Given a function $f(x)$ which maps $[-1,1]$ to $\mathbb{C}$, we define the function $g : [0,\pi] \to \mathbb{C}$ via the formula

$$g(t) = f(\cos(t)).$$

Since g is even and periodic on the interval $[-\pi,\pi]$, we can then represent it via a cosine series:

$$g(t) = \sum_{n=0}^{\infty} a_n \cos(nt)$$

with

$$a_0 = \frac{1}{\pi}\int_0^{\pi} g(t)dt$$

and

$$a_n = \frac{2}{\pi}\int_0^{\pi} g(t)\cos(nt)dt, \qquad n = 1, 2, \ldots.$$

From Fourier To Chebyshev

We can also write this as:

$$f(\cos(t)) = \sum_{n=0}^{\infty} a_n \cos(nt)$$

with

$$a_0 = \frac{1}{\pi} \int_0^{\pi} f(\cos(t))dt$$

and

$$a_n = \frac{2}{\pi} \int_0^{\pi} f(\cos(t)) \cos(nt)dt, \qquad n = 1, 2, \ldots .$$

From Fourier To Chebyshev

It is instructive to change back to the original variable x.

To do so, we let $t = \arccos(x)$ in

$$a_n = \frac{2}{\pi} \int_0^{\pi} g(t) \cos(nt)\ dt.$$

This yields

$$\begin{aligned} a_n &= -\frac{2}{\pi} \int_1^{-1} g(\arccos(x)) \cos(n \arccos(x))\ \frac{dx}{\sqrt{1-x^2}} \\ &= \frac{2}{\pi} \int_{-1}^{1} f(x) \cos(n \arccos(x))\ \frac{dx}{\sqrt{1-x^2}} \end{aligned}$$

Note that

$$\frac{d}{dt} \arccos(x) = \frac{-1}{\sqrt{1-x^2}}$$

Obviously, the analogous formula holds for a_0.

From Fourier To Chebyshev

Our expansions for f can now be written in the form

$$f(x) = \sum_{n=0}^{\infty} a_n \cos(n \arccos(x))$$

with

$$a_0 = \frac{1}{\pi} \int_{-1}^{1} f(x) \frac{dx}{\sqrt{1-x^2}}$$

and

$$a_n = \frac{2}{\pi} \int_{-1}^{1} f(x) \cos(n \arccos(x)) \frac{dx}{\sqrt{1-x^2}}, \qquad n = 1, 2, \ldots.$$

From Fourier To Chebyshev

Let us introduce a name and notation for the functions which appear in our expansion of f.

For each nonnegative number λ, we call

$$T_\lambda(x) = \cos(\lambda \arccos(x))$$

the Chebyshev function of the first kind of degree λ and the function

$$U_\lambda(x) = \sin(\lambda \arccos(x))$$

the Chebyshev function of the second kind of degree λ.

Only the Chebyshev functions of integer orders appear in our expansions, and they are relatively simple functions.

Theorem

If n is a nonnegative integer, then the function

$$T_n(x) = \cos(n \arccos(x))$$

is a polynomial of degree n and the function

$$U_n(x) = \sin(n \arccos(x))$$

is of the form $q(x)\sqrt{1-x^2}$ with q a polynomial of degree $n-1$.

Proof: This follows by induction on n. When $n = 1$, we have

$$T_1(x) = \cos(\arccos(x)) = x$$

and

$$U_1(x) = \sin(\arccos(x)) = \sqrt{1-x^2},$$

so the theorem is true in the case $n = 1$. We now assume that it true for Chebyshev functions of order $n-1$ and will show that this implies it is true for Chebyshev functions of order n.

Letting $x = \cos(t)$, we have

$$T_n(x) = \cos(nt) = \cos((n-1)t)\cos(t) - \sin((n-1)t)\sin(t)$$

and

$$U_n(x) = \sin(nt) = \cos((n-1)t)\sin(t) + \sin((n-1)t)\cos(t).$$

But

$$\cos(t) = x, \quad \sin(t) = \sqrt{1-x^2}, \quad \cos((n-1)t) = T_{n-1}(x) \text{ and } \sin((n-1)t) = U_{n-1}(x),$$

so

$$T_n(x) = T_{n-1}(x)x - U_{n-1}(x)\sqrt{1-x^2}$$

and

$$U_n(x) = T_{n-1}(x)\sqrt{1-x^2} - U_{n-1}(x)x.$$

By the induction hypothesis, $U_{n-1}(x) = \sqrt{1-x^2}q_{n-2}(x)$ with q_{n-2} a polynomial of degree $n-2$ and $T_{n-1}(x) = p_{n-1}(x)$ with p_{n-1} a polynomial of degree $n-1$.

It follows that

$$T_n(x) = T_{n-1}(x)x - U_{n-1}(x)\sqrt{1-x^2} = xp_{n-1}(x) - q_{n-2}(x)(1-x^2)$$

and

$$U_n(x) = p_{n-1}(x)\sqrt{1-x^2} - xq_{n-2}(x)\sqrt{1-x^2}$$

Both $(1-x^2)q_{n-2}$ and $xp_{n-1}(x)$ are polynomials of degree n, so the first expression tells us that T_n is a polynomial of degree n

Likewise, p_{n-1} and xq_{n-1} are polynomials of degree $n-1$, so

$$U_n(x) = \left(p_{n-1}(x) - xq_{n-2}(x)\right)\sqrt{1-x^2}$$

is the product of a polynomial of degree $n-1$ with $\sqrt{1-x^2}$. ■

Chebyshev polynomials

Here are the first few Chebyshev polynomials:

$$
\begin{aligned}
T_0(x) &= 1 \\
T_1(x) &= x \\
T_2(x) &= 2x^2 - 1 \\
T_3(x) &= 4x^3 - 3x \\
T_4(x) &= 8x^4 - 8x^2 + 1 \\
T_5(x) &= 16x^5 - 20x^3 + 5x \\
&\vdots
\end{aligned}
$$

From this list, one might guess that the Chebyshev polynomials of even degree are sums of monomials even degree while the Chebyshev polynomials of odd degree are sums of monomials of odd degrees.

One might also speculate that the leading coefficient of $T_n(x)$ is 2^{n-1} for $n > 1$.

Chebyshev Expansions

If $f : [-1, 1] \to \mathbb{C}$ is a continuous function, then we call the expansion

$$f(x) = \sum_{n=0}^{\infty}{}' a_n T_n(x),$$

where

$$a_n = \frac{2}{\pi} \int_{-1}^{1} f(x) T_n(x) \frac{dx}{\sqrt{1-x^2}} \tag{3}$$

and

$$T_n(x) = \cos(n \arccos(x)),$$

the Chebyshev expansion of the function f.

The "prime" symbol by the sum is a convention which means that the first term in the sum should be multiplied by $1/2$. This allows us to use the expression (3) to define a_0 as well as for the other coefficients.

Chebyshev Expansions

Since cosine is bounded in absolute value by 1, we have

$$|T_n(x)| = |\cos(n \arccos(x)| \leq 1$$

as well. It follows that if $\{a_n\}$ are the Chebyshev coefficients of f then

$$\left| f(x) - \sum_{n=0}^{N} a_n T_n(x) \right| \leq \sum_{n=N+1}^{\infty} |a_n| \, .$$

So the rate of decay of the magntiude of the coefficients $\{a_n\}$ gives us a good idea of the accuracy of the Chebyshev expansion.

Chebyshev Expansions

Our knowledge of Fourier series allows us to draw many conclusions about the Chebyshev expansion of a function f

Convergence of Chebyshev Series

If $f : [-1, 1] \to \mathbb{C}$ is continuously differentiable, then its Chebyshev series converges absolutely and uniformly to f on the interval $[-1, 1]$.

Chebyshev Coefficients of C^k functions

Suppose that $f : [-1, 1] \to \mathbb{C}$ is k-times continuously differentiable, and that $\{a_n\}$ are its Chebyshev coefficients. Then

$$|a_n| = o\left(\frac{1}{n^k}\right).$$

Chebyshev Expansions

We know that if a periodic function g is analytic on a strip of radius $r > 0$ containing the real line and $\{a_n\}$ are its Fourier coefficients, then

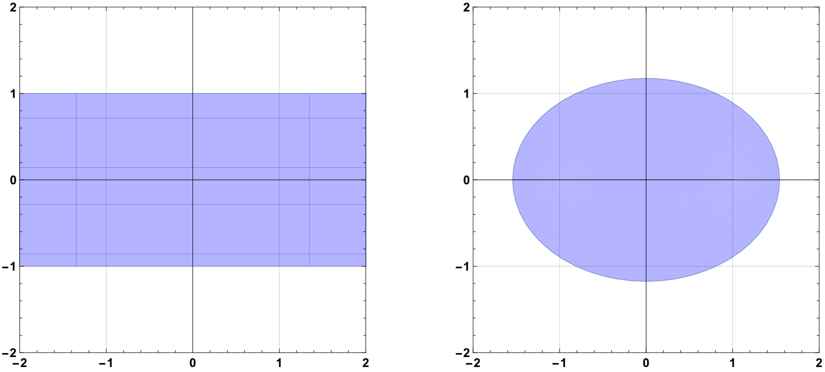
$$|a_n| = \mathcal{O}\left(\exp(-rn)\right).$$

The Chebyshev coefficients of $f(x)$ are multiples of the Fourier coefficients for $g(t) = f(\cos(t))$, so we can use this fact to make a statement about the decay of the Chebyshev coefficients of an analytic function.

The only difficult thing here is that we need to translate the condition that $g(t)$ is analytic on a strip containing the real line to a condition on $f(x)$.

Chebyshev Expansions

The mapping $x = \cos(t)$ maps the strip seen on the left into the region on the right, which is bounded by an ellipse.



So if $f(x)$ is analytic in the region shown on the left, then $g(t)$ is analytic on the region shown on the right.

Chebyshev Expansions

We can easily find a parameterization for ellipse which is the boundary of this region. It is the image of the boundary of the strip, which is parameterized by

$$\{t + ir : -\pi < t < \pi\}.$$

Since

$$\begin{aligned}\cos(t + ir) &= \frac{\exp(i(t + ir)) + \exp(-i(t + ir))}{2} \\ &= \frac{1}{2}\exp(-r)\cos(t) + \frac{1}{2}\exp(r)\cos(t) \\ &+ i\left(\frac{1}{2}\exp(-r)\sin(t) + \frac{1}{2}\exp(r)\sin(t)\right) \\ &= \cosh(r)\cos(t) + i\sinh(r)\sin(t),\end{aligned}$$

that ellipse can be parameterized as follows:

$$\begin{cases} x(t) = \cosh(r)\cos(t) \\ y(t) = \sinh(r)\sin(t) \end{cases} \quad \text{for all} \quad -\pi < t < \pi.$$

Chebyshev Expansions

Chebyshev Coefficients of Analytic Functions

If $r > 0$ and f is analytic in the region bounded by the curve

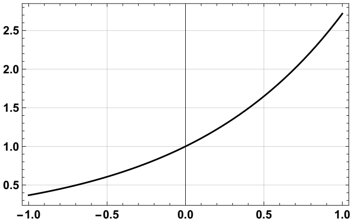
$$\begin{cases} x(t) = \cosh(r)\cos(t) \\ y(t) = \sinh(r)\sin(t) \end{cases} \quad \text{for all} \quad -\pi < t < \pi,$$

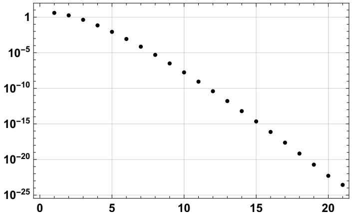
then the Chebyshev coefficients $\{a_n\}$ of f decay as

$$|a_n| = o\left(\exp(-rn)\right).$$

Chebyshev Expansions

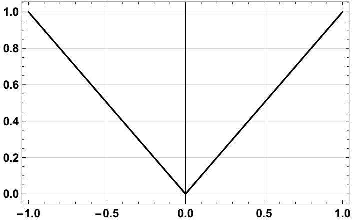
$$f(x) = \exp(x)$$

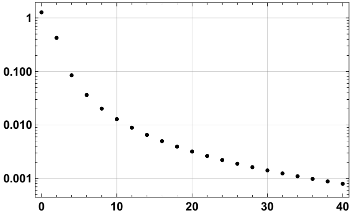




Chebyshev Expansions

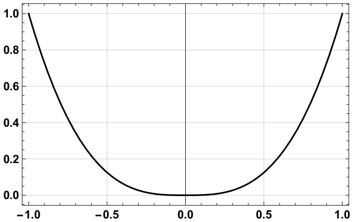
$$f(x) = |t|$$

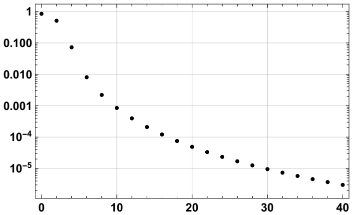




Chebyshev Expansions

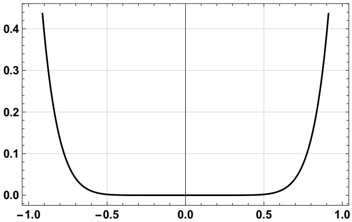
$$f(x) = |t|^3$$

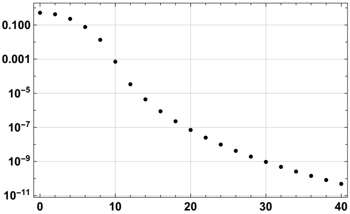




Chebyshev Expansions

$$f(x) = |t|^9$$





MAT128A: Numerical Analysis
Lecture Eleven: Calculation of Chebyshev Expansions, Part II

October 19, 2018

Chebyshev expansions

Suppose that f is a continously differentiable function on the interval $[-1, 1]$. The Chebyshev expansion of f is essentially the same as the cosine expansion of

$$g(t) = f(\cos(t)),$$

which is an even periodic function on $[-\pi, \pi]$.

The cosine expansion of g is

$$g(t) = \sum_{n=0}^{\infty}{}' a_n \cos(nt)$$

where

$$a_n = \frac{2}{\pi} \int_0^{\pi} g(t) \cos(nt)\ dt$$

and the prime sign next to the summation means that the first term in the series is scaled by $\frac{1}{2}$.

Chebyshev expansions

If we make the change of variables $t = \arccos(x)$, we obtain

$$f(x) = {\sum_{n=0}^{\infty}}' a_n \cos(n \arccos(x))$$

where

$$a_n = \frac{2}{\pi} \int_{-1}^{1} f(x) \cos(n \arccos(x)) \frac{dx}{\sqrt{1-x^2}}.$$

As we saw last time, the function $\cos(n \arccos(x))$ is a polynomial of degree n. We call it the Chebyshev polynomial of degree n and denote it by T_n.

Chebyshev expansions

Using the notation T_n for Chebyshev polynomials, we see that the Chebyshev expansion of f is

$$f(x) = {\sum_{n=0}^{\infty}}' a_n T_n(x),$$

where

$$a_n = \frac{2}{\pi} \int_{-1}^{1} f(x) T_n(x) \frac{dx}{\sqrt{1 - x^2}}.$$

Two Quadrature Rules for Calculating Cosine Expansions

We are going to derive **two** different quadrature rules for computing the coefficients in cosine expansions. Each of these will give us a method for computing Chebyshev expansions.

These quadratures correspond to two different sets of points which are commonly used as discretization points for Chebyshev expansions.

Two Quadrature Rules for Calculating Cosine Expansions

The set of points

$$\left\{\cos\left(\frac{\pi j}{n-1}\right) : j = 0, 1, 2, \ldots, n-1\right\}$$

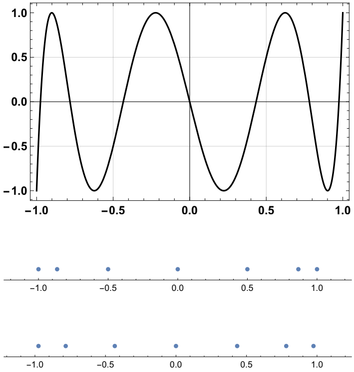
is called the n-**point Chebyshev extrema grid**. It is so named because the local maximums and minimums of the function $T_{n-1}(x)$ occur at these points. Note that the endpoints -1 and 1 of the interval $[-1, 1]$ are included in this set.

The set of points

$$\left\{\cos\left(\frac{j+\frac{1}{2}}{n}\pi\right) : j = 0, 1, 2, \ldots, n-1\right\}$$

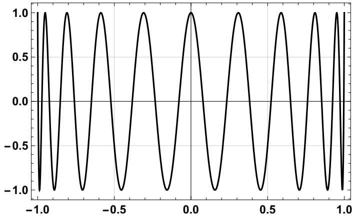
is called the n-**point Chebyshev root grid**. It is so named because these the roots of the function $T_n(x)$ occur at these points. Note that the endpoints -1 and 1 of the interval $[-1, 1]$ are **not** included in this set.

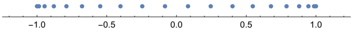
Two Quadrature Rules for Calculating Cosine Expansions

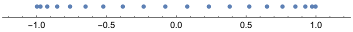


Two Quadrature Rules for Calculating Cosine Expansions

Observation: the nodes of both grids cluster close to the endpoints ± 1 of the intervals.







A Rule for the Root Grid

In this lecture, we will construct a quadrature rule which will allow us to compute Chebyshev expansions given the values of a function on the root grid.

We start by introducing a variant of the periodic trapezoidal rule which integrates exponential functions.

A Rule for the Root Grid

You can verify that the quadrature rule

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} f(t)\,dt \approx \frac{1}{2N}\left(\sum_{j=1}^{N} f\left(\frac{j-\frac{1}{2}}{N}\pi\right) + \sum_{j=1}^{N} f\left(\frac{-j+\frac{1}{2}}{N}\pi\right)\right)$$

is exact for the exponential functions

$$\{\exp(int) : n \in \mathbb{Z} \ \text{ such that } \ |n| \leq 2N-1.\}\,.$$

This is a $2N$-point rule which integrates exactly a collection of $4N-1$ functions. This is similar to the periodic trapezoidal rule; note, though, that the nodes in this quadrature are symmetric about 0 and do not include 0.

A Rule for the Root Grid

If f is even and in the span of

$$\{\exp(int) : n \in \mathbb{Z} \;\text{ such that }\; |n| \leq 2N - 1.\}\,.$$

then we can rewrite the previous identity as

$$\frac{1}{\pi}\int_0^{\pi} f(t)\, dt = \frac{1}{N}\sum_{j=1}^{N} f\left(\frac{j-\frac{1}{2}}{N}\pi\right).$$

This is an N-point quadrature rule which is exact for the functions

$$\{\cos(nt) : n = 0, 1, \ldots, 2N-1\}\,.$$

A Rule for the Root Grid

The cosine expansion of an even periodic function g is

$$g(t) = \sum_{n=0}^{N}{}' a_n \cos(nt),$$

where

$$a_m = \frac{2}{\pi} \int_0^{\pi} g(t) \cos(mt)\ dt.$$

The function $g(t)\cos(mt)$ is in the span of the set

$$\{\cos(nt) : n = 0, 1, \ldots, 2N+1\},$$

which is integrated exactly by the $(N+1)$-point rule we just constructed.

A Rule for the Root Grid

Theorem

If g is in the span of the set

$$\{\cos(nt) : n = 0, 1, \ldots, N\},$$

then the cosine expansion of g is

$$g(t) = \sum_{n=0}^{N}{}' a_n \cos(nt),$$

where

$$a_m = \frac{2}{\pi}\int_0^{\pi} g(t)\cos(mt)\ dt = \frac{2}{N+1}\sum_{j=0}^{N} g\left(\frac{j+\frac{1}{2}}{N+1}\pi\right)\cos\left(\frac{j+\frac{1}{2}}{N+1}m\pi\right)$$

for each $m = 0, 1, \ldots, N$.

A Rule for the Root Grid

Once again, we introduce the variable $x = \cos(t)$ in order to make an analagous statement for the Chebyshev expansion of a function $f(x)$.

Theorem

If $f(x)$ is a polynomial of degree N, then

$$f(x) = \sum_{n=0}^{N}{}' a_n T_n(x),$$

where

$$\begin{aligned} a_m &= \frac{2}{\pi} \int_{-1}^{1} f(x) T_m(x) \, \frac{dx}{\sqrt{1-x^2}} \\ &= \frac{2}{N+1} \sum_{j=0}^{N} f\left(\cos\left(\frac{j+\frac{1}{2}}{N+1}\pi \right) \right) T_m\left(\cos\left(\frac{j+\frac{1}{2}}{N+1}\pi \right) \right) \end{aligned}$$

for each $m = 0, 1, \ldots, N$.

Review

We have developed two techniques for computing the Chebyshev expansion of a polynomial of degree N.

The first uses the values of the polynomial at the $(N+1)$ nodes $\{x_0, x_1, \ldots, x_N\}$ of the Chebyshev extrema grid, which are defined via the formula

$$x_j = \cos\left(\frac{\pi}{N} j\right). \tag{1}$$

Theorem

If f is a polynomial of degree N and $x_0, \ldots, x_N$ are the nodes of the $(N+1)$-point Chebyshev extrema grid defined via (1), then

$$f(x) = \sum_{n=0}^{N}{}'' b_n T_n(x),$$

where the coefficients are defined via the formula

$$b_m = \frac{2}{N} \sum_{j=0}^{N}{}'' f(x_j)\, T_m(x_j).$$

Review

The second uses the values of the polynomial at the $(N+1)$ nodes $\{\widetilde{x_0}, \widetilde{x_1}, \ldots, \widetilde{x_N}\}$ of the Chebyshev root grid, which are defined via the formula

$$\tilde{x}_j = \cos\left(\frac{j+\frac{1}{2}}{N+1}\pi\right). \tag{2}$$

Theorem

If f is a polynomial of degree N and $\{\tilde{x_0}, \ldots, \tilde{x_n}\}$ are the nodes of the $(N+1)$-point Chebyshev root grid, then

$$f(x) = \sum_{n=0}^{N}{}' a_n T_n(x)$$

where the coefficients are defined via the formula

$$a_m = \frac{2}{N+1}\sum_{j=0}^{N} f\left(\tilde{x_j}\right) T_m\left(\tilde{x_j}\right).$$

Aliasing Error

In the case of Fourier series, we found that if we used the $(2N+1)$-point periodic trapezoidal rule to form approximations $\{\tilde{a}_n\}$ of the coefficients in the Fourier series

$$f(t) = \sum_{n=-\infty}^{\infty} a_n \exp(int),$$

then

$$\left| f(t) - \sum_{n=-N}^{N} \tilde{a}_n \exp(int) \right| \le 2 \sum_{|n|>N} |a_n| \, .$$

That is, the error is at most twice the bound

$$\left| f(t) - \sum_{n=-N}^{N} a_n \exp(int) \right| \le \sum_{|n|>N} |a_n|$$

we have for the truncated series with exact coefficients.

Aliasing Error

Since the Chebyshev expansion of $f(x)$ is merely the Fourier expansion of $f(\cos(t))$ (slightly rewritten as a cosine expansion), the same is obviously true for Chebyshev expansions.

Theorem

If

$$f(x) = {\sum_{n=0}^{\infty}}' b_n T_n(x)$$

and $\{\widetilde{b_0}, \ldots, \widetilde{b_N}\}$ are defined by the formula

$$\widetilde{b_m} = \frac{2}{N} {\sum_{j=0}^{N}}'' f\left(\cos\left(\frac{\pi}{N} j\right)\right) T_m\left(\cos\left(\frac{\pi}{N} j\right)\right),$$

then

$$\left| f(x) - {\sum_{n=0}^{N}}'' \widetilde{b_n} T_n(x) \right| \leq 2 \sum_{|n|>N} |b_n| .$$

Aliasing Error

Since the Chebyshev expansion of $f(x)$ is merely the Fourier expansion of $f(\cos(t))$ (slightly rewritten as a cosine expansion), the same is obviously true for Chebyshev expansions.

Theorem

If

$$f(x) = \sum_{n=0}^{\infty}{}' b_n T_n(x)$$

and $\{\widetilde{b_0}, \ldots, \widetilde{b_N}\}$ are defined by the formula

$$\widetilde{b_m} = \frac{2}{N} \sum_{j=0}^{N} f\left(\cos\left(\frac{j+\frac{1}{2}}{N+1}\pi\right)\right) T_m\left(\cos\left(\frac{j+\frac{1}{2}}{N+1}\pi\right)\right).$$

then

$$\left| f(x) - \sum_{n=0}^{N}{}' \widetilde{b}_n T_n(x) \right| \leq 2 \sum_{|n|>N} |b_n|.$$

Evaluating Chebyshev Expansions

Now that we know how to compute the coefficients in an approximate expansion of f of the form

$$f(x) \approx \sum_{n=0}^{N} a_n T_n(x), \tag{3}$$

it is natural to ask how we might go about evaluating this sum for a specified value of x given the coefficients $\{a_n\}$.

The most obvious method is probably to compute the values

$$T_0(x), T_1(x), \ldots, T_N(x)$$

using the formula

$$T_n(x) = \cos(n \arccos(x)),$$

and then sum (3) in the "obvious" way.

Evaluating Chebyshev Expansions

There is nothing really wrong with this. The use of the formula

$$T_n(x) = \cos(n \arccos(x)) \tag{4}$$

will lead to roundoff error for large value of n, but we will very rarely use Chebyshev expansions of large enough order to make this a problem (we will talk shortly about what we will do instead).

The bigger issue is that (4) tends to be costly to evaluate. On my laptop, (4) it takes about 5×10^{-7} seconds to compute $T_n(x)$ via (4). This makes the cost of evaluating all of the values

$$T_0(x), \ldots, T_N(x)$$

around $5N \times 10^{-7}$ seconds.

Evaluating Chebyshev Expansions

There is a procedure which is much faster, at least for relatively small values of N.

In your homework, you showed that the Chebyshev polynomials satisfy the three-term recurrence relation.

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x). \tag{5}$$

Moreover, it is evident that

$$T_0(x) = 1 \quad \text{and} \quad T_1(x) = x. \tag{6}$$

Together (5) and (6) allow us to compute the values

$$T_0(x), T_1(x), \ldots, T_N(x)$$

quite quickly. Note too, that this procedure does not suffer from roundoff error as long as x is not close to ± 1.

Evaluating Chebyshev Expansions

In your homework assignment this week, you will show that a technique called Clenshaw's Recurrence Formula can further accelerate the evaluation of Chebyshev expansions.

Although we use it in the case of Chebyshev polynomials, it can reduce the cost of evaluating any sum of the form

$$\sum_{n=0}^{N} a_n \phi_n(x)$$

where the functions $\{\phi_n\}$ satisfy a three-term recurrence relation.

Evaluating Chebyshev Expansions

In the particular case of Chebyshev polynomials, Clenshaw's recurrence formula for evaluating

$$\sum_{n=0}^{N} a_n T_n(x)$$

proceeds by computing the sequence

$$b_{N+2}(x), b_{N+1}(x), b_N(x), \ldots, b_0(x)$$

defined via the formulas

$$b_{N+2}(x) = b_{N+1}(x) = 0 \quad \text{and} \quad b_n(x) = a_n + 2x b_{n+1}(x) - b_{n+2}(x).$$

It turns out (as you will show) that

$$b_0(x) = \sum_{n=0}^{N} a_n T_n(x).$$

Functions given on Intervals Other than $[-1, 1]$

If f is a smooth function defined on an interval other than $[-1, 1]$, then we can still represent it using a Chebyshev expansion.

If f is defined on $[a, b]$ with $a < b$, then we map $[-1, 1]$ onto $[a, b]$ using the affine transformation

$$\Lambda(x) = \frac{b-a}{2}x + \frac{b+a}{2}.$$

Note that Λ is invertible — indeed, if $y = \Lambda x$, then

$$x = \frac{2}{b-a}y - \frac{b+a}{b-a}.$$

That is,

$$\Lambda^{-1}(y) = \frac{2}{b-a}y - \frac{b+a}{b-a}.$$

The Chebyshev expansion of $f : [a, b] \to \mathbb{R}$ is

$$f(y) = {\sum_{n=0}^{\infty}}' a_n T_n(\Lambda^{-1} y),$$

with

$$a_n = \frac{2}{\pi} \int_0^{\pi} f(\Lambda x) T_n(x)\, dx.$$

The coefficients can be approximated as

$$a_n \approx \frac{2}{N+1} \sum_{j=0}^{N} f\left(\frac{b-a}{2}\widetilde{x}_j + \frac{b+a}{2}\right) T_m\left(\widetilde{x}_j\right),$$

where

$$\widetilde{x}_j = \cos\left(\frac{j+\frac{1}{2}}{N+1}\pi\right), \quad j = 0, 1, \ldots, N.$$

We call the points

$$\left\{\frac{b-a}{2}\widetilde{x}_j + \frac{b+a}{2} : j = 0, 1, \ldots, N\right\}$$

the $(N+1)$-point Chebyshev root grid on the interval $[a, b]$.

Chebyshev Expansions of Large Order

Although we will pursue this line of thought in this class, it is worth noting that because of the relationship between Chebyshev and Fourier expansions, Chebyshev coefficients can be computed using the Fast Fourier Transform.

This means that expansions of truly huge orders ($1,000,000$ is no problem) can be constructed quite rapidly.

It is quite expensive to evaluate a Chebyshev expansion with a large number of terms at on point, but if we wish to evaluate it at a large number of points, there are methods related to the Fast Fourier Transform which reduce the cost of doing so quite dramatically.

MAT128A: Numerical Analysis
Lecture Twelve: Piecewise Chebyshev Expansions

October 24, 2018

Piecewise Chebyshev Expansions

We say the function

$$f(x) = \sqrt{1 - x^2}$$

is singular at the points ± 1 because its derivative become infinite at those points. In your homework, you are asked to compute the Chebyshev coefficients $\{a_n\}$ of this function.

You will find that

$$|a_n| = \mathcal{O}\left(\frac{1}{n^2}\right),$$

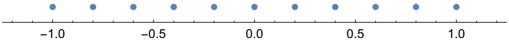
so that it would take a Chebyshev expansions with many terms to approximate f to high accuracy.

It is easy to construct many other examples of this type, where a singularity or discontinuity leads to difficulty in representing a function via a Chebyshev expansion.

Piecewise Chebyshev Expansions

A good solution to this problem is to use a **piecewise Chebyshev expansions** to represent f instead of a single Chebyshev expansion for f.

The idea is to decompose the domain $[-1, 1]$ on which f is defined into subintervals and to represent f via a Chebyshev expansion on each subinterval.



Piecewise Chebyshev Expansions

More explicitly, given a partition

$$a = a_1 < a_2 < a_3 < \cdots < a_m = b$$

of the interval $[a, b]$, we form a Chebyshev expansion for each of the functions $f_1, \ldots, f_{m-1}$ defined via

$$f_k(x) = f\left(\frac{a_{k+1} - a_k}{2}x + \frac{a_{k+1} + a_k}{2}\right)$$

The computation of the Chebyshev expansions is straightforward since we know how to compute a Chebyshev expansion for the restriction of f to each subinterval.

To evaluate f at a point x, we find an interval $[a_k, a_{k+1}]$ containing x, let

$$\tilde{x} = \frac{2}{a_{k+1} - a_k}x - \frac{a_{k+1} + a_k}{a_{k+1} - a_k}$$

and evaluate f_k at the point $\tilde{x}$ using its Chebyshev expansion.

Piecewise Chebyshev Expansions

We can find an interval $[c, d]$ containing the point x via a straightforward brute-force search in $\mathcal{O}(m)$ operations.

The pseudocode for that might look something like:

```
for k = 1 to m − 2 do
    if x < a_{k+1} then
        break
    end if
    c = a_k
    d = a_{k+1}
end for
```

This assumes no error checking is needed. Unless the number of intervals is quite large, this is the most efficient method to find the interval in question.

Piecewise Chebyshev Expansions

If the number of intervals is large, we can find the interval $[c, d]$ containing the point x via a binary search in $\mathcal{O}(\log(m))$ operations.

The pseudocode for that might look something like:

$i_1 = 1$
$i_2 = m$
while $i_2 > i_1 + 1$ **do**
    $k = (i_1 + i_2)/2$
    if $x < a_k$ **then**
        $i_2 = k$
    else
        $i_1 = k$
    end if
end while
$c = a_{i_1}$
$d = a_{i_1+1}$

Piecewise Chebyshev Expansions

In some cases, we can find the correct interval in $\mathcal{O}(1)$ operations. For intance, if we use equispaced intervals or intervals whose endpoints conform to a known pattern — e.g.,

$$\left[2^{-k}, 2^{-k-1}\right].$$

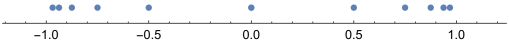
In any case, the worst case operation count for evaluating $f(x)$ at a point is

$$\mathcal{O}\left(\log(m) + N\right),$$

where N is the number of terms in the Chebyshev expansions we use.

The Chebyshev Expansion of a Singular Function

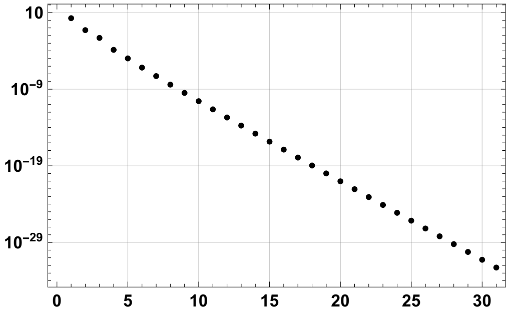
In the case of a function like $f(x) = \sqrt{1 - x^2}$, the usual course of action would be to represent it using intervals which become smaller as they approach the singularities at ± 1.



This ensures that $f(x)$ is analytic in a neighborhood of each of the subintervals, with the consequence that a Chebyshev expansions of modest length will represent the function over that interval.

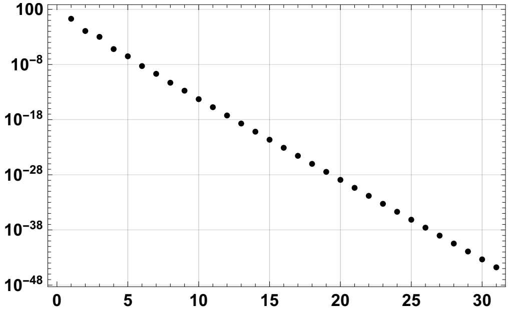
The Chebyshev Expansion of a Singular Function

$$f(x) = \sqrt{1 - x^2} \quad \text{on the interval} \quad \left[\frac{1}{2^{k+1}}, \frac{1}{2^k}\right] \quad \text{with} \quad k = 1$$



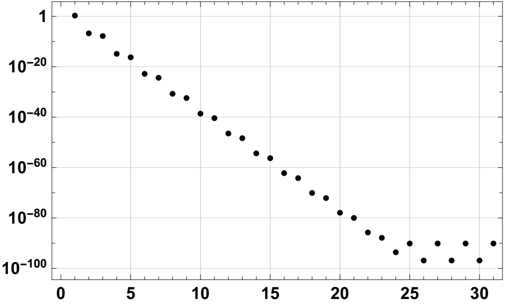
The Chebyshev Expansion of a Singular Function

$$f(x) = \sqrt{1 - x^2} \quad \text{on the interval} \quad \left[\frac{1}{2^{k+1}}, \frac{1}{2^k}\right] \quad \text{with} \quad k = 2$$



The Chebyshev Expansion of a Singular Function

$$f(x) = \sqrt{1 - x^2} \quad \text{on the interval} \quad \left[\frac{1}{2^{k+1}}, \frac{1}{2^k}\right] \quad \text{with} \quad k = 12$$



The Chebyshev Expansion of a Singular Function

Let's do an estimate to see which of these approaches we think will be more accurate. Should we:

- Represent $f(x) = \sqrt{1-x^2}$ on the interval $[-1, 1]$ using a single Chebyshev expansion of large order, or
- Represent $f(x) = \sqrt{1-x^2}$ on the interval $[-1, 1]$ using piecewise Chebyshev expansions of a relatively small, fixed order?

The Chebyshev Expansion of a Singular Function

If we use 30-term expansions on each of the intervals

$$\left[1 - 2^{-k+1}, 1 - 2^{-k},\right] \quad k = 0, 1, \ldots, 40$$

and

$$\left[-\left(1 - 2^{-k}\right), -\left(1 - 2^{-k+1}\right),\right] \quad k = 0, 1, \ldots, 40,$$

then we would need

$$30 \times 41 \times 2 = 2460$$

coefficients to represent $f(x)$. This would let us evaluate f to near machine precision accuracy anywhere on the interval

$$\left[-1 + 2^{-40}, 1 - 2^{-40}\right].$$

The Chebyshev Expansion of a Singular Function

To get similar accuracy using one Chebyshev expansion over the whole of the inerval $[-1, 1]$, we would need on the order of

$$\sqrt{10^{16}} = 10^8 = 100,000,000$$

points since the Chebyshev coefficients of $f(x) = \sqrt{1 - x^2}$ behave as

$$|a_n| = \mathcal{O}\left(\frac{1}{n^2}\right).$$

Adaptive Piecewise Chebyshev Discretization

Given a particular function f on an interval $[a, b]$, it might not be obvious how to go about partitioning $[a, b]$ so that f is accurately represented by Chebyshev expansions of a specified order n on each resulting subinterval.

We will now discuss a fairly general *adaptive discretization procedure* which will automatically determine some collection of discretization subintervals for us.

This procedure is not foolproof, it can fail to construct a sufficiently accurate discretization. However, it work quite well in practice.

Adaptive Piecewise Chebyshev Discretization

The algorithm takes as input the length n of the Chebyshev expansions to use to represent the function f, the endpoints a and b of the interval on which f is given, a means of evaluating f at any point on the interval $[a, b]$, and a real number $\epsilon > 0$ which will affect the accuracy of the obtained approximation of f.

The output will be a list of disjoint subintervals which cover $[a, b]$ and which hopefully have the property that the restriction of f to each subinterval can be represented with relative accuracy on the order of ϵ via an n-term Chebyshev expansion.

The algorithm is iterative, and make use of two lists: a list of output intervals and a list of intervals under consideration. At the outset, the list of output intervals is empty and the list of intervals under consideration consists only of $[a, b]$.

Adaptive Piecewise Chebyshev Discretization

Here is the algorithm:

1. Extract an interval $[c, d]$ from the list of intervals under consideration.
2. Approximate the Chebyshev coefficients $a_0, a_1, \dots, a_{n-1}$ for the restriction of f to $[c, d]$ using the mechanisms we have developed.
3. Let $d_1 = \sum_{j=0}^{n} |a_j|^2$ and $d_2 = \sum_{j=\lceil n/2 \rceil}^{n} |a_j|^2$.
4. If $d_2 < \epsilon^2 d_1$ then add $[c, d]$ to the list of accepted intervals. Otherwise, add $\left[c, \frac{c+d}{2}\right]$ and $\left[\frac{c+d}{2}, d\right]$ to the list of intervals under consideration.
5. If the list of intervals under consideration is empty, the algorithm terminates; otherwise, goto Step 1.

Adaptive Piecewise Chebyshev Discretization

There are many conditions which can be used to decide if the coefficients

$$a_0, a_1, \ldots, a_{n-1}$$

in a Chebyshev expansion decay sufficiently fast. Note, though, that some measure of the relative magnitudes of the trailing coefficients should be used rather than a measure of their absolute magnitudes.

Other good choices include

$$\max_{j=\lceil n/2 \rceil, \ldots, n} |a_j| < \epsilon \max_{j=0,\ldots,n} |a_j|$$

and

$$\sum_{j=\lceil n/2 \rceil} |a_j| < < \epsilon \sum_{j=0}^{n} |a_j|.$$

Adaptive Piecewise Chebyshev Discretization

In cases in which a function $f : [a, b] \to \mathbb{R}$ has a singularity in the interval (a, b), it is usually best (but not always necessary) to introduce a partition point at the singularity.

When we use the preecding algorithm to discretize

$$f(x) = \begin{cases} \cos(13t^2) & t < 0.41 \\ \dfrac{\exp(t^2 - 1)}{1 + t^{16}} & t \geq .41 \end{cases}$$

over the interval $[0, 1]$ with $n = 30$, we get 60 intervals for a total of 1800 coefficients.

On the other hand, if we apply the same algorithm on the interval $[0, 0.41]$ and then again on the interval $[0.41, 1]$, then the total number of intervals produced is 12, for a total of 360 coefficients.

In some cases, the algorithm will fail unless a singularity is treated in this way. One example is given by the function:

$$g(x) = \sqrt{|x - 0.41|}.$$

MAT128A: Numerical Analysis
Lecture Thirteen: Chebyshev Interpolation

October 26, 2018

Polynomial Interpolation

We have seen that if a function f is continuously differentiable, then it has a uniformly convergent Chebyshev expansion.

That is, the series

$$\sum_{n=0}^{\infty}{}' a_n T_n(x), \quad a_n = \frac{2}{\pi} \int_{-1}^{1} f(x) T_n(x) \frac{dx}{\sqrt{1-x^2}}$$

converges uniformly to f.

The uniform converence of the series means that

$$\sup_{x \in [-1,1]} \left| f(x) - \sum_{n=0}^{N}{}' a_n T_n(x) \right| \to 0 \text{ as } N \to \infty.$$

Polynomial Interpolation

In practice, we represent a function f using a finite Chebyshev expanions whose coefficients are computed via one of two quadrature rules.

For instance, we might approximate f as

$$f(x) \approx \sum_{n=0}^{N}{}' \widetilde{a}_n T_n(x),$$

where

$$\widetilde{a_m} = \frac{2}{N+1} \sum_{j=0}^{N} f\left(x_j\right) T_m\left(x_j\right)$$

with the nodes $x_0, \ldots, x_N$ defined via

$$x_j = \cos\left(\frac{j+\frac{1}{2}}{N+1}\pi\right)$$

Polynomial Interpolation

We derived all of these expressions without making explicit reference to **polynomial interpolation**, but we will now make the connection between Chebyshev expansions and polynomial interpolation clear.

The idea behind polynomial interpolation is to find a polynomial which agrees with $f(x)$ at a collection of specified points.

That is, given points $x_0, \ldots, x_N$, we seek a polynomial p such that

$$p(x_j) = f(x_j) \quad \text{for all} \quad j = 0, 1, \ldots, N.$$

Definition

We say that the polynomial $p(x)$ interpolates f at the points $x_0, x_1, \ldots, x_n$ if

$$f(x_j) = p(x_j)$$

for all $j = 0, 1, \ldots, N$.

Polynomial Interpolation

Theorem

If $x_0, x_1, \ldots, x_N$ are distinct points on the real line and $f : \mathbb{R} \to \mathbb{R}$, then there is a unique polynomial p of degree N which interpolates f at the points $x_0, \ldots, x_N$.

Proof:

Any polynomial of degree N can be written in the form

$$p(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_N x^N.$$

That p agrees with f at the point x_j means that the equation

$$a_0 + a_1 x_j + a_2 x_j^2 + \ldots + a_N x_j^N = f(x_j)$$

is satisfied.

Polynomial Interpolation

There are $N+1$ such equations which must be satisfied:

$$\begin{aligned} a_0 + a_1 x_0 + a_2 x_0^2 + \ldots + a_N x_0^N &= f(x_0) \\ a_0 + a_1 x_1 + a_2 x_1^2 + \ldots + a_N x_1^N &= f(x_1) \\ &\vdots \\ a_0 + a_1 x_N + a_2 x_N^2 + \ldots + a_N x_N^N &= f(x_N) \end{aligned}$$

We can write this system of equations in the form

$$\begin{pmatrix} 1 & x_0 & x_0^2 & x_0^3 & \cdots & x_0^N \\ 1 & x_1 & x_1^2 & x_1^3 & \cdots & x_1^N \\ \vdots & & \ldots & & \ddots & \vdots \\ 1 & x_N & x_N^2 & x_N^3 & \cdots & x_N^N \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ f(x_2) \\ \vdots \\ f(x_N) \end{pmatrix}$$

Polynomial Interpolation

The matrix

$$\begin{pmatrix} 1 & x_0 & x_0^2 & x_0^3 & \cdots & x_0^N \\ 1 & x_1 & x_1^2 & x_1^3 & \cdots & x_1^N \\ \vdots & & \cdots & & \ddots & \vdots \\ 1 & x_N & x_N^2 & x_N^3 & \cdots & x_N^N \end{pmatrix}$$

is a "Vandermonde matrix." Its determinant is nonzero provided the points $x_0, x_1, \ldots, x_N$ are distinct. In fact, its determinant is

$$\prod_{0 \le i \le j \le N} (x_j - x_i)$$

which can be establish using column and row operations (the procedure is tedious to write out, but fairly straightforward).

Polynomial Interpolation

That the Vandermonde matrix is invertible tells us that the system of equations

$$\begin{aligned} a_0 + a_1 x_0 + a_2 x_0^2 + \ldots + a_N x_0^N &= f(x_0) \\ a_0 + a_1 x_1 + a_2 x_1^2 + \ldots + a_N x_1^N &= f(x_1) \\ &\vdots \\ a_0 + a_1 x_N + a_2 x_N^2 + \ldots + a_N x_N^N &= f(x_N) \end{aligned}$$

has a unique solution. It follows that there is a unique polynomial of degree N which interpolates p at the nodes $x_0, \ldots, x_N$. $\blacksquare$

Chebyshev Interpolation

Theorem

Suppose that $f : [-1, 1] \to \mathbb{R}$ is a continuous function, $x_0, x_1, \ldots, x_N$ are defined by

$$x_j = \cos\left(\frac{j\pi}{N}\right),$$

and $a_0, a_1, \ldots, a_N$ are given by the formula

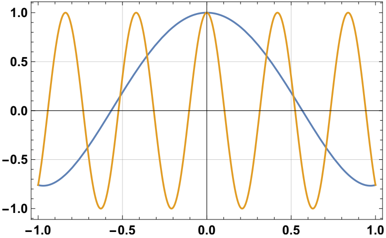
$$a_m = \frac{2}{N} \sum_{j=0}^{N}{}'' f(x_j)\, T_m(x_j).$$

Then

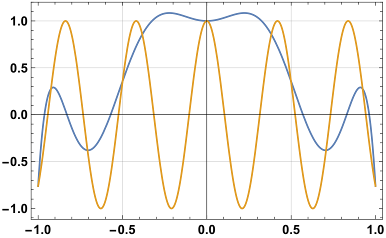
$$\sum_{n=0}^{N}{}'' a_n T_n(x)$$

is the unique polynomial of degree N which interpolates f at the points

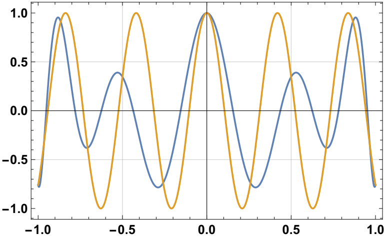
$$x_0, x_1, \ldots, x_N.$$



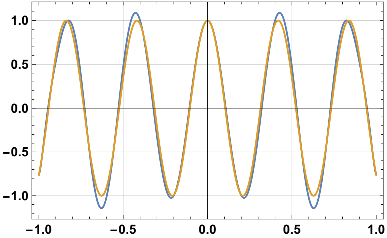
$$f(x) = \cos(15x), \quad N = 4$$



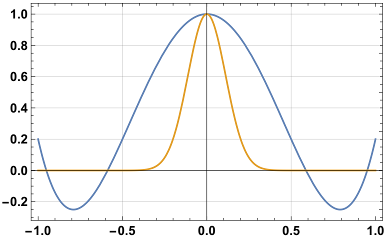
$$f(x) = \cos(15x), \quad N = 8$$



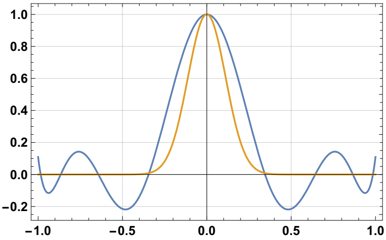
$$f(x) = \cos(15x), \quad N = 12$$



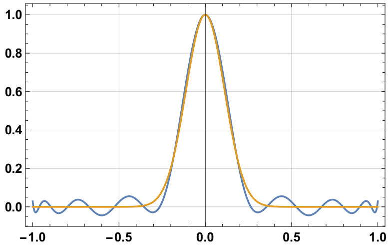
$$f(x) = \cos(15x), \quad N = 16$$



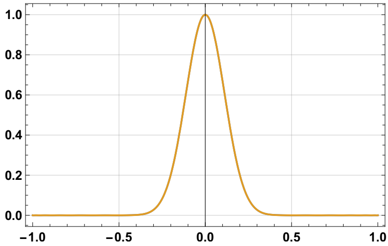
$$f(x) = \exp(-40x^2), \quad N = 4$$



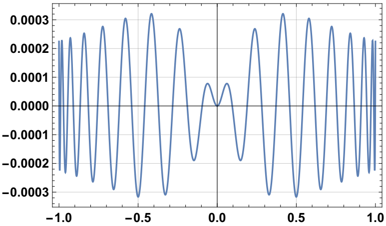
$$f(x) = \exp(-40x^2), \quad N = 8$$



$$f(x) = \exp(-40x^2), \quad N = 16$$



$$f(x) = \exp(-40x^2), \quad N = 32$$



$$f(x) = \exp(-40x^2), \quad N = 32$$

Chebyshev Interpolation

Proof:

Let p is the unique polynomial of degree N which interpolates f at $x_0, x_1, \ldots, x_N$. We can write p as

$$p(x) = \sum_{n=0}^{N}{}'' c_n T_n(x)$$

since any polynomial of degree N can be expressed in this way. Since p interpolates f, we have

$$f(x_i) = p(x_i) = \sum_{n=0}^{N}{}'' c_n T_n(x_i)$$

for all $i = 0, 1, \ldots, N$. It follows that

$$\begin{aligned}\frac{2}{N}\sum_{i=0}^{N}{}'' f(x_i) T_j(x_i) &= \frac{2}{N}\sum_{i=0}^{N}{}'' \sum_{n=0}^{N}{}'' c_n T_n(x_i) T_j(x_i)\\ &= \sum_{n=0}^{N}{}'' c_n \left(\frac{2}{N}\sum_{i=0}^{N}{}'' T_n(x_i) T_j(x_i)\right) = c_j.\end{aligned}$$

Chebyshev Interpolation

So

$$p(x) = \sum_{n=0}^{N}{}'' c_n T_n(x)$$

with

$$c_n = \frac{2}{N} \sum_{i=0}^{N}{}'' f(x_i) T_n(x_i),$$

which is what we wanted to prove. ∎

Chebyshev Interpolation

We omit the proof of the following since it is extremely similar to the proof of the preceeding theorem.

Theorem

Suppose that $f : [-1, 1] \to \mathbb{R}$ is a continuous function, $\widetilde{x_0}, \widetilde{x_1}, \ldots, \widetilde{x_N}$ are defined by

$$\widetilde{x_j} = \cos\left(\frac{j+\frac{1}{2}}{N+1}\pi\right),$$

and $a_0, a_1, \ldots, a_N$ are given by the formula

$$a_m = \frac{2}{N+1}\sum_{j=0}^{N} f\left(x_j\right) T_m\left(x_j\right).$$

Then

$$\sum_{n=0}^{N}{}' a_n T_n(x)$$

is the unique polynomial of degree N which interpolates f at the points

$$\widetilde{x_0}, \widetilde{x_1}, \ldots, \widetilde{x_N}.$$

Polynomial Interpolation

All of this suggests that we investigate polynomial interpolation in more generality.

That is, we should consider polynomials which interpolate f at points other than the Chebyshev nodes.

We will do precisely that, starting in the next lecture.

MAT128A: Numerical Analysis
Lecture Fourteen: Polynomial Interpolation

October 29, 2018

Polynomial Interpolation

In the last lecture, we saw that the polynomial

$$p(x) = {\sum_{n=0}^{N}}{}' a_n T_n(x),$$

where

$$a_n = \frac{2}{N+1} \sum_{j=0}^{N} f\left(\cos\left(\frac{j+\frac{1}{2}}{N+1}\pi\right)\right) T_n\left(\frac{j+\frac{1}{2}}{N+1}\pi\right),$$

is the unique polynomial of degree N which interpolates the function f at the points

$$\left\{\cos\left(\frac{j+\frac{1}{2}}{N+1}\pi\right) : j = 0, 1, \ldots, N\right\}.$$

The Lagrange Interpolation Formula

We will now develop a method for constructing the unique polynomial of degree N which interpolates a function f at an arbitrary collection of N distinct points

$$x_0, x_1, \ldots, x_N.$$

To that end, for each $n = 0, \ldots, N$, we let L_j be defined by

$$L_j(x) = \prod_{\substack{0 \le i \le N \\ i \neq j}} \frac{x - x_i}{x_j - x_i}.$$

Then L_J is a polynomial of degree N,

$$L_j(x_j) = \prod_{\substack{0 \le i \le N \\ i \neq j}} \frac{x_j - x_i}{x_j - x_i} = 1,$$

and

$$L_j(x_k) = \prod_{\substack{0 \le i \le N \\ i \neq j}} \frac{x_k - x_i}{x_k - x_i} = 0$$

for any $k \neq j$.

The Lagrange Interpolation Formula

In other words,

$$L_j(x) = \prod_{\substack{0 \le i \le N \\ i \ne j}} \frac{x - x_i}{x_j - x_i}.$$

is a polynomial of degree N such that

$$L_j(x_i) = \begin{cases} 1 & i = j \\ 0 & i \ne j. \end{cases}$$

We now let

$$L(x) = \sum_{j=0}^{N} f(x_j) L_j(x).$$

The Lagrange Interpolation Formula

Since $L_j(x_i) = \delta_{ij}$,

$$L(x_i) = \sum_{j=0}^{N} f(x_j) L_j(x_i) = \sum_{j=0}^{N} f(x_i)$$

for each $i = 0, \ldots, N$. In other words, L is the polynomial of degree N which interpolates f at each of the nodes $x_0, \ldots, x_N$.

The expression

$$L(x) = \sum_{j=0}^{N} f(x_j) \prod_{\substack{0 \leq i \leq N \\ i \neq j}} \frac{x - x_i}{x_j - x_i}$$

defining L is called the Lagrange interpolation formula.

The Lagrange Interpolation Formula

Theorem

If $x_0, x_1, \ldots, x_N$ are distinct real numbers, then

$$L(x) = \sum_{j=0}^{N} f(x_j) \prod_{\substack{0 \le i \le N \\ i \ne j}} \frac{x - x_i}{x_j - x_i}$$

is the unique polynomial of degree N such that

$$L(x_i) = f(x_i) \quad \text{for all} \quad i = 0, 1, \ldots, N.$$

Error Estimate for the Lagrange Formula

We have a fairly robust theory for bounding the error in Chebyshev interpolation.

An obvious next step is to develop an error bound for interpolation in the case of more general interpolation nodes.

Once we do that, we can compare different sets of interpolation nodes to Chebyshev nodes and see if we can do better than we have been doing.

Error Estimate for the Lagrange Formula

Theorem

Suppose that $f : [a, b] \to \mathbb{R}$ *is an element of* $C^{N+1}[a, b]$*, that*

$$x_0 < x_1 < \ldots < x_N$$

are points in $[a, b]$*, and that* p *is the unique polynomial of degree* N *which interpolates* f *at the nodes* $x_0, \ldots, x_N$*. Then, for each* $x \in [a, b]$*, there is a point* $\xi_x \in (a, b)$ *such that*

$$f(x) = p(x) + \frac{f^{(N+1)}(\xi_x)}{(N+1)!}(x - x_0)(x - x_1) \cdots (x - x_N).$$

Error Estimate for the Lagrange Formula

This theorem is very similar to Taylor's theorem.

Recall that if f is $C^{(N+1)}[a, b]$ and x_0 in $[a, b]$, then for every $x \in [a, b]$ there exists a point ξ_x such that

$$f(x) = \sum_{n=0}^{N} \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n + \frac{f^{(n+1)}(\xi_x)}{(n+1)!}(x - x_0)^{N+1}$$

The difference between the interpolating polynomial and the Taylor polynomial is that the interpolating polynomial agrees with f at a collection of points while the Taylor polynomial agrees with f and its some of derivatives at a single point.

Error Estimate for the Lagrange Formula

Lemma (Rolle)

If f is differentiable on $[a, b]$ and $f(a) = f(b) = 0$, then there is a point ξ in (a, b) such that

$$f'(\xi) = 0$$

Proof: Either $f(x) = 0$ for all $x \in [a, b]$ or there is a point y such that $f(y) \neq 0$. In the latter case, f must have a local extrema at a point in the interval (a, b). The derivative of f is zero at this point. ∎

Error Estimate for the Lagrange Formula

Lemma (Generalized Rolle Theorem)

If f is a $C^{N+1}[a,b]$ function with $N+2$ distinct roots in the interval $[a,b]$, then there is a point ξ in (a,b) such that

$$f^{(N+1)}(\xi) = 0$$

Proof: Let $x_0 < x_1 < \ldots < x_N < x_{N+1}$ be the distinct roots of f in the interval $[a,b]$. By Rolle's theorem, there exist points

$$x_0^{(1)}, x_1^{(1)}, \ldots, x_N^{(1)}$$

such that

$$x_k < x_k^{(1)} < x_{k+1} \quad \text{for all} \quad k = 0, 1, \ldots, N$$

and

$$f'\left(x_k^{(1)}\right) = 0 \quad \text{for all} \quad k = 0, 1, \ldots, N.$$

Now we apply the Rolle theorem to f' to show that there exist points

$$x_0^{(2)}, x_1^{(2)}, \dots, x_{N-1}^{(2)}$$

such that

$$x_k^{(1)} < x_k^{(2)} < x_{k+1}^{(1)} \quad \text{for all} \quad k = 0, 1, \dots, N-1$$

and

$$f''\left(x_k^{(2)}\right) = 0 \quad \text{for all} \quad k = 0, 1, \dots, N-1.$$

Continuing in this fashion, we find that f''' has at least $N-2$ zeros, f'''' has $N-3$, and so on. Eventually, we find that there is a point ξ in $[a, b]$ such that

$$f^{(N+1)}(\xi) = 0.$$

■

Proof: Let $x \in [a, b]$ and define a function $g : [a, b] \to \mathbb{R}$ via the formula

$$g(t) = f(t) - p(t) - (f(x) - p(x)) \prod_{n=0}^{N} \frac{t - x_n}{x - x_n}.$$

Since f is in $C^{N+1}[a, b]$ and p is infinitely differentiable, $g \in C^{N+1}$. Moreover,

$$g(x_i) = f(x_i) - p(x_i) - (f(x) - p(x)) \prod_{n=0}^{N} \frac{x_i - x_n}{x - x_n} = 0$$

for each $i = 0, \ldots, N$ and

$$\begin{aligned} g(x) &= f(x) - p(x) - (f(x) - p(x)) \prod_{n=0}^{N} \frac{x - x_n}{x - x_n} \\ &= f(x) - p(x) - (f(x) - p(x)) \\ &= 0. \end{aligned}$$

In particular, g is an $C^{N+1}[a, b]$ function with $N + 2$ zeros. By our earlier lemma, there exists a point $\xi_x \in [a, b]$ such that

$$g^{(N+1)}(\xi_x) = 0.$$

It remains to compute the $(N+1)^{st}$ derivative of g. Since p is a polynomial of degree N,

$$p^{(N+1)}(x) = 0.$$

The $(N+1)$ derivative of

$$h(t) = \prod_{n=0}^{N} \frac{t - x_n}{x - x_n}$$

looks tricky to compute. But it is a polynomial of degree $N+1$ — in fact, we can write it as

$$h(t) = c_0 + c_1 t + c_2 t^2 + \cdots + c_{N+1} t^{N+1}$$

with

$$c_{N+1} = \prod_{n=0}^{N} \frac{1}{x - x_n}.$$

It follows that

$$h^{(N+1)}(t) = (N+1)! \prod_{n=0}^{N} \frac{1}{x - x_n}.$$

So by taking the $(N+1)^{st}$ derivatives of both sides of

$$g(t) = f(t) - p(t) - (f(x) - p(x)) \prod_{n=0}^{N} \frac{t - x_n}{x - x_n}.$$

we obtain

$$g^{(N+1)}(t) = f^{(N+1)}(t) - (f(x) - p(x))\,(N+1)! \prod_{n=0}^{N} \frac{1}{x - x_n}.$$

Since $g^{(N+1)}(\xi_x) = 0$,

$$0 = f^{(N+1)}(\xi_x) - (f(x) - p(x))\,(N+1)! \prod_{n=0}^{N} \frac{1}{x - x_n},$$

which implies that

$$\frac{f^{(N+1)}(\xi_x)}{(N+1)!}(x - x_0)(x - x_1)\cdots(x - x_N) = f(x) - p(x).$$

$\blacksquare$

MAT128A: Numerical Analysis
Lecture Sixteen: Equispaced Interpolation Nodes

November 2, 2018

Review

We saw last time that Chebyshev nodes are "good" interpolation nodes.

More explicitly, if $f : [-1, 1] \to \mathbb{R}$ is continuous and

$$p_N(x) = \sum_{n=0}^{N}{}' a_n T_n(x) \quad \text{with} \quad a_n = \frac{2}{\pi} \int_0^{\pi} f(x) T_n(x) \, \frac{dx}{\sqrt{1 - x^2}},$$

then

$$\|f - P_N\|_{\infty} \leq \mathcal{O}\left(\log(N)\right) \|f - P_N^*\|_{\infty},$$

where P_N^* is the minimax polynomial. If f is smooth enough then we have

$$\|f - P_N\|_{\infty} \leq \mathcal{O}\left(1\right) \|f - P_N^*\|_{\infty},$$

and this is the case for most functions of interest.

Review

What's more is that we have simple algorithms for approximating the Chebyshev expansion of a function f. For instance, if

$$\widetilde{P_N}(x) = {\sum_{n=0}^{N}}' \widetilde{a}_n T_n(x) \quad \text{with} \quad \widetilde{a}_n = \frac{2}{N+1} \sum_{j=0}^{N} f\left(\cos\left(\frac{j+\frac{1}{2}}{N}\pi\right)\right) T_n\left(\cos\left(\frac{j+\frac{1}{2}}{N}\pi\right)\right),$$

then

$$\left\| P_N(x) - \widetilde{P_N}(x) \right\|_\infty \le \sum_{n=N+1}^{\infty} |a_n| \, .$$

Of course, $\widetilde{P_N}$ is the unique polynomial of degree N which interpolates f at the nodes

$$\cos\left(\frac{j+\frac{1}{2}}{N}\pi\right) \quad j = 0, 1, \ldots, N.$$

Other choices of Interpolation Nodes

So we have a rigorous statement to the effect that Chebyshev nodes are "good."

But how do they compare to other choices of interpolation nodes? As far as we know at this point, any set of interpolation nodes might be almost as good as Chebyshev nodes.

Perhaps being close to the minimax approximation is a typical property enjoyed by pretty much any collection of interpolation nodes.

Equipspaced Interpolation Nodes

In this lecture, we will see that this is not the case at all.

Indeed, we will see that equispaced interpolation nodes are **bad** interpolation nodes in a rather decisive way. This is somewhat vexing since equispaced nodes are one of the most natural and obvious choices.

Note that this does not mean that one never performs interpolation using equispaced nodes. In fact, interpolation using equispaced nodes is one of the most commonly used techniques in all of numerical analysis — we will discuss why after we see that equispaced nodes are "bad."

Equispaced Interpolation Nodes

Theorem (Runge's Example)

Let $f : [-1, 1]$ be defined by

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

and, for each positive integer N, let Q_N be the polynomial of degree N which interpolates f at the points

$$-1 + \frac{2j}{N}, \quad j = 0, 1, \ldots, N.$$

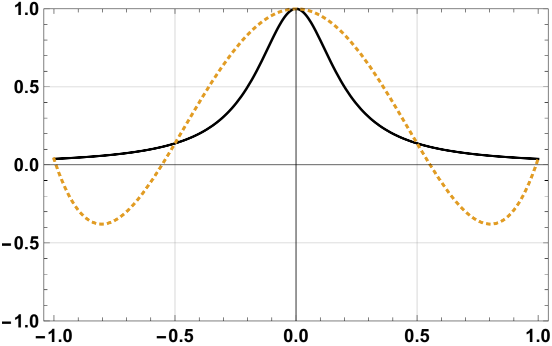
Then

$$\|f - Q_N\|_\infty = \mathcal{O}\left(2^N\right).$$

Here we see that not only does the interpolating polynomial not converge, the difference in the uniform norm grows **exponentially fast** in N (ouch).

Numerical Experiments

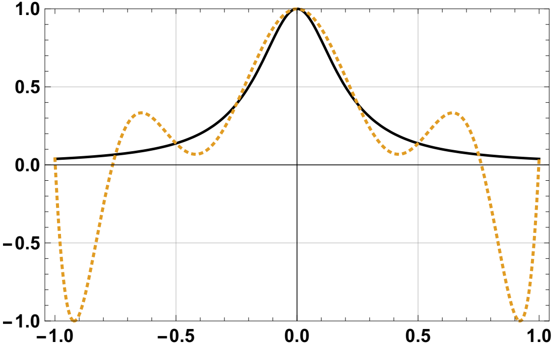
We will not prove this theorem, but we will do some numerical experiments to demonstrate the result. First, let's do some numerical experiment to test the hypothesis.



$$f(x) = \frac{1}{1+25x^2} \quad N = 4$$

Numerical Experiments

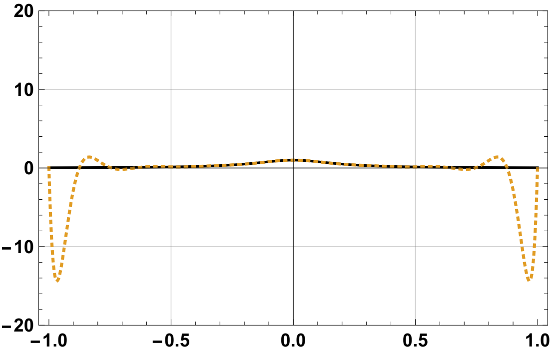
We will not prove this theorem, but we will do some numerical experiments to demonstrate the result. First, let's do some numerical experiment to test the hypothesis.



$$f(x) = \frac{1}{1+25x^2} \quad N = 8$$

Numerical Experiments

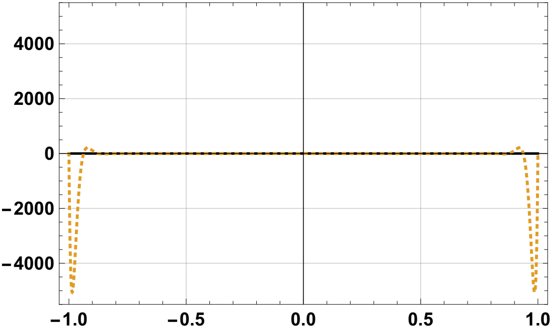
We will not prove this theorem, but we will do some numerical experiments to demonstrate the result. First, let's do some numerical experiment to test the hypothesis.



$$f(x) = \frac{1}{1 + 25x^2} \quad N = 16$$

Numerical Experiments

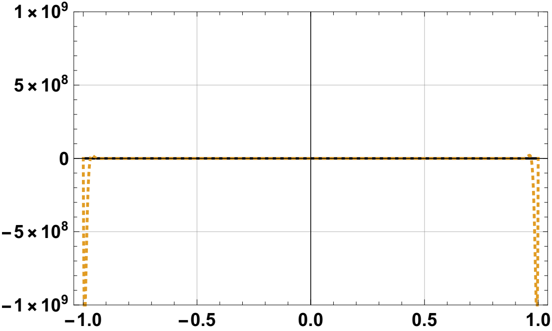
We will not prove this theorem, but we will do some numerical experiments to demonstrate the result. First, let's do some numerical experiment to test the hypothesis.



$$f(x) = \frac{1}{1+25x^2} \quad N = 32$$

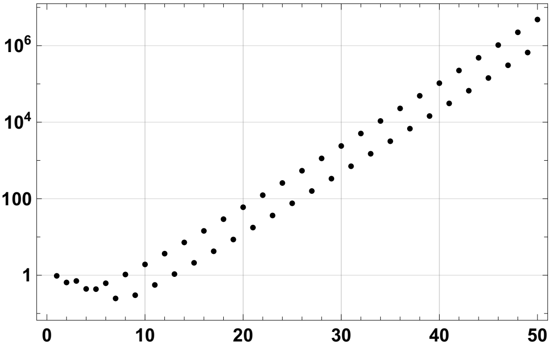
Numerical Experiments

We will not prove this theorem, but we will do some numerical experiments to demonstrate the result. First, let's do some numerical experiment to test the hypothesis.



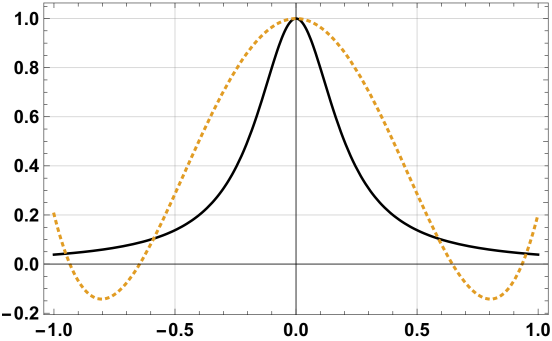
$$f(x) = \frac{1}{1+25x^2} \quad N = 64$$

Numerical Experiments



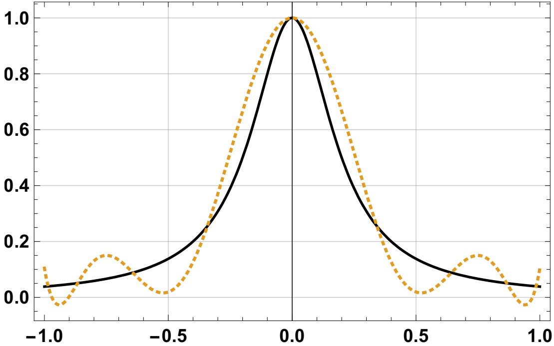
$\|Q_N - f(x)\|_\infty$ as a function of N.

The situation is quite different when we use Chebyshev interpolation nodes.



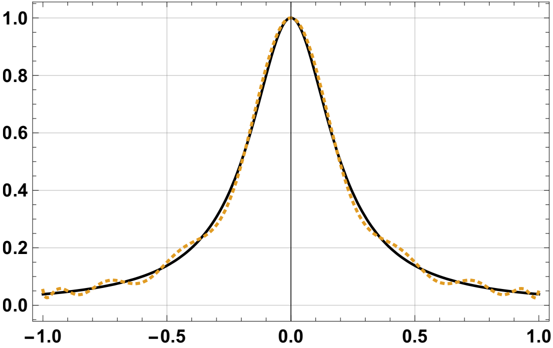
$$f(x) = \frac{1}{1+25x^2} \quad N = 4$$

The situation is quite different when we use Chebyshev interpolation nodes.



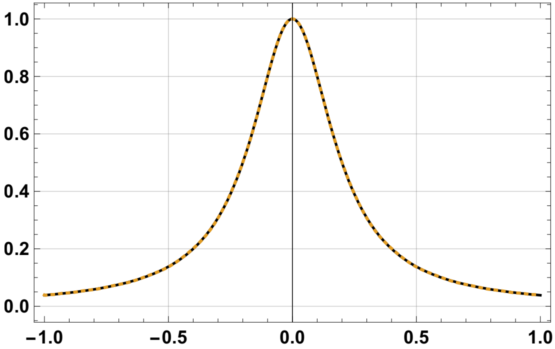
$$f(x) = \frac{1}{1 + 25x^2} \quad N = 8$$

The situation is quite different when we use Chebyshev interpolation nodes.



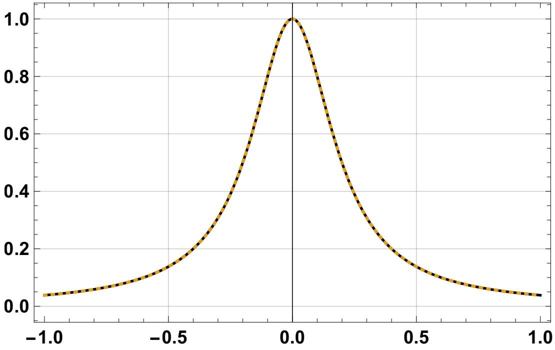
$$f(x) = \frac{1}{1+25x^2} \quad N = 16$$

The situation is quite different when we use Chebyshev interpolation nodes.



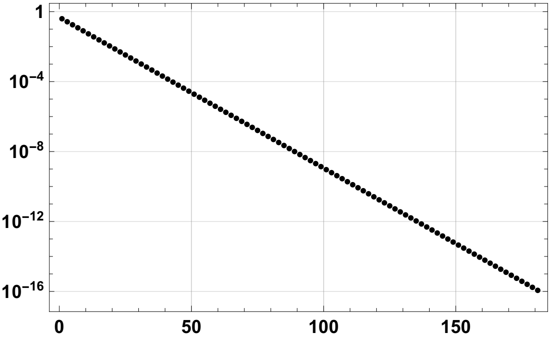
$$f(x) = \frac{1}{1+25x^2} \quad N = 32$$

The situation is quite different when we use Chebyshev interpolation nodes.



$$f(x) = \frac{1}{1+25x^2} \quad N = 64$$

Numerical Experiments



$\|P_N - f(x)\|_\infty$ as a function of N.

Why might use equispaced interpolation nodes despite all this?

We still often use equispaced interpolation.

Equispaced interpolation performs unusually poorly in the case of Runge's example $f(x) = \dfrac{1}{1+25x^3}$. In other cases, the equispaced inteprolation converges, albeit usually not as fast as in the case of Chebyshev interpolation.

To give an example, if we let $f(x) = \cos(x)$ and Q_N be the polynomial of degree N that interpolates f at the nodes

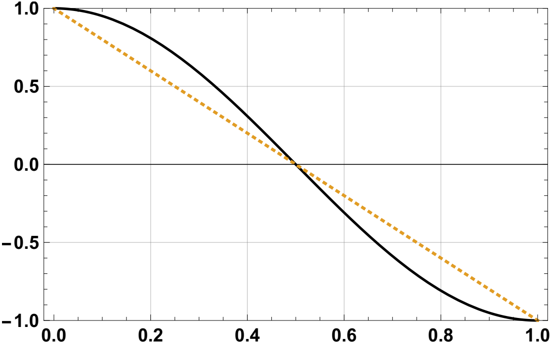
$$x_j = \frac{j}{N} \quad j = 0, 1 \ldots, N,$$

then from the interpolation error formula we proved earlier we see that

$$\begin{aligned} |f(x) - Q_N(x)| &= \left| \frac{f^{N+1}(\xi)}{(N+1)!} \prod_{j=0}^{N} (x - x_j) \right| \\ &\leq \frac{1}{(N+1)!}, \end{aligned}$$

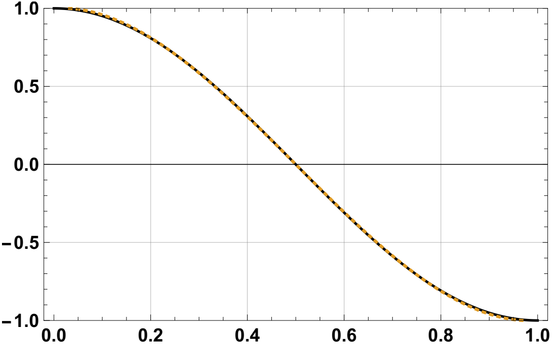
so Q_N does converge to f.

Why might use equispaced interpolation nodes despite all this?



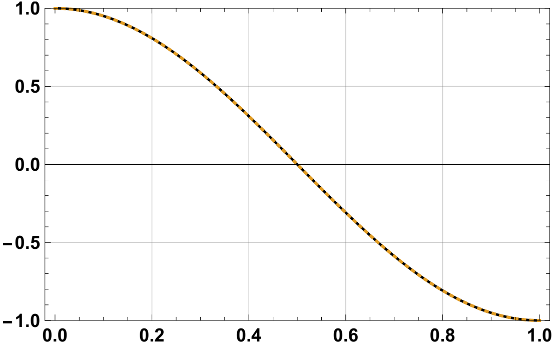
$$f(x) = \cos(x) \quad N = 2$$

Why might use equispaced interpolation nodes despite all this?



$$f(x) = \cos(x) \quad N = 4$$

Why might use equispaced interpolation nodes despite all this?



$$f(x) = \cos(x) \quad N = 8$$

Why might use equispaced interpolation nodes despite all this?

In many cases, we cannot control at what points we know the values of f. For instance, we might get data from an experiment and that experiment might be designed to give values only at equispaced nodes (a very common state of affairs).

We might deliberately place points at equispaced nodes. When using finite differences methods to approximate the derivatives of functions, the formulas are much simpler when equispaced nodes are used. This advantage can outweigh the disadvantages which arise from using equispaced nodes.

Moreover, interpolation from equispaced points is quite effective as long as we stay near the middle of interpolation interval. You will note that in the case of Runge's example the errors occurred near ± 1.

Finally, interpolation at equispaced nodes is usually fine at **low orders.** If we only use a small number of interpolation nodes, then the result errors are roughly the same as if we use Chebyshev nodes. When we do this, we usually use piecewise representations (splines) because we cannot hope to approximate complicated functions using low order polynomials.

MAT128A: Numerical Analysis
Lecture Fifteen: Chebyshev Interpolation, Again

October 31, 2018

Review

We first showed the existence of interpolating polynomials.

Theorem

If $x_0, x_1, \ldots, x_N$ are distinct points on the real line and $f : \mathbb{R} \to \mathbb{R}$, then there is a unique polynomial p of degree N which interpolates f at the points $x_0, \ldots, x_N$.

Recall that p interpolates f at the nodes $x_0, \ldots, x_N$ means that

$$f(x_j) = p(x_j) \quad \text{for all} \quad j = 0, 1, \ldots, N.$$

Review

Next, we show that the truncated Chebyshev expansion for f interpolates f at the points of the Chebyshev grid.

Theorem

Suppose that $f : [-1, 1] \to \mathbb{R}$ is a continuous function, $x_0, x_1, \ldots, x_N$ are defined by

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

and $a_0, a_1, \ldots, a_N$ are given by the formula

$$a_n = \frac{2}{N+1}\sum_{j=0}^{N} f\left(x_j\right) T_n\left(x_j\right).$$

Then

$$\sum_{n=0}^{N}{}' a_n T_n(x)$$

is the unique polynomial of degree N which interpolates f at the points

$$x_0, x_1, \ldots, x_N.$$

Review

We then developed a constructive formula for the polynomial interpolating a function at any given set of nodes.

Theorem

If $x_0, x_1, \ldots, x_N$ are distinct real numbers, then

$$L(x) = \sum_{j=0}^{N} f(x_j) \prod_{\substack{0 \le i \le N \\ i \neq j}} \frac{x - x_i}{x_j - x_i}$$

is the unique polynomial of degree N such that

$$L(x_i) = f(x_i) \quad \text{for all} \quad i = 0, 1, \ldots, N.$$

Review

Finally, we developed an expression for interpolation error.

Theorem

Suppose that $f : [a, b] \to \mathbb{R}$ is an element of $C^{N+1}[a, b]$, that

$$x_0 < x_1 < \ldots < x_N$$

are points in $[a, b]$, and that p is the unique polynomial of degree N which interpolates f at the nodes $x_0, \ldots, x_N$. Then, for each $x \in [a, b]$, there is a point $\xi_x \in (a, b)$ such that

$$f(x) = p(x) + \frac{f^{(N+1)}(\xi_x)}{(N+1)!}(x - x_0)(x - x_1)\cdots(x - x_N).$$

Good Interpolation Nodes

Now we will investigate the question of what interpolations nodes should be chosen.

An obvious stragegy is to try to minimize the magnitude of the error term

$$\frac{f^{(N+1)}(\xi_x)}{(N+1)!}(x-x_0)(x-x_1)\cdots(x-x_N).$$

We cannot hope to control the magnitude of the $(N+1)^{st}$ derivative of f if we want to choose nodes which do not depend on what function we are interpolating, but we can choose nodes

$$x_0, x_1, \ldots, x_N$$

which minimize the magnitude of

$$(x-x_0)(x-x_1)\cdots(x-x_N).$$

Good Interpolation Nodes

Before we state the next theorem about "good interpolation node," let's recall a few facts.

We say that a polynomial is monic if its leading coefficient is 1.

The uniform norm of a function $f : [-1, 1] \to \mathbb{R}$ is

$$\sup_{-1 \leq x \leq 1} |f(x)| .$$

We denote it by $\|f\|_\infty$.

We also recall that the leading coefficient of T_{n+1} is 2^n (this follows by induction and the recurrence relations).

Good Interpolation Nodes

Theorem

For each $j = 0, 1, \ldots, N$, *let*

$$x_j = \cos\left(\frac{j + \frac{1}{2}}{N+1}\pi\right).$$

Then

$$(x - x_0)(x - x_1)\cdots(x - x_N) = \frac{1}{2^N} T_{N+1}(x)$$

is the monic polynomial of degree $N + 1$ *with the smallest possible uniform norm, and that norm is* 2^{-N}.

Proof:

First of all, let's make sure we understand why

$$(x - x_0)(x - x_1)\cdots(x - x_N) = \frac{1}{2^N}T_{N+1}(x).$$

We know that T_{N+1} is a polynomial of degree $N + 1$, and the formula

$$T_{N+1}(x) = \cos((N+1)\arccos(x))$$

implies that its roots are

$$\cos\left(\frac{j+\frac{1}{2}}{N+1}\pi\right) \quad j = 0, 1, \ldots, N+1$$

since the zeros of cosine are

$$\frac{\pi}{2} + k\pi \quad k \in \mathbb{Z}.$$

Since T_{N+1} and $(x - x_0)(x - x_1)\cdots(x - x_N)$ have the same roots, there must be a constant C such that

$$T_{N+1}(x) = C(x - x_0)(x - x_1)\cdots(x - x_N).$$

That the correct constant C is 2^{-N} then follows from the fact that the leading coefficient (i.e., the coefficient of x^{N+1}) of

$$(x - x_0)(x - x_1)\cdots(x - x_N)$$

is 1 while the leading coefficient of T_{N+1} is 2^N.

So

$$T_{N+1}(x) = 2^N(x - x_0)(x - x_1)\cdots(x - x_N).$$

We will now show that $2^{-N}T_{N+1}$ is the monic polynomial of degree $N + 1$ with the smallest uniform norm. That it is a monic polynomial means that its leading coefficient is 1.

Suppose that p is a monic polynomial of degree $N+1$ such that

$$|p(x)| < 2^{-N}$$

for all $x \in [-1, 1]$. For each $j = 0, 1, \ldots, N, N+1$, let

$$y_j = \cos\left(\frac{\pi}{N+1} j\right).$$

These are the minima and maxima of the Chebysev polynomial T_{N+1} and the value of

$$T_{N+1}\left(\cos\left(\frac{\pi}{N+1} j\right)\right)$$

alternatives between 1 and -1. It follows that

$$\begin{aligned}
p(y_0) &< 2^{-N} T_{N+1}(y_0) \\
p(y_1) &> 2^{-N} T_{N+1}(y_1) \\
p(y_2) &< 2^{-N} T_{N+1}(y_2) \\
&\vdots
\end{aligned}$$

We let

$$q(x) = p(x) - 2^{-N} T_{N+1}(x).$$

Then q alternates signs between the points $y_0, y_1, \ldots, y_N, y_{N+1}$, so it has at least $N+1$ zeros. But q is a polynomial of degree at most N since the leading term in p and $2^{-N} T_{N+1}$ cancel. It follows that q must be identically zero (the only way a polynomial of degree less than or equal to N can have $N+1$ zeros is if it is identically zero). In other words, we must have

$$p(x) = 2^{-N} T_{N+1}(x).$$

But this contradicts our assumption that

$$|p(x)| < 2^{-N},$$

since $2^{-N} T_{nN1}(x)$ assumes the value 2^{-N}. We conclude that there can be no monic polynomial p such that

$$|p(x)| < 2^{-N}$$

for all $x \in [-1, 1]$. ■

Good Interpolation Nodes

We conclude this theorem that Chebyshev nodes are reasonably good interpolation nodes.

Note, though, that this does not mean that the polynomial

$$p(x) = {\sum_{n=0}^{N}}' a_n T_n(x), \quad a_n = \frac{2}{N+1} \sum_{j=0}^{N} f\left(\cos\left(\frac{j+\frac{1}{2}}{N+1}\right)\right) T_n\left(\cos\left(\frac{j+\frac{1}{2}}{N+1}\right)\right)$$

minimizes the error

$$\{\|f-q\|_\infty : \ q \text{ is a polynomial of degree } N+1\},$$

only that it minimizes a factor which appears in one particular expression for the error in the Lagrange formula.

Minimax Approximations

Theorem

Suppose that $f : [-1, 1] \to \mathbb{R}$ is a continuous function. There is a unique polynomial p_N^* of degree N such that

$$\|f - p_N^*\|_\infty = \min \{\|f - q\|_\infty : \ q \text{ is a polynomial of degree } N\},$$

where $\|\cdot\|_\infty$ is the uniform norm on $[-1, 1]$. We call p_N^* the minimax polynomial of degree N for the function f.

The polynomial p_N^* is called the minimax polynomial because

$$\|f - p_N^*\|_\infty = \min_{q \in \mathbb{P}^n} \ \max_{x \in [-1,1]} \ |f(x) - q(x)|,$$

where $\mathbb{P}^n$ denotes the vector spaces of polynomials of degree less than or equal to N.

Minimax Approximations vs Chebyshev Approxmations

Computing the minimax polynomials is computationally difficult, and there is very little profit in it, as the next theorem demonstrates.

Theorem

Suppose that $f : [-1,1] \to \mathbb{R}$ is a continuous function, that p_N^* is the minimax polynomial of degree N for f, that $\{a_n\}$ are the Chebyshev coefficients of f — that is,

$$a_n = \frac{2}{\pi}\int_0^{\pi} f(x)T_n(x)\ \frac{dx}{\sqrt{1-x^2}}$$

— and that

$$p_N(x) = {\sum_{n=0}^{N}}' a_n T_n(x).$$

Then

$$\|f - p_N\|_\infty \le \left(4 + \frac{4}{\pi^2}\log(N)\right)\|f - p_N^*\|_\infty$$

and

$$\frac{\pi}{4}|a_{N+1}| \le \|f - p_N^*\|_\infty .$$

Minimax Approximations

We will not prove the preceding theorem, but we will discuss some of its implications.

We note first that the theorem bounds the error in the approximation of f by the truncated Chebyshev expansion with **exact** coefficients. This is not a serious difficulty, though, because we know that if

$$\widetilde{P_N}(x) = {\sum_{n=0}^{N}}' \tilde{a}_n T_n(x) \quad \text{with} \quad \tilde{a}_n = \frac{2}{N+1} \sum_{j=0}^{N} f\left(\cos\left(\frac{j+\frac{1}{2}}{N}\pi\right)\right) T_n\left(\cos\left(\frac{j+\frac{1}{2}}{N}\pi\right)\right),$$

then

$$\left\| P_N(x) - \widetilde{P_N}(x) \right\|_\infty \leq \sum_{n=N+1}^{\infty} |a_n| .$$

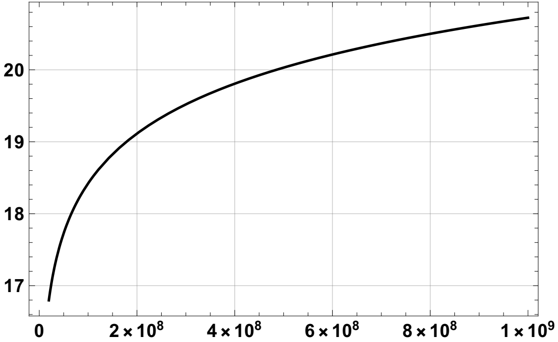
This means that if the Chebyshev coefficients of f decay rapidly, then

$$P_N(x) \approx \widetilde{P_N}(x)$$

once N is of moderate size.

Minimax Approximations vs Chebyshev Approxmations

The logarithm is a very slowly growing function:



This means that unless N is very large, the inequality

$$\|f - p_N\|_\infty \leq \left(4 + \frac{4}{\pi^2}\log(N)\right)\|f - p_N^*\|_\infty$$

shows that the minimax approximation of the continuous function f is not that much better than the Chebyshev approximation.

Minimax Approximations vs Chebyshev Approxmations

Moreover, the second bound

$$\frac{\pi}{4}\left|a_{N+1}\right| \leq \left\|f - p_N^*\right\|_\infty$$

is useful for showing that if the Chebyshev coefficients of a function decay rapidly, then the minimax approximation is not much better than the Chebyshev approximation.

For instance, suppose that $|a_n| \leq r^{-n}$. Then

$$\left\|f - p_N\right\|_\infty \leq \sum_{n=N+1}^{\infty} |a_n| \leq \sum_{n=N+1}^{\infty} r^{-n} = \frac{r^{-N}}{1-r} = \frac{r}{1-r}\left|a_{N+1}\right| \leq \frac{4r}{\pi(1-r)}\left\|f - p_N^*\right\|_\infty .$$

This bound can be improved, but in this form it already shows that if f is analytic then accuracy of the Chebyshev approximation of f is within a constant factor of the accuracy of the minimax approximation.

The same can be shown to be true if f is C^k, although doing so requires a much more involved argument.