$\dfrac{\sin x}{x}$ と円周率

2021 年 5 月 27 日

1　はじめに

$\dfrac{\sin x}{x}$ の不等式を用いて、2 つの手法で円周率を数値計算しました. 必要な知識は高校数学のみです. 高校数学だけでもかなり高精度に円周率 $\pi = 3.14159265\ldots$ を求めることができました.

2　半角公式

半角公式として、$0 \leq x \leq \pi$ のとき、

$$\cos\frac{x}{2} = \sqrt{\frac{1+\cos x}{2}}, \quad \sin\frac{x}{2} = \sqrt{\frac{1-\cos x}{2}}$$

が知られている. $\left(\cos\dfrac{\pi}{2}, \sin\dfrac{\pi}{2}\right) = (1, 0)$ から初めて、半角公式を次々に適用していくことにより、$\left(\cos\dfrac{\pi}{4}, \sin\dfrac{\pi}{4}\right)$, $\left(\cos\dfrac{\pi}{8}, \sin\dfrac{\pi}{8}\right)$, $\left(\cos\dfrac{\pi}{16}, \sin\dfrac{\pi}{16}\right)$, $\left(\cos\dfrac{\pi}{32}, \sin\dfrac{\pi}{32}\right)$, $\left(\cos\dfrac{\pi}{64}, \sin\dfrac{\pi}{64}\right), \ldots,$ を数値計算できる.

3　手法 1

Nelsen (2015) の Cameo 2 を参考にする. $\dfrac{\sin x}{x}$ の不等式として、$0 < x < \dfrac{\pi}{2}$ のとき

$$\cos x < \frac{\sin x}{x} < 1$$

が知られている．n を 3 以上の自然数として、上式の x に $\dfrac{\pi}{n}$ を代入して整理する．すると、円周率 π に関する不等式

$$n\sin\frac{\pi}{n} < \pi < n\frac{\sin\dfrac{\pi}{n}}{\cos\dfrac{\pi}{n}}, \quad (n \geq 3)$$

が得られる．これにより, 円周率 π の下限及び上限を数値計算できる．

4　手法 2

Nelsen (2020) を参考にする．$\dfrac{\sin x}{x}$ の不等式として、$0 < x < \dfrac{\pi}{2}$ のとき

$$\frac{3\cos x}{2\cos x + 1} < \frac{\sin x}{x} < \frac{\cos x + 2}{3}$$

が知られている．左側は Huygens の不等式, 右側は Cusa の不等式とそれぞれ呼ばれている．n を 3 以上の自然数として、上式の x に $\dfrac{\pi}{n}$ を代入して整理する．すると、円周率 π に関する不等式

$$\frac{3}{2+\cos\dfrac{\pi}{n}}n\sin\frac{\pi}{n} < \pi < \frac{1+2\cos\dfrac{\pi}{n}}{3}n\frac{\sin\dfrac{\pi}{n}}{\cos\dfrac{\pi}{n}}, \quad (n \geq 3)$$

が得られる．これにより、手法 1 より高精度で、円周率 π の下限及び上限を数値計算できる．

References

Nelsen, R. B. (2015). *Cameos for Calculus: Visualization in the First-Year Course*. Mathematical Association of America.

Nelsen, R. B. (2020). Elementary Proofs of the Trigonometric Inequalities of Huygens, Cusa, and Wilker. *Mathematics Magazine*, *93*(4), 276–283.

1

$\dfrac{\cos x + 2}{3}$

$\dfrac{\sin x}{x}$

$\cos x$

$\dfrac{3\cos x}{2\cos x + 1}$

0

$\pi/2$

```
# method_1_lb.py
# calculate the lower bound of the number pi
# [0] init
from decimal import *
getcontext().prec = 200
N = 30

# [1] calculate sequences c[n], s[n], and a[n]
# [1.1] calculate sequences c[n] and s[n]
c = {}
s = {}
c[1] = Decimal(0)
s[1] = Decimal(1)
for n in range(1, N):
	c[n+1] = ((1 + c[n]) / 2).sqrt()
	s[n+1] = ((1 - c[n]) / 2).sqrt()
# [1.2] calculate sequence a[n]
a = {}
for n in range(2, N+1):
	a[n] = pow(2, n) * s[n]

# [2] print a[n] as decimal
print("n, a[n]")
for n in range(2, N+1):
	print(f"{n:0>2d},", str(a[n])[:2 + 100])
```

method_1_lb.py の出力, 手法 1 の下限値
n, a[n]
02, 2.828427124746190097603377448419396157139343750753896146353359475981464956924214077700775068655283145 4
03, 3.061467458920718173827679872243190934090756499885016331470405085020368271680717537896110282738268377 1
04, 3.121445152258052285572557895632355854843065884031276924072033431916213100574793283605204947745446107 6
05, 3.136548490545939263814258044436539067556373541360018152232479353900393689250801177827554468682193905 7
06, 3.140331156954752912317118524331690132143703233648186893447843409226620132093488436622129095155246320 2
07, 3.141277250932772868062019770788214408379663262649789129824867044927097361586525961491274380225786743 8
08, 3.141513801144301076328515059456822307935313815492928009789868351167455172565255655412623785954502488 5
09, 3.141572940367091384135800110270761429533637794504360630647019674089791720688888220569832031359651657 1
10, 3.141587725277159700628854262701918739399280858574843286678421400551216704533146781507679194480102278 5
11, 3.141591421511199973997971763740833955747562650086180797675232600350217806532080723667742169640626417 1
12, 3.141592345570117742340375994157369930305206075651203808028308307852456612413801155370068241059485625 8
13, 3.141592576584872665681606922378753097320532784311379811651125229630854452864540975848340689011521664
14, 3.141592634338562989095478263627791293954032170748458571775908236117328314831705960134103153190897456 5
15, 3.141592648776985669485107969277177075697766001906320623752582406859399734989988925411636662601435043 2
16, 3.141592652386591345803525521057963884338655244174421082962290909162730041410422520639892770382739288 5
17, 3.141592653288992765271943042173740003460576037525655749897579213503043643974248760580115125455219329 4
18, 3.141592653514593120163348243281080432651659519822842849171815078089303231331893226340633026276554882 0
19, 3.141592653570993208887718344859772114691517515350723793706505103886063817320309766186357881164147616 9
20, 3.141592653585093231068905795335812349244408194233282667882629740224384271720677493980377159976979277 3
21, 3.141592653588618236614208590772407884980962902467228863671004376712449175998964487575221300804217847 8
22, 3.141592653589499488000534660432655861545788724937530667929875770082608329607396495795635986834492080 5
23, 3.141592653589719800847116201022786548981288773909045956879528662786767369632919114955434450950774467 4
24, 3.141592653589774879058761587618761014171133317717260112577063537496052278483365155551760969179283304 1
25, 3.141592653589788648611672934358282242551781239735595767592040325621464774346040468842170873511511270 4
26, 3.141592653589792090999900771048820525402142412968438698878924313857426612078015379648205793686197307 2
27, 3.141592653589792951596957730221808719599432656112776813816427701207176655405609997272687140686373060 7
28, 3.141592653589793166746221970015077869616548963768160103831789342661455267523494233234110068818071225 0
29, 3.141592653589793220533538029963396538462565149861408048914109324497551701370298571151806303466822889 5
30, 3.141592653589793233980367044950476292007927765708433776439569573787845066592588256843702635324571440 2

```
# method_1_ub.py
# calculate the upper bound of the number pi
# [0] init
from decimal import *
getcontext().prec = 200
N = 30

# [1] calculate sequences c[n], s[n], and a[n]
# [1.1] calculate sequences c[n] and s[n]
c = {}
s = {}
c[1] = Decimal(0)
s[1] = Decimal(1)
for n in range(1, N):
	c[n+1] = ((1 + c[n]) / 2).sqrt()
	s[n+1] = ((1 - c[n]) / 2).sqrt()
# [1.2] calculate sequence a[n]
a = {}
for n in range(2, N+1):
	a[n] = pow(2, n) * s[n] / c[n]

# [2] print a[n] as decimal
print("n, a[n]")
for n in range(2, N+1):
	print(f"{n:0>2d},", str(a[n])[:2 + 100])
```

method_1_ub.py の出力, 手法 1 の上限値
n, a[n]
02, 4.0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
03, 3.31370849898476039041350979367758462855737500301558458541343790392585982769685631080310027462113258 18
04, 3.18259787807452811058556196231481965756560802114545571073787193320687367136443300885311849692183542 01
05, 3.15172490742925609847032068132247783337767842237448627138931587630142203938601511220752685377195520 43
06, 3.14411838524590426274197256136407149310052809935325995164134263491397539086024523553864465092473642 94
07, 3.14222362994245684538620850699631631263503343576830556356920592479584069167870219253085315318693721 23
08, 3.14175036916896645910721362797332388998853178422358338355352638256957032281570590718310464321543679 39
09, 3.14163208070318180571871518787114766909716102098541353220131003071278247136368947426004798844850067 16
10, 3.14160251025680894676368965849266001179227345810259658340932984544272226854166180135434177232117614 87
11, 3.14159511774958905035309223598169594543611109596205096415196831049024971661654698767389059454346072 09
12, 3.14159326962930731078945878682797576435643237997117774039135658976251918006907832060915330663420369 25
13, 3.14159280759964457652825443596058346351174698641210637934021446094084286879343215933609236765560377 50
14, 3.14159269209225437422841955718088385878551822004344910793288645791027741268981071867259329718271976 61
15, 3.14159266321540841623217919009007384043462538580697503815928899202461198690941206083605870100895666 32
16, 3.14159265599619702626928316277128760036621901717952022435309009031966586856583019247074875900201236 27
17, 3.14159265419139418499956931883584370262528997944885539439332439376579385198301568565087491974865067 80
18, 3.14159265374019347507095399160890296096132398947375825401875312626203351629566653722283690727568841 14
19, 3.14159265362739329761310098063972575019763441334695762409122070718289881321718729410084379775527721 11
20, 3.14159265359919325325015652919943117186192988134062325939404585727611551794598931749375045241542939 96
21, 3.14159265359214324215951534142072707803363501484021197507470407818629816571815651863223250861900002 27
22, 3.14159265359038073938686097729363656662352935338754130218009541292578300853503434995852766503682774 95
23, 3.14159265358994011369369775706296303194776252119524770463886073070155534795196734095762263781964142 02
24, 3.14159265358982995727040697518036334158164678834109459328788066659940215454660411098204795371735098 57
25, 3.14159265358980241816458428115815521232122070515914132452258855823489762058539569770404546937444341 45
26, 3.14159265358979553338812860774313079208930305353097027452939013186622015059836818594922028625092105 18
27, 3.14159265358979381219401468939503266278652286590064020379999678124621728756978063881696338618063536 89
28, 3.14159265358979338189548620980836175394552776933775456689336355709940438665746969331690452083364710 00
29, 3.14159265358979327432085408991171612820307274207427843127050249007436705984145335594605760632404139 22
30, 3.14159265358979324742719605993755610310919609443793565994049822359209486703885202522833268818479675 43

```
# method_2_lb.py
# calculate the lower bound of the number pi
# [0] init
from decimal import *
getcontext().prec = 200
N = 30

# [1] calculate sequences c[n], s[n], and a[n]
# [1.1] calculate sequences c[n] and s[n]
c = {}
s = {}
c[1] = Decimal(0)
s[1] = Decimal(1)
for n in range(1, N):
	c[n+1] = ((1 + c[n]) / 2).sqrt()
	s[n+1] = ((1 - c[n]) / 2).sqrt()
# [1.2] calculate sequence a[n]
a = {}
for n in range(2, N+1):
	a[n] = 3 / (2 + c[n]) * pow(2, n) * s[n]

# [2] print a[n] as decimal
print("n, a[n]")
for n in range(2, N+1):
	print(f"{n:0>2d},", str(a[n])[:2 + 100])
```

method_2_lb.py の出力, 手法 2 の下限値
n, a[n]
02, 3.134446499564897310177218483004679126524589287006679108034330530253939926155795561772757260551913963603, 3.141169899319954592852989078193816330447475618573659279203612159625420151759420410531471464801640864404, 3.141566592648827609818335164631976565279496670000824435606486256405987867589715753446867295344017414205, 3.14159103037775376361482522929345052022281158007478692700834435132663082995182364396911387289402548356, 3.1415925522263065200790334227087821456405685319515575044143987150030106322998610788436633688636149381
07, 3.14159264725593812774633538358539701223385651092579678841307848007797866200173024513501728840788891568, 3.141592653193948585130934530257257706046787217251820748304512653510669440593202764094183233582066747009, 3.14159265356505328029170368706745161032760938935839051838142370208163894639077870099148697861268902470, 3.1415926535882469962747651583192440520438556316382130558091459333870628936772265544818833808692977581
11, 3.14159265358969659840711653803218313978211802214634423774000298522303275501416552287052245793108074101, 3.14159265358978719846044194767041791533029665587209864650191742145890549703066232725938980009513459470, 3.14159265358979286096252562155480339065367727061972767228655151518407401148117739657707092399530435890, 3.14159265358979321486888633298421276624900464965071349395293692222839502017517987175404246877511933490
15, 3.14159265358979323698803357247686758164782389967279636421633971452107916986227792452378942701934879500, 3.141592653589793238370480270179975995289225346490818148525845342120742048581052989734548024516589806400, 3.14159265358979323845688318871196429453828803268309657799957672093017564311698430348206796584057452960
18, 3.14159265358979323846228337111905018860751613608059751960204116206767273373273470157824736126541093560, 3.14159265358979323846262088251947487925819148044989877549741863519289053617432598023058599768938216540
20, 3.14159265358979323846264197698200113839684852018859997103764110706113107611722816691191512027702585790
21, 3.14159265358979323846264329538590902515509255131247212827904946742792817325513330198743855219939132630
22, 3.14159265358979323846264337778615326800814027147879259553702088490715460037405375930686651348013441330
23, 3.14159265358979323846264338293616853318537227693014251384252413658309571248293940676448890204346203080
24, 3.14159265358979323846264338325804448725893234794180430598119641188277260470085187352788124996886579900
25, 3.14159265358979323846264338327816173438852958785926679959564731477163767837996384974205672742505448930
26, 3.14159265358979323846264338327941906233412941122097123094042294890582781464082865584912056232699208340
27, 3.14159265358979323846264338327949764533072940011649749267281330842321621835145796579534289642375472290
28, 3.14159265358979323846264338327950255676801689942145881738692117329479718611224375552422413087492159580
29, 3.14159265358979323846264338327950286373284736812800313351523781277933533620233708997842669115050902610
30, 3.14159265358979323846264338327950288291814927242216190691909642927728369497756541360243381022821311850

```
# method_2_ub.py
# calculate the upper bound of the number pi
# [0] init
from decimal import *
getcontext().prec = 200
N = 30

# [1] calculate sequences c[n], s[n], and a[n]
# [1.1] calculate sequences c[n] and s[n]
c = {}
s = {}
c[1] = Decimal(0)
s[1] = Decimal(1)
for n in range(1, N):
	c[n+1] = ((1 + c[n]) / 2).sqrt()
	s[n+1] = ((1 - c[n]) / 2).sqrt()
# [1.2] calculate sequence a[n]
a = {}
for n in range(2, N+1):
	a[n] = (1 + 2 * c[n]) / 3 * pow(2, n) * s[n] / c[n]

# [2] print a[n] as decimal
print("n, a[n]")
for n in range(2, N+1):
	print(f"{n:0>2d},", str(a[n])[:2 + 100])
```

method_2_ub.py の出力, 手法 2 の上限値
n, a[n]
02, 3.2189514164974600650689182989462641047595625005025974309022396506543099712828093851338500457701887636
03, 3.1455478056087322460229565127213221655796293342618724161180826913221987903527637955317736133658897787
04, 3.1418293941968775605768925845265104557505799297360031862939795990130999575046731920211761308042425451
05, 3.1416072961737115420329455900651853228301418350315075252847581947007364726292058226208785970454476719
06, 3.1415935663851366957920698700091505857959781888832112461790098177890718850157407029276342804117430233
07, 3.1415927106026675271700826828575817097981199870226279410729800048833451382839180385044673045461702333
08, 3.1415926571525228705880812489623228352863864717364798010444210283014935559820724060027840717081472570
09, 3.1415926538124548579967718028042235093881455366647115978317831262974553042471553051332373503892679953
10, 3.1415926536037094493404660612988324968636117250840943855887242155150518925359851214565667204271269019
11, 3.1415926535906629994496785878211212856437454653781375198341445037302284432269028116697916446082378517
12, 3.1415926535898475984900702583809052083222815104245284521493244018224774682988935437830965962510583147
13, 3.1415926535897966359638222068121113609919511810914607805568131689556712531221134515019201684859693693
14, 3.1415926535897934508064586948121554822311941871801220838282343100483113474510742129802665345215048931
15, 3.1415926535897932517341317095481426639433857965398720952214846019144704856297966372197773420706089166
16, 3.1415926535897932392921114016290717896811765018427874634258906362150419837955584112501780999224969799
17, 3.1415926535897932385144851343944412365154806848333889647294942735906270466438377356037017235530297789
18, 3.1415926535897932384658834927236879420882143430398146507874610941468799929864843299680343199429327252
19, 3.1415926535897932384628458901197566598602231480161350705014103049850088159526022754911865200278574817
20, 3.1415926535897932384626560399570186234502487566023961983864351125749613537957814351515015907897959848
21, 3.1415926535897932384626441743218476159985202732582232341389042772037321725720284979275583700758119061
22, 3.1415926535897932384626434327196494299050356010875342126799489843636665559166091138499332129019373035
23, 3.1415926535897932384626433863695120433034466896711132061326393520916966957392685236228305132403967850
24, 3.1415926535897932384626433834726284566413044745918716061473359138638355705044448073618566306919725313
25, 3.1415926535897932384626433832915732324749277282101109532355564031592757230924922117961290721324886518
26, 3.1415926535897932384626433832802572809645292931559492240957462531936911249181329817485439578744385554
27, 3.1415926535897932384626433832795500339951293927087312771442840612201901994603335444541125558511271634
28, 3.1415926535897932384626433832795058310595418989580249548523140808074383072348193865950415528232631834
29, 3.1415926535897932384626433832795030683760676805990315096995737130231568208606834994165567377525623904
30, 3.1415926535897932384626433832795028957083505419516010709398791237225950000746761796385793196113132116

出力まとめ
各手法の数列の第 30 項をまとめておく.
3.14159265358979323398036704495047629200792776570843377643956957378784506659258825684370263532457144 02, 手法 1 の下限値
3.14159265358979324742719605993755610310919609443793565994049822359209486703885202522833268818479675 43, 手法 1 の上限値
3.14159265358979323846264338327950288291814927242216190691909642927728369497756541360243381022821311 85, 手法 2 の下限値
3.14159265358979323846264338327950289570835054195160107093987912372259500007467617963857931961131321 16, 手法 2 の上限値