円周率の公式と計算法

大浦拓哉

1　はじめに

この講座では、円周率をいかにして高速かつ高精度で計算するかということに主眼をおきます。まず、１・２日目は、円周率計算の基本となる公式について数学的な説明や歴史的話題などについて触れる予定です。次に、３・４日目では、コンピュータを用いた高速かつ高精度な計算技法について紹介したいと思います。とくにコンピュータを用いた計算で使われている数学的手法に関しては、一般的にあまり知られていないと思われるので、その点に重点をおいてお話したいと思います。また、予稿にあげる以外の公式や算法も多々ありますので、時間の許す限り講義の中で説明したいと思っています。

2　円周率計算の基本となる公式

2.1　正多角形による方法

円周率の古くからの計算法は正多角形で円を近似する方法です。$a_0 = 2\sqrt{3}$, $b_0 = 3$ として

$$\begin{aligned} a_{n+1} &= \frac{2a_n b_n}{a_n + b_n}, \\ b_{n+1} &= \sqrt{a_{n+1} b_n} \end{aligned}$$

とすれば、a_n、b_n はそれぞれ直径 1 の円に外接、内接する正 $6 \cdot 2^n$ 角形の長さになります。したがって、$b_n < \pi < a_n$ です。これらの式は、初等的な幾何学で容易に証明でき、紀元前 3 世紀頃アルキメデスは $n = 4$ を評価して $3 + 10/71 < \pi < 3 + 1/7$ という関係を導いたとされています。また、1600 年頃オランダのルドルフが一生かけて 35 桁計算したのも正多角形による方法です。

ヴィエトによる π を表す最初の公式

$$\pi = 2 \cdot \frac{2}{\sqrt{2}} \cdot \frac{2}{\sqrt{2+\sqrt{2}}} \cdot \frac{2}{\sqrt{2+\sqrt{2+\sqrt{2}}}} \cdots$$

も、導出は正多角形によるもので、最初の k 項までの積が半径 1 の円に内接する正 2^{k+1} 角形の面積になっています。

これらの正多角形による方法は計算効率が悪く、十進 N 桁を得るためには約 $1.66N$ 回の乗除算と平方根の計算が必要になり、計算する桁数が増えるにつれて莫大な計算時間がかかります。たとえば、この方法で 10 億桁計算する場合、10 億桁の精度での約 17 億回の平方根と乗除算の計算が必要になります。この場合の計算は、のちに話す算術幾何平均の方法と比較して数千万倍の時間がかかり、現代の高速なコンピュータを用いても天文学的な時間のかかるものです。

2.2 級数による方法

もっともよく知られた級数による方法のひとつは、アークタンジェントのテイラー展開によるものです。グレゴリー級数として知られる

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots \tag{1}$$

は、テイラー展開

$$\tan^{-1} x = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \frac{x^9}{9} - \cdots \tag{2}$$

に、$x = 1$ を代入して得られます。この公式は 1670 年代にライプニッツも発見しましたが、インドのマーダヴァ学派のほうが先で 1400 年ごろに発見していたとされています。グレゴリー級数は、そのままでは収束が極めて遅く、数値計算にはまったく向いてはおりません。たとえば、10 桁の値を得るためには約 100 億項もの計算が必要になるのです。しかし、次に解説するオイラー変換という収束の加速法を用いることで、これは大幅に改善されます。

2.2.1 オイラー変換

一般にオイラー変換とは、級数

$$S = \sum_{n=0}^{\infty} (-1)^n a_n \tag{3}$$

の収束を改善するための方法です。導出方法は、まず、$Ea_n = a_{n+1}$ で定義されるシフト演算 E を用いて、(3) を

$$S = \sum_{n=0}^{\infty} (-E)^n a_0 = \frac{1}{1+E} a_0$$

と形式的に書き換えます。次に、$\Delta a_n = a_{n+1} - a_n$ で定義される差分演算子 $\Delta = E - 1$ を用いて、

$$S = \frac{1}{1+E} a_0 = \frac{1}{2+\Delta} a_0 = \frac{1}{2} \sum_{n=0}^{\infty} \left(-\frac{1}{2} \Delta \right)^n a_0 \tag{4}$$

と変形します。級数 (3) から級数 (4) への変換ををオイラー変換といいます。この導出は形式的なものであるので、次にもう少し厳密な話をしましょう。まず、複素関数

$$f(z) = \sum_{n=0}^{\infty} a_n z^{n+1} \tag{5}$$

を考えます。ここで、$S = -f(-1)$ であることに注意します。次に、変数変換

$$z = \frac{t}{2+t} \tag{6}$$

を (5) に施して、t のテイラー級数として展開しなおすと、

$$\begin{aligned} f\left(\frac{t}{2+t}\right) &= \sum_{n=0}^{\infty} a_n \left(\frac{t}{2+t}\right)^{n+1} = \sum_{n=0}^{\infty} a_n \sum_{j=0}^{\infty} \binom{n+j}{n} (-1)^j \left(\frac{t}{2}\right)^{n+j+1} \\ &= \sum_{k=0}^{\infty} \sum_{n=0}^{k} a_n \binom{k}{n} (-1)^{k-n} \left(\frac{t}{2}\right)^{k+1} \\ &= -\frac{1}{2} \sum_{k=0}^{\infty} \left[\left(-\frac{1}{2}\Delta\right)^k a_0 \right] (-t)^{k+1} \end{aligned} \tag{7}$$

となります。ここで、$t=-1$ とおくと (7) 式はオイラー変換と同じになることがわかります。また、(6) 式は z 平面の単位円板を t 平面の $\mathrm{Re}\, t \geq -1$ の領域に写像する変換であり、収束を遅くする単位円上の特異点を原点からより遠くへ写すためのものです。したがって、この写像によって収束半径が広がれば、オイラー変換後の級数の収束は加速されることがわかります。

この (5) 式から (7) 式の導出を、(2) 式に適用すると

$$\tan^{-1} x = \frac{x}{1+x^2} \left(1 + \frac{2}{3} \frac{x^2}{1+x^2} + \frac{2\cdot 4}{3\cdot 5} \left(\frac{x^2}{1+x^2}\right)^2 + \frac{2\cdot 4\cdot 6}{3\cdot 5\cdot 7} \left(\frac{x^2}{1+x^2}\right)^3 + \cdots \right) \tag{8}$$

が得られます。オイラーはこの級数を 1755 年に発見しています。この級数は (2) 式とは異なり、すべての実数 x で収束します。また、(8) 式に $x=1$ を代入することで、グレゴリー級数のオイラー変換が得られ、10 桁の値を得るための項数はわずか 30 項で済みます。このことからも、オイラー変換の威力がわかると思います。

2.2.2 アークタンジェント公式

マチンは、1706 年に公式

$$\frac{\pi}{4} = 4\tan^{-1}\frac{1}{5} - \tan^{-1}\frac{1}{239}$$

を発見しました。証明はタンジェントの加法定理を繰り返し用いることで、容易にできます。アークタンジェントのテイラー展開 (2) を用いて計算すれば、わずか 10 項程度の計算で 10 桁以上の値が得られます。計算する項数と得られる桁数はほぼ比例して、円周率一桁当たりに必要な項数はおよそ $1/\log_{10} 5^2 + 1/\log_{10} 239^2 \simeq 0.926$ です。マチンはこの公式を用いて 100 桁の計算をしました。

マチンの公式のような

$$\pi = p_1 \tan^{-1}\frac{1}{q_1} + p_2 \tan^{-1}\frac{1}{q_2} + \cdots + p_m \tan^{-1}\frac{1}{q_m}$$

という形の公式はたくさん知られていています。以下にいくつかのアークタンジェント公式をあげておきます。

- クリンゲンシュティルナ (1730 年)

$$\frac{\pi}{4} = 8\tan^{-1}\frac{1}{10} - \tan^{-1}\frac{1}{239} - 4\tan^{-1}\frac{1}{515}$$

- ガウス (1863 年)

$$\frac{\pi}{4} = 12\tan^{-1}\frac{1}{18} + 8\tan^{-1}\frac{1}{57} - 5\tan^{-1}\frac{1}{239}$$

- シュテルメル (1896 年)

$$\frac{\pi}{4} = 6\tan^{-1}\frac{1}{8} + 2\tan^{-1}\frac{1}{57} + \tan^{-1}\frac{1}{239}$$

- 高野喜久雄 (1982 年)

$$\frac{\pi}{4} = 12\tan^{-1}\frac{1}{49} + 32\tan^{-1}\frac{1}{57} - 5\tan^{-1}\frac{1}{239} + 12\tan^{-1}\frac{1}{110443}$$

2.3 算術幾何平均による方法

1976 年、サラミンとブレントは独立かつ同時に、非常に速く収束する円周率の公式を発見しました。この方法は、以下に示す楕円積分を計算する算術幾何平均による方法と、ルジャンドルの関係式を組み合わせるものです。ここでは、算術幾何平均が何かということをまず説明し、次に楕円積分との関係を明らかにします。その上で、楕円積分に関するルジャンドルの関係式を導出し、それらを組み合わせることによって、この公式を得ることにします。

2.3.1 算術幾何平均

算術幾何平均反復とは漸化式

$$a_{n+1} = \frac{1}{2}(a_n + b_n)\,, \tag{9}$$

$$b_{n+1} = \sqrt{a_n b_n} \tag{10}$$

で定義されます。便宜上 $0 < b_0 \leq a_0$ として、補助的な数列を

$$c_{n+1} = \frac{1}{2}(a_n - b_n) \tag{11}$$

とします。このとき、関係式 $b_n \leq b_{n+1} \leq a_{n+1} \leq a_n$ と、

$$0 \leq a_{n+1} - b_{n+1} = \frac{1}{2}\frac{(a_n - b_n)^2}{(\sqrt{a_n} + \sqrt{b_n})^2} \tag{12}$$

が成り立ち、a_n, b_n は必ず同じ極限に収束することが容易にわかります。以後この極限を

$$M(a_0, b_0) = \lim_{n\to\infty} a_n = \lim_{n\to\infty} b_n$$

と表記することにします。次に、a_n, b_n の収束の速さを考えます。まず、$a_0 > 0$ と仮定すれば、(12) 式から

$$c_{n+2} = \frac{c_{n+1}^2}{4a_{n+2}} \leq \frac{1}{4M(a_0, b_0)}c_{n+1}^2 \tag{13}$$

が成り立ち、c_n は 0 に非常に速く収束することがわかります。一般に、

$$\frac{x_{n+1}-\alpha}{(x_n-\alpha)^p}=O(1)$$

が成り立つとき、x_n は α に p 次収束するといい、算術幾何平均の収束の速さは二次収束になります。この収束の速さは、たとえば c_n が 0 に 100 桁一致したならば、c_{n+1} は 200 桁、c_{n+2} は 400 桁と、その桁数が倍々に増えていくという急激なものです。さらに、

$$a_n-M(a_0,b_0)=c_{n+1}+c_{n+2}+c_{n+3}+\cdots\,,$$
$$b_n-M(a_0,b_0)=-c_{n+1}+c_{n+2}+c_{n+3}+\cdots$$

が成り立つので、a_n, b_n も $M(a_0,b_0)$ に二次収束することになります。この収束は、わずか $n=20$ 程度で 100 万桁一致し、$n=40$ 程度で 1 兆桁一致するというスピードです。

2.3.2 算術幾何平均と楕円積分

算術幾何平均と楕円積分との関係に最初に気づいたのはガウスであるといわれています。1799 年 5 月 30 日、彼の日記によると算術幾何平均のある極限 $1/M(1,\sqrt{2})$ と積分

$$\frac{2}{\pi}\int_0^1\frac{dt}{\sqrt{1-t^4}}$$

とが 11 桁以上の精度で数値的に一致することを確認したことが発端になっています [2]。

まず、算術幾何平均は以下の楕円積分

$$I(a,b) = \int_0^{\pi/2}\frac{d\theta}{\sqrt{a^2\cos^2\theta+b^2\sin^2\theta}}\,, \tag{14}$$
$$J(a,b) = \int_0^{\pi/2}\sqrt{a^2\cos^2\theta+b^2\sin^2\theta}\,d\theta \tag{15}$$

と密接な関係があることを示しましょう。

定理 1 $a_{n+1}=\frac{1}{2}(a_n+b_n)$, $b_{n+1}=\sqrt{a_nb_n}$, $0<b_n<a_n$ とすると

$$I(a_{n+1},b_{n+1}) = I(a_n,b_n)\,, \tag{16}$$
$$J(a_{n+1},b_{n+1}) = \frac{1}{2}(J(a_n,b_n)+a_nb_nI(a_n,b_n)) \tag{17}$$

が成り立つ。

証明 第一式 (16) の証明は、まず、(14) 式を変数変換 $x=b\tan\theta$ で書き換えて

$$I(a,b)=\int_0^\infty\frac{dx}{\sqrt{(a^2+x^2)(b^2+x^2)}}$$

とすることから始めます。次に、

$$I(\frac{a+b}{2},\sqrt{ab})=\frac{1}{2}\int_{-\infty}^\infty\frac{dx}{\sqrt{((\frac{a+b}{2})^2+x^2)(ab+x^2)}}$$

に対して変数変換 $x = \frac{1}{2}(t - ab/t)$ を施して

$$I(\frac{a+b}{2}, \sqrt{ab}) = \int_0^\infty \frac{dt}{\sqrt{(a^2+t^2)(b^2+t^2)}} = I(a,b)$$

を得ます。

第二式 (17) の証明の前に、準備をしておきましょう。次の関係式が成り立つことに注目します。

$$\frac{\partial J(a,b)}{\partial a} = \frac{a}{a^2-b^2}(J(a,b) - b^2 I(a,b)) , \tag{18}$$

$$\frac{\partial I(a,b)}{\partial a} = \frac{1}{a(a^2-b^2)}(J(a,b) - a^2 I(a,b)) \tag{19}$$

(18) 式の導出は、(15) 式を微分することで得られます。また、(19) 式の導出は (14) 式を微分した後、部分積分を行うことで得られます。

そして、これらの微分の関係式を用いることで、第二式 (17) の証明は次のようにして得られます。まず、第一の関係式 (16) を a_n に関して微分します。

$$\begin{aligned}\frac{\partial I(a_n,b_n)}{\partial a_n} &= \frac{\partial I(a_{n+1},b_{n+1})}{\partial a_n} \\ &= \frac{\partial a_{n+1}}{\partial a_n}\frac{\partial I(a_{n+1},b_{n+1})}{\partial a_{n+1}} + \frac{\partial b_{n+1}}{\partial a_n}\frac{\partial I(a_{n+1},b_{n+1})}{\partial b_{n+1}}\end{aligned}$$

これに、(19) 式を代入して右辺と左辺を比較すると

$$J(a_n,b_n) - a_n^2 I(a_n,b_n) = 2(J(a_{n+1},b_{n+1}) - a_n a_{n+1} I(a_{n+1},b_{n+1}))$$

が得られ、(17) 式が得られます。

証明終り

この定理から楕円積分 $I(a,b)$, $J(a,b)$ を計算する次の公式が作成されます。

定理 2 (計算公式)　$0 < b_0 \le a_0$, $c_0^2 = a_0^2 - b_0^2$ とし、

$$a_{n+1} = \frac{1}{2}(a_n + b_n) , \quad b_{n+1} = \sqrt{a_n b_n} , \quad c_{n+1} = \frac{1}{2}(a_n - b_n)$$

とするとき、

$$I(a_0,b_0) = \frac{\pi}{2}\frac{1}{M(a_0,b_0)} , \tag{20}$$

$$J(a_0,b_0) = \left(a_0^2 - \sum_{n=0}^{\infty} 2^{n-1} c_n^2\right) I(a_0,b_0) \tag{21}$$

である。

証明　第一式 (20) は、定理 1 の (16) 式

$$I(a_0,b_0) = I(a_1,b_1) = I(a_2,b_2) = \cdots = I(a_n,b_n)$$

から直ちに導かれます。ここで、$\lim_{n\to\infty} a_n = \lim_{n\to\infty} b_n = M(a_0, b_0)$ なので、

$$I(a_0, b_0) = \int_0^{\pi/2} \frac{d\theta}{\sqrt{M(a_0,b_0)^2\cos^2\theta + M(a_0,b_0)^2\sin^2\theta}} = \frac{\pi}{2}\frac{1}{M(a_0,b_0)}$$

となります。

第二式 (21) の証明は、定理 1 の (17) 式から次のようにして導かれます。まず、(17) 式を

$$D(a_n, b_n) = 2^n(a_n^2 I(a_0, b_0) - J(a_n, b_n))$$

を用いて書き換えると、

$$D(a_{n+1}, b_{n+1}) - D(a_n, b_n) = -2^{n-1}(a_n^2 - b_n^2)I(a_0, b_0)$$

となります。ここで、$c_n^2 = a_n^2 - b_n^2$ であることに注意して和をとると

$$D(a_{n+1}, b_{n+1}) - D(a_0, b_0) = -\sum_{j=0}^{n-1} 2^{j-1} c_j^2 I(a_0, b_0)$$

となります。もし、$n \to \infty$ で $D(a_n, b_n) \to 0$ と仮定すると、

$$D(a_0, b_0) = \sum_{n=0}^{\infty} 2^{n-1} c_n^2 I(a_0, b_0)$$

となり、(21) 式が得られます。

証明を完了させるために、最後に $\lim_{n\to\infty} D(a_n, b_n) = 0$ を示しましょう。

$$\begin{aligned} D(a_n, b_n) &= 2^n(a_n^2 I(a_n, b_n) - J(a_n, b_n)) \\ &= 2^n \int_0^{\pi/2} \frac{(a_n^2 - b_n^2)\sin^2\theta}{\sqrt{a_n^2\cos^2\theta + b_n^2\sin^2\theta}}\, d\theta \end{aligned}$$

であるので、

$$0 \le D(a_n, b_n) \le 2^n c_n^2 I(a_n, b_n)$$

が成立します。したがって、c_n が 0 に二次収束することを考慮すると

$$D(a_n, b_n) \to 0\,, \quad n \to \infty$$

となります。

証明終り

この計算公式の計算量は、a_n, b_n が二次収束するため非常に少なく、N 桁の精度を得るための反復は $\log N$ に比例する回数になります。したがって、必要な乗算回数も $\log N$ に比例する回数になります。

2.3.3 楕円積分とルジャンドルの関係式

まず、楕円積分の性質について触れておきます。第一種完全楕円積分 $K(k)$、第二種完全楕円積分 $E(k)$ は

$$\begin{aligned} K(k) &= \int_0^{\pi/2} \frac{d\theta}{\sqrt{1-k^2\sin^2\theta}} = I(\sqrt{1-k^2},1)\,, \\ E(k) &= \int_0^{\pi/2} \sqrt{1-k^2\sin^2\theta}\,d\theta = J(\sqrt{1-k^2},1) \end{aligned}$$

で定義されます。k は母数と呼ばれます。さらに補母数を $k' = \sqrt{1-k^2}$ として補積分を

$$K'(k) = K(\sqrt{1-k^2}) = K(k')\,, \quad E'(k) = E(\sqrt{1-k^2}) = E(k')$$

で定義します。このとき、J, I の微分の関係式 (18)、(19) を K, E に書き換えることで、

$$\frac{E(k)}{dk} = \frac{E(k)-K(k)}{k}\,, \quad \frac{K(k)}{dk} = \frac{E(k)-k'^2K(k)}{kk'^2}$$

が得られます。さらに、$K(k)$, $K'(k)$ は微分方程式

$$(k^3-k)\frac{d^2y}{dk^2} + (3k^2-1)\frac{dy}{dk} + ky = 0 \tag{22}$$

を満たします。これは、上の微分の関係式から容易に確認できます。

これらの性質からルジャンドルの関係式

$$E(k)K'(k) + E'(k)K(k) - K(k)K'(k) = \frac{\pi}{2} \tag{23}$$

が導かれます。この式の導出は、微分方程式 (22) を

$$G(k) = \sqrt{k}k'K(k)\,, \quad G^*(k) = \sqrt{k}k'K'(k)$$

に関する方程式に置き換えることで得られます。この G, G^* の満たす方程式は

$$\frac{d^2y}{dk^2} = -\frac{1}{4k^2}\left(\frac{1+k^2}{1-k^2}\right)^2 y$$

であり、

$$G\frac{d^2G^*}{dk^2} = G^*\frac{d^2G}{dk^2}$$

が成り立ちます。これを積分して、

$$G\frac{dG^*}{dk} - G^*\frac{dG}{dk} = \text{定数}$$

を得ます。これを K, K' で表し、微分を除去すると

$$EK' + E'K - KK' = \text{定数}$$

となります。積分定数は、テイラー展開を用いて $k \to 0$ の極限を計算して $\pi/2$ となります。このルジャンドルの関係式から、算術幾何平均を用いた円周率の計算が可能となります。

2.3.4 算術幾何平均による公式の導出

定理2と楕円積分におけるルジャンドルの関係式を用いて円周率を計算する方法を示します。楕円積分の母数を $k = k' = 1/\sqrt{2}$ と選ぶとき、この算法は以下のように得られます。

計算公式 1 (サラミン・ブレント) $a_0 = 1,\ b_0 = 1/\sqrt{2},\ c_0^2 = a_0^2 - b_0^2$ とし、

$$a_{n+1} = \frac{1}{2}(a_n + b_n)\ , \quad b_{n+1} = \sqrt{a_n b_n}\ , \quad c_{n+1} = \frac{1}{2}(a_n - b_n)$$

とするとき、

$$\pi = \lim_{n\to\infty} \frac{2a_{n+1}^2}{1 - \sum_{j=0}^{n} 2^j c_j^2} \tag{24}$$

である。

この算法では、a_n, c_n はともに二次収束するため、(24) 式も二次収束します。したがって、N 桁の精度を得るための反復回数は $M \simeq \log_2 N$ 回程度で済みます。また、この算法を M 回反復したときの主要な演算量は、平方根 $M+1$ 回と乗算 $2M-1$ 回と除算1回になることがわかります。

次に、この演算量を減らすことを考えます。最初に、平方根の回数は同じで乗算回数を半分にする算法を示します。これは、前の算術幾何平均反復を

$$A_n = a_n^2\ , \quad B_n = b_n^2\ , \quad C_n = c_n^2$$

で置き換えることで得られます [10]。

計算公式 1′ (改良サラミン・ブレント) $A_0 = 1,\ B_0 = 1/2,\ C_0 = A_0 - B_0$ とし、

$$\begin{aligned} A_{n+1} &= \frac{1}{2}\left(\frac{1}{2}(A_n + B_n) + \sqrt{A_n B_n}\right)\ , \\ B_{n+1} &= \sqrt{A_n B_n}\ , \\ C_{n+1} &= A_{n+1} - B_{n+1} \end{aligned}$$

とするとき、

$$\pi = \lim_{n\to\infty} \frac{A_n + B_n}{1 - \sum_{j=0}^{n} 2^j C_j} \tag{25}$$

である。

この改良算法の最後の式 (25) は、

$$2a_{n+1}^2 = 2A_{n+1} = A_n + B_n + O(c_{n+1}^2)$$

を用いて導出してあります。したがって、(24) 式と (25) 式の収束のオーダーは同じになります。この改良算法は、最初の反復での1による自明な乗算を考慮すると、M 回の反復計算で平方根 M 回と乗算 $M-1$ 回と除算1回を必要とします。したがって、この変形で乗算 M 回と平方根1回を節約できたことになります。

次に、四次収束する算法を示します。これは、本来の算術幾何平均反復を

$$\begin{aligned}
\alpha_n &= \sqrt{\frac{1}{2}(a_{2n+1}+b_{2n+1})} = \frac{1}{2}\left(\sqrt{a_{2n}}+\sqrt{b_{2n}}\right), \\
\beta_n &= \sqrt{\frac{1}{2}(a_{2n+1}-b_{2n+1})} = \frac{1}{2}\left(\sqrt{a_{2n}}-\sqrt{b_{2n}}\right), \\
\gamma_n &= c_{2n}^2+\frac{1}{2}c_{2n-1}^2
\end{aligned}$$

で置き換えることで得られます。

計算公式 2 (四次収束)　$\alpha_0=\frac{1}{2}(1+1/\sqrt[4]{2})$, $\beta_0=\frac{1}{2}(1-1/\sqrt[4]{2})$, $\gamma_0=1/2$ とし、

$$\begin{aligned}
\alpha_{n+1} &= \frac{1}{2}\left(\alpha_n+\sqrt[4]{(\alpha_n^2+\beta_n^2)(\alpha_n^2-\beta_n^2)}\right), \\
\beta_{n+1} &= \frac{1}{2}\left(\alpha_n-\sqrt[4]{(\alpha_n^2+\beta_n^2)(\alpha_n^2-\beta_n^2)}\right), \\
\gamma_{n+1} &= (2\alpha_n^2+\beta_n^2)\beta_n^2
\end{aligned}$$

とするとき、

$$\pi=\lim_{n\to\infty}\frac{2\alpha_n^4-\beta_n^4}{1-\sum_{j=0}^{n+1}4^j\gamma_j} \tag{26}$$

である。

この四次収束算法の最後の式 (26) は、

$$2a_{2n+3}^2=2\alpha_n^4-\beta_n^4+O(c_{2n+3}^2)$$

を用いて導出してあります。したがって、この四次収束算法の M 回の反復は、本来の計算公式 1 の $2M+2$ 回の反復に相当します。この算法は、$M/2-1$ 回の反復計算で四乗根 $M/2$ 回と乗算 $2M+1$ 回と除算 1 回を必要とします。ここで、四乗根の計算は直接のニュートン法の計算法を用いると、二回の平方根よりも少ない手間で計算できることに注意します。こうすることで、計算公式 1 よりも演算量は少なくなります。

このような高次収束する算法は、モジュラー関数の理論から組織的に導出することができます [2]。

3　多倍長計算の技法

ここでは、巨大な桁数をコンピュータを用いて計算する方法を解説します。まず、現代のコンピュータのハードウェアに備わった基本演算の精度は、32 ビット計算機ならば十進 9 桁の整数まで、64 ビット計算機ならば十進 19 桁の整数まで、浮動小数点 (実数) はどんなマシンでもたいてい十進 16 桁までしか扱えないということを念頭においてください。数十桁を超えるような計算は、四則演算などの基本演算ですらソフトウェアが必要で、計算手順 (算法) を指定しなければなりません。しかし、誰もが知っている小中学校で習う筆算の計算手順は、教育上は良いけれど実は非常に効率

の悪い計算手順であり、効率のよい計算手順と比較して、1 万桁で百倍程度、1 億桁で数十万倍の計算時間のロスが発生してしまうものなのです。高速で効率のよい計算を行うためには、良い計算手順が必要になります。この良い計算手順の導出には、さまざまな数学が使われており、その一部をここで紹介したいと思います。

3.1 乗算の高速化 1 (カラツバの方法)

R 進法の N 桁の乗算 fg を考えましょう。カラツバの方法は、まず N を偶数と仮定して

$$f = u_0 + R^{N/2}u_1\,, \quad g = v_0 + R^{N/2}v_1$$

のように桁を上下二つに分割し、

$$fg = (1 + R^{N/2})u_0v_0 + R^{N/2}(u_1 - u_0)(v_0 - v_1) + (R^{N/2} + R^N)u_1v_1$$

として計算します。$N/2$ 桁の乗算は 3 回しかないので、筆算の方法と比べて $N/2$ 桁の乗算 1 回分得をしたことになります。さらにこの計算を u_0v_0、$(u_1-u_0)(v_0-v_1)$、u_1v_1 の乗算に繰り返し適用することで、計算量はどんどん減って最終的に $N^{\log_2 3} \simeq N^{1.585}$ に比例する手間で計算できます。

この方法はディジタル法とも呼ばれ、1962 年にカラツバが公表したのが最初とされています。このカラツバの方法は、後に述べる高速フーリエ変換による方法と比較して非常に単純なので、よく使われる方法です。またカラツバの方法は、拡張することができて、最初の分割数を 2 分割から 3 分割、4 分割と増やせば、計算量はもっと少なくなりますが、いくら分割しても高速フーリエ変換による方法より演算量を少なくすることはできません。

3.2 乗算の高速化 2 (高速フーリエ変換による方法)

高速フーリエ変換を用いて乗算を高速化する方法を説明します。まず、乗算を畳み込み演算に変換し、次に畳み込み演算を高速フーリエ変換の問題に帰着させて、算法を導出します。最後は計算量の考察です。なお、高速フーリエ変換の詳細は 3.4 章で行います。

3.2.1 乗算と畳み込み演算の関係

まず、R 進法の N 桁整数表現を

$$f(R) = a_0 + a_1R^1 + a_2R^2 + \cdots + a_{N-1}R^{N-1}$$

で表すことにします。a_j は各桁の値で整数であり、通常は $0 \le a_j < R$ と正規化されているものとします。同様に、

$$g(R) = b_0 + b_1R^1 + b_2R^2 + \cdots + b_{N-1}R^{N-1}$$

として f と g の乗算を考えます。この乗算は、多項式の乗算と正規化のための桁処理演算に置き換えられることが容易にわかると思います。すなわち、この計算手順は次のようになります。まず、多項式 $f(x)$ と $g(x)$ の乗算

$$h(x) = f(x)g(x) \tag{27}$$

を行い、$h(x)$ の係数を求めます。ここで、$h(x)$ とその係数は

$$h(x) = c_0 + c_1 x^1 + c_2 x^2 + \cdots + c_{2N-2} x^{2N-2}\,,$$

$$c_j = \sum_{j'=0}^{N-1} a_{j'} b_{j-j'} \tag{28}$$

となります。(28) 式では $0 = b_{-1} = b_{-2} = \cdots$, $0 = b_N = b_{N+1} = \cdots$ であると仮定しています。この多項式の乗算係数 c_j は $0 \le c_j < R$ を満たさないため、次に桁処理 (桁上げの計算) を行う必要があり、その上で乗算結果 $h(R)$ が得られます。桁処理の演算は N のに比例する計算量で計算できるため、ここで問題となるのは多項式の乗算 (27) または畳み込み演算 (28) になります。

3.2.2 畳み込みと高速フーリエ変換の関係

ここでは、N 個の複素数点列 $a_0, a_1, \ldots, a_{N-1}$ に対する離散フーリエ変換を

$$A_k = \sum_{j=0}^{N-1} a_j W_N^{jk}\,, \quad W_N = \mathrm{e}^{-2\pi \mathrm{i}/N} \tag{29}$$

で定義します。一般に離散フーリエ変換は有限体の上でも定義できて、高速フーリエ変換まで破綻なく議論できますが、ここでは簡単のため、複素数の上でのフーリエ変換に限って考えます。この逆変換は、

$$a_k = \frac{1}{N} \sum_{j=0}^{N-1} A_j W_N^{-jk} \tag{30}$$

となります、このことは、(30) 式に (29) 式を代入することで容易に確認できます。便宜上、点列 a_j, A_k は、整数 m に対して $a_{Nm+j} = a_j$, $A_{Nm+k} = A_k$ を満たすように周期的に拡張しておきます。

この離散フーリエ変換は通常のフーリエ変換と同様に畳み込みを単純な積に変換する性質を持ちます。すなわち、畳み込み

$$c_j = \sum_{j'=0}^{N-1} a_{j'} b_{j-j'} \tag{31}$$

は、a_j, b_j, c_j の離散フーリエ変換 A_k, B_k, C_k を用いて

$$C_k = A_k B_k \tag{32}$$

に変換されます。(31) 式と (32) とが等価であることは、(32) 式に

$$A_k = \sum_{j=0}^{N-1} a_j W_N^{jk}\,, \quad B_k = \sum_{j=0}^{N-1} b_j W_N^{jk}$$

を代入することで容易に確認できます。また、a_j, b_j, c_j を多項式の係数としたとき、$A_k,\ B_k,\ C_k$ は $x = W_N^k$ での多項式の値そのものであることからも容易に確認できます。

この性質を用いることで、畳み込み (31) を離散フーリエ変換を用いて計算することができます。計算手順は

1. a_j, b_j の離散フーリエ変換 A_k, B_k を計算する。
2. $C_k = A_k B_k$ を計算する。
3. C_k に対して逆離散フーリエ変換を行い c_j を得る。

となります。離散フーリエ変換は後に示す高速フーリエ変換の算法で $N \log N$ に比例する計算量で計算できるため、畳み込みの計算量もこのオーダーとなります。

ここで少し注意しなければならないのは、(31) 式は巡回畳み込みになるということです。すなわち、周期的拡張 $b_{Nm+j} = b_j$ が暗黙のうちになされてしまいます。この巡回畳み込みを用いて、乗算の自然な畳み込み (28) 式を計算するためには少し工夫がいります。最も単純な方法は N を二倍にして上位の桁の部分に 0 をつめて巡回しないようにすることです。また 0 詰めを行わない方法として、1/2 ずれた拡張離散フーリエ変換

$$A'_k = \sum_{j=0}^{N-1} a_j W_N^{j(k+1/2)}, \quad W_N = \mathrm{e}^{-2\pi \mathrm{i}/N}$$

を用いる方法が考えられます。この拡張離散フーリエ変換により、巡回して回り込んだ部分が負になる巡回畳み込みが計算できます。この負の巡回畳み込みと通常の巡回畳み込みの両方を計算することで、回り込んだ部分を分離することができます。

3.2.3 乗算の演算量

前の節より、実数 (または複素数) の N 点の畳み込みを実行する演算量は高速フーリエ変換を用いることで、$N \log N$ の演算量になることを示しました。しかし、N 桁の乗算の計算量はこの $N \log N$ のオーダーよりも大きくなります。これは、高速フーリエ変換を行うときの実数 (または複素数) が有限の精度であるという当然の事実に起因します。例えば、計算機での実数の精度が 53 ビットで、この精度で高速フーリエ変換を行うと仮定すると、N と R は任意には選べず、(28) 式の畳み込みは $NR^2 > 2^{53}$ のときオーバーフロー (浮動小数点の場合は下位の整数部が削除) する可能性が出てきてしまいます。したがって、この方法では計算できる桁数に限界があり、たとえば $R = 2$ としたときの N の上限は 2^{50} 程度 (10 進で約千兆桁) になります。実際には高速フーリエ変換の誤差も入るので、これよりも条件はきつくなってしまいます。この限界を回避するための方法の一つは、乗算アルゴリズムを再帰的に用いることです。しかしそうすることで、計算量は $N \log N$ のオーダーよりも大きくなるのです。N 桁の乗算に必要な基本的な演算数を $\mu(N)$ で表すとすると、この方法の演算量は

$$\mu(N \log R) = O(N \log N \mu(\log(NR^2)))$$

を満たします。したがって、R を固定した場合の計算量は

$$\mu(N) = O(N(\log N)^2(\log\log N)^2(\log\log\log N)^2\cdots)$$

となります。これはまだ最適なオーダーではなく、最良と思われる評価は

$$\mu(N) = O(N\log N\log\log N)$$

であり、シュトラッセンの算法 [1] で実現されます。

次に、もう少し現実的な桁数の演算量について説明しましょう。ここでは、浮動小数点演算を用いて整数の畳み込みを実現する方法を考えます。ここで問題となるのが、R の選び方です。N を固定した場合、R を大きくすると多倍長乗算の計算桁数は上昇します。しかし、浮動小数点表現の計算機エプシロンを ε とするときの畳み込み (28) で整数が正しく表現される条件は、

$$NR^2 < \frac{1}{\varepsilon}$$

であるため、R には限界があります。実際には高速フーリエ変換の誤差も考慮しないといけないので、何らかの誤差評価を行い最大誤差を見積り、R を動的に定めることが必要になります。ここで、この誤差を少し小さくする技法について示します。これは、R 進法の N 桁整数表現

$$f(R) = a_0 + a_1R^1 + a_2R^2 + \cdots + a_{N-1}R^{N-1}\,, \quad 0 \le a_j < R$$

を、

$$f'(R) = a'_0 + a'_1R^1 + a'_2R^2 + \cdots + a'_{N-1}R^{N-1}\,, \quad |a'_j| \le \lfloor\frac{R}{2}\rfloor$$

に直すことでなされます。これで、約 4 ビットの精度の節約 (実際には a'_j はランダムに正と負が現れる場合が多いので効果は大きい) になります。この桁表現の変換法は、a_j, $j = N-1, N-2, \ldots, 1, 0$ に対して a_j が $R/2$ を越えたときに強制的に桁上げをすることでなされます。

3.3 除算、平方根の高速化

除算、平方根の計算は乗算と加減算の演算からなるニュートン法を用いて行うのが一般的です。この方法をうまく用いることで、除算、平方根の計算は乗算と同じオーダーの手間で計算できるようになります。

除算 b/a は、まず逆数 $1/a$ を計算して b を乗じる方法を用います。逆数 $1/a$ の計算は、方程式 $a - 1/x = 0$ に対するニュートン法

$$x_{n+1} = x_n + x_n(1 - ax_n) \tag{33}$$

を用います。このとき、この漸化式は、

$$x_{n+1} - a^{-1} = -a(x_n - a^{-1})^2$$

と書き換えることができるので、適切な初期値に対して二次収束することがわかります。要するに、一回の反復で桁数は約倍に増えることになります。

次に、逆数計算の演算量を調べます。その前に、N 桁の乗算に必要な演算量 $\mu(N)$ に条件

$$\mu(N/2) \leq \frac{1}{2}\mu(N)$$

があると仮定します。さらに、M 回ニュートン反復で N 桁の精度が得られると仮定します。このとき、$M-j$ 回目の反復で (33) 式は $N/2^j$ 桁の乗算と $(N/2)/2^j$ 桁の乗算がそれぞれ 1 回ずつ必要になります。$N/2^j$ 桁の乗算は a と x_n との乗算です。$(1-ax_n)$ の引き算で半分以上の桁落ちが発生するため、x_n と $(1-ax_n)$ の乗算は半分の $(N/2)/2^j$ 桁でよいことになります。したがってすべての反復で、乗算に費やす演算量は

$$\sum_{j=0}^{M-1}(\mu(N/2^j)+\mu((N/2)/2^j)) \leq \mu(N)\sum_{j=0}^{\infty}(3/2)2^{-j} = 3\mu(N)$$

となります。

平方根に関しては、まず方程式 $1/x^2 - a = 0$ に対するニュートン反復

$$x_{n+1} = x_n + \frac{x_n}{2}(1-ax_n^2)$$

を計算して、逆数平方根 $1/\sqrt{a}$ を求め、最後に a を乗算して平方根とします。このとき、逆数平方根と平方根の演算量はそれぞれ $4\mu(N)$ 以下と $5\mu(N)$ 以下という評価になります。

この演算量は、ニュートン反復を連立にすることで減ります [10]。このニュートン反復は、

$$x_n = x_{n-1} + x_{n-1}(1-y_n x_{n-1}), \tag{34}$$

$$y_{n+1} = y_n + \frac{1}{2}x_n(a-y_n^2) \tag{35}$$

であり、x_0, y_1 を $1/\sqrt{a}, \sqrt{a}$ の粗い初期値として反復させます。この反復も二次収束します。このニュートン反復の演算量は、逆数のときと同様に次のように評価できます。M 回の反復で x_M, y_{M+1} がそれぞれ $N/2, N$ 桁の精度が得られるとすると、$M-j$ 回目の反復で (34) 式において $(N/4)/2^j$ 桁の乗算が 3 回必要になり、(35) 式において $(N/2)/2^j$ 桁の乗算が 2 回必要になります。したがって、すべての反復で乗算に費やす演算量は

$$\sum_{j=0}^{M-1} 3\mu((N/4)/2^j) + 2\mu((N/2)/2^j) \leq 3.5\mu(N)$$

となります。

同様に、p 乗根に対する連立ニュートン反復は、x_0, y_1 を $a^{-(p-1)/p}, a^{1/p}$ の近似値として

$$x_n = x_{n-1} + x_{n-1}(1-y_n^{p-1}x_{n-1}),$$

$$y_{n+1} = y_n + \frac{1}{p}x_n(a-y_n^p)$$

の反復で求めることができ、逆数根をとる方法より高速に計算できます。

3.4 高速フーリエ変換の算法

高速フーリエ変換が一般に知られるようになったのは、1965 年のクーリーとチューキーによる短い論文 [7] からとされています。それ以前にも、何人かの数学者は知っていたようですが、一般に知られることはありませんでした。高速フーリエ変換があまり知られていなかったころは、N 点の離散フーリエ変換を計算するためには N^2 回の計算が必要であると信じられてきました。しかし、高速フーリエ変換を用いると $N \log N$ に比例する計算で済みます。この高速フーリエ変換の基本原理は、簡単な添字の変換で大きなサイズの離散フーリエ変換を計算が楽な小さな離散フーリエ変換に分解するという考えに基づきます。

まず、N 点の離散フーリエ変換

$$A_k = \sum_{j=0}^{N-1} a_j W_N^{jk}\,, \quad W_N = \mathrm{e}^{-2\pi \mathrm{i}/N} \tag{36}$$

を素直に計算する場合を考えます。この場合、A_0 から A_{N-1} までの各項の計算に N 回の乗算が入るため、全体で N^2 回の乗算が必要となります。しかし、もし N が 2 で割り切れるならば、添字 k を偶数と奇数に分けることで N 点の離散フーリエ変換は二つの $N/2$ 点の離散フーリエ変換

$$A_{2k} = \sum_{j=0}^{N/2-1} (a_j + a_{N/2+j}) W_{N/2}^{jk}\,, \tag{37}$$

$$A_{2k+1} = \sum_{j=0}^{N/2-1} (a_j - a_{N/2+j}) W_N^j W_{N/2}^{jk} \tag{38}$$

に容易に分解できます。$N/2$ 点の離散フーリエ変換は素直に計算して $N^2/4$ 回の乗算で実行できるので、この分解で計算量は約半分に減ることになります。さらに、この分解を 2 回 3 回と繰り返せば計算量は約 1/4, 1/8 と激減します。これがクーリー・チューキー算法 (正確には、基数 2、周波数間引き算法) の基本的な考え方になります。

この分解を $\log_2 N$ 回行い、1 点の自明な離散フーリエ変換になるまで行ったときの計算量を考えます。この分解自体には各々の段で W_N^j を乗ずる $N/2$ 回の複素数乗算と N 回の複素数加算が必要で、複素数乗算回数は $(N/2)\log_2 N$ に減少します。したがって、浮動小数点演算の量は $N \log_2 N$ のオーダーとなります。これは、クーリー・チューキー算法の典型的な演算量で、次にあげる様々な高速フーリエ変換の演算量の削減法は、基本的にこのオーダーの比例定数と、$N \log_2 N$ より低い次数の項を小さくするものです。

3.4.1 様々な高速フーリエ変換の算法

ここでは単純な二分割ではなく、より複雑な分解による高速フーリエ変換の算法について調べてみます。まず、N が $N = N_1 N_2$ と因数分解できると仮定します。このとき、(36) 式の添字 j を次の二つの添字 $j_1 = 0, 1, 2, \ldots, N_1 - 1,\ j_2 = 0, 1, 2, \ldots, N_2 - 1$ に置き換えることを考えます。そこで、J_1, J_2 をある自然数として、j_1, j_2 から j に変換する写像

$$j \equiv (J_1 j_1 + J_2 j_2) \bmod N \tag{39}$$

を定義します。まず第一に、この写像は1対1とならなければならないが、このための必要十分条件は、p, q をある自然数とするとき、

1. N_1 と N_2 が互いに素の場合：
   $J_1 = pN_2$, $J_2 = qN_1$ の少なくとも一方が満たされ $\gcd(J_1, N_1) = \gcd(J_2, N_2) = 1$

2. N_1 と N_2 が互いに素ではない場合：
   $J_1 = pN_2$ かつ $J_2 \bmod N_1 \not\equiv 0$ かつ $\gcd(p, N_1) = \gcd(J_2, N_2) = 1$　または
   $J_1 \bmod N_2 \not\equiv 0$ かつ $J_2 = qN_1$ かつ $\gcd(J_1, N_1) = \gcd(q, N_2) = 1$

であることが簡単に証明できます [6]。さらに、添字 k に関しても同様の写像

$$k \equiv (K_1 k_1 + K_2 k_2) \bmod N \tag{40}$$

を定義して、(36) 式に対してこれらの変換を適用すると、次のようになります。

$$A_{K_1 k_1 + K_2 k_2} = \sum_{j_2=0}^{N_2-1} \sum_{j_1=0}^{N_1-1} a_{J_1 j_1 + J_2 j_2} W_N^{J_1 K_1 j_1 k_1} W_N^{J_1 K_2 j_1 k_2} W_N^{J_2 K_1 j_2 k_1} W_N^{J_2 K_2 j_2 k_2} \tag{41}$$

しかし、まだこのままでは演算ブロックの順序が入れ換えることができず、小さな離散フーリエ変換に分解することはできません。式の中の二番目と三番目の W の項が邪魔なのです。もし、条件

$$J_1 K_2 \equiv 0 \bmod N \text{ または } J_2 K_1 \equiv 0 \bmod N \text{ の少なくとも一方} \tag{42}$$

が成り立つのならば、(41) 式は N_1 と N_2 の二つの小さな離散フーリエ変換に分解されることがわかります。これらの条件を満たす例として、次の二種類の分解が考えられます。

1. N_1 と N_2 が互いに素の場合：

$$J_1 = N_2 \text{ かつ } J_2 = N_1 \text{ かつ } K_1 = N_2 \text{ かつ } K_2 = N_1$$

2. N_1 と N_2 が任意の場合：

$$J_1 = N_2 \text{ かつ } J_2 = 1 \text{ かつ } K_1 = 1 \text{ かつ } K_2 = N_1$$

   または

$$J_1 = 1 \text{ かつ } J_2 = N_1 \text{ かつ } K_1 = N_2 \text{ かつ } K_2 = 1$$

第一の場合は N_1 と N_2 が互いに素の場合のみに用いられる分解で、(41) 式の W の項を二つ消去して N_1 と N_2 の二次元離散フーリエ変換に分解します。この分解は、N_1, N_2 を互いに素になるように選ぶ必要があるけれども、分解に必要な演算量はゼロになるという利点があります。しかし、分解しきれずに残った離散フーリエ変換はたいてい素数の長さであり、ある程度の計算量は必要になります。この分解による算法は素因数高速フーリエ変換 [9] やウイノグラード算法 [9, 13] に用いられます。一方、第二の分解は N_1 と N_2 は任意でよい代わりに (41) 式の W の項は一つしか消去されず、N_1 と N_2 の離散フーリエ変換への分解に W を乗算する演算 (回転因子の乗算) が

必要になります。しかし、N_1 または N_2 を離散フーリエ変換が容易に計算できる数に固定できるため、分解以外に必要な計算量は少なくなるという利点があります。この分解による算法はクーリー・チューキー算法 [7] であり、N_1 を固定して分解を再帰的に繰り返すのが基本算法になります。このとき N_1 を基数といい、(41) 式の二項目の W を消し去る場合を周波数間引き、三項目の W を消し去る場合を時間間引き算法と呼びます。このクーリー・チューキー算法には多くの種類があり、通常の基数 2 の算法、任意基数算法、混合基数算法、演算量が少ないとされる Split-Radix 算法 [8, 12] などがあげられます。

4　終わりに

以上のように、円周率計算について主にコンピュータを用いた計算技法を紹介してきました。高精度な円周率を求めるという行為は、人間がどうしても手の届かないものに少しでも近づこうとするかのようです。これは一見、実用的でない無駄な作業のように見えるかもしれません。しかし、古くから円周率の計算で用いられる技法の多くは、数値計算の分野で広く活用されており、その計算の効率化により、科学技術の発展に寄与しているものと私は確信しています。

参考文献

[1] A. Aho, J. Hopcroft, J. Ullman 著, 野崎昭弘, 野下浩平 訳, アルゴリズムの設計と解析 *I, II* , サイエンス社 1977.

[2] J. Borwein, P. Borwein, *Pi and the AGM* , A Wiley-Interscience Publication 1987.

[3] J. Delahaye 著, 畑政義 訳, π –魅惑の数, 朝倉書店 2001.

[4] 金田康正 著, π のはなし, 東京図書 1991.

[5] R. Brent, *Fast Multiple-Precision Evaluation of Elementary Functions*, J. ACM 23 1976.

[6] C. Burrus, *Index Mappings for Multidimensional Formulation of the DFT and Convolution*, IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.25 No.3 (1977), 239–242.

[7] J. Cooley, J. Tukey, *An Algorithm for the Machine Calculation of Complex Fourier Series*, Mathematics of Computation, Vol.19 (1965), 297–301.

[8] P. Duhamel, H. Hollmann, *Split-Radix FFT Algorithm*, Electronics Letters, Vol.20 (1984), 14–16.

[9] D. Kolba, *A Prime Factor FFT Algorithm Using High-Speed Convolution*, IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.25 No.4 (1977), 281–294.

[10] 大浦拓哉, 円周率公式の改良と高速多倍長計算の実装, 日本応用数理学会論文誌, Vol.9 No.4, 1999.

[11] E. Salamin, *Computation of π Using Arithmetic-Geometric Mean*, Mathematics of Computation, Vol.30 1976.

[12] H. Sorensen, M. Heideman, C. Burrus, *On Computing the Split-Radix FFT*, IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.34 No.1 (1986), 152–156.

[13] S. Winograd, *On Computing the Discrete Fourier Transform*, Mathematics of Computation, Vol.32 (1978), 175–199.