Proc. Natl. Acad. Sci. USA
Vol. 86, pp. 8178–8182, November 1989
Mathematics

The computation of classical constants

D. V. Chudnovsky and G. V. Chudnovsky

Department of Mathematics, Columbia University, New York, NY 10027

Communicated by Herbert E. Robbins, August 11, 1989

ABSTRACT Hypergeometric representations of classical constants and efficient algorithms for their calculation are discussed. Particular attention is devoted to algorithms for computing π.

The arithmetic nature of classical constants of analysis and geometry is the main focus of transcendental number theory. Typical questions include: are the constants irrational, transcendental, algebraically independent? how well are they approximated by rational numbers? what are the continued fraction expansions of these numbers? Less often are questions posed about radix expansions of classical constants: are these constants normal? do they resemble "random" digit expansions in other ways? These problems were raised a long time ago, and answers are important not only in number theory but also in complexity theory, where the basic problem of randomness versus high complexity needs to be understood in these crucial test cases. Among practical applications one can mention the problem of reliable generation of the truly random sequences needed in many computational, test, and communication applications.

These considerations provide the incentive for undertaking multimillion digit computations of interesting classical constants. Such high-precision computations are also needed for other number theoretic applications such as continued fraction expansions, where theoretical results are still inadequate. From the hardware point of view, multimillion digit computations have been an accepted test of the viability and integrity of computer systems since the first days of large digital devices (von Neumann, Shanks). There is a race for leadership in these computations, where the performance of algorithms and supercomputers is measured in terms of millions of digits of π computed using a given algorithm on a given supercomputer. The first million digit mark was passed in 1973 by Guilloud and Bonyer on a CDC 7600. The 2 million mark was passed in 1981 by Miyoshi and Kanada and by Guilloud. In 1982 Tamura and Kanada computed 4 million and 8 million digits of π. In 1983 Kanada and Tamura computed about 16 million digits on a Hitachi S-810. Gosper in 1985 computed more than 17 million digits of π (and as many terms in the continued fraction expansion of π) using only a SYMBOLICS workstation. In early 1986 Bailey computed about 30 million digits of π using a newly constructed Cray 2. Then in 1987 Kanada computed 134 million digits on a NEC SX-2 supercomputer. In 1988 Kanada raised this to 201 million, using a Hitachi supercomputer.

Algorithm development helped considerably in these computations. One of the components was the fast multiplication of long integers: if $M(n)$ denotes the complexity of multiplication of two n-digit integers, then the theoretical upper bound $M(n) = O(n\log n\log\log n)$ (1, 2) is practically realizable in "bignum" packages. While classical computation of π and other similar constants had complexity $O(n^2)$ for the first n digits, new algorithms were developed in the early 1970s that reduced this to $O(M(n)\log^2 n)$ or even $O(M(n)\log n)$. Particularly popular are Salamin–Brent algorithms for π or e with complexity of only $O(M(n)\log n)$, using Gauss' arithmetic–geometric mean (agm) and Legendre's identity between periods and quasi-periods of an arbitrary elliptic curve (3, 4). Salamin algorithms and their generalizations (5) using elliptic modular transformations were used by Kanada and by Bailey (see ref. 5) in their record-breaking computations. Unfortunately, fast agm algorithms are inherently floating point in nature, so that error accumulations due to round-off are not self-correcting. As a result, there is no independent means of verifying the final result short of recomputing everything on different hardware or comparing it with the result derived in a different way. Gosper's computation was different in that he used a generalized hypergeometric representation of a multiple of π derived by Ramanujan (6) as a part of his period relation identities for singular moduli of elliptic curves (see Eq. **2**).

Since 1984 we have been interested in period relations and their representation in terms of hypergeometric function identities from the point of view of applications to diophantine approximations to numbers such as π, $\pi/\sqrt{3}$, ... Aided by the computer algebra system SCRATCHPAD, new identities were discovered and used for diophantine approximations (see ref. 7). These identities were generalized in refs. 8 and 9 from elliptic curves to arbitrary CM-varieties, so that a larger class of classical constants could be represented through combination of values of convergent hypergeometric series. Combining then fast convolution and long integer multiplication algorithms (2) with fast power series techniques (8), one can derive a relatively simple implementation of high-precision calculations of classical constants adaptable to any modern (super) computer. A series of such calculations was started by us at the end of 1988 on several machines in a time-shared environment. By May 1989, on two machines of different architecture 480 million decimal digits of π were computed using identity **1**. These machines were the Cray 2 at the Minnesota Supercomputer Center (Minneapolis) and the IBM 3090-VF at the IBM T. J. Watson Research Center (Yorktown Heights, NY).

Section 1. Hypergeometric Functions and Constants

Classical constants often arise from interesting classes of functions. Among these many have an arithmetic nature, at least in the sense that such functions are well defined and convergent in both the archimedean (real or complex) and nonarchimedian (p-adic) domains. In transcendental number theory, solutions of linear differential equations with such arithmetic properties were introduced by Siegel (10). Let a_0, . . . , a_n, . . . be algebraic numbers such that all sizes $\overline{|a_n|}$ and common denominators den $\{a_0, . . . , a_n\}$ are bounded by C^n for some constant $C > 1$. Then the function $f(x) = \sum_{n=0}^{\infty} a_n x^n$ is called a G-function, and $f(x) = \sum_{n=0}^{\infty}(a_n/n!)x^n$ is called an E-function (10). It was proved in ref. 11 that linear differential equations satisfied by G-functions have special geometric

The publication costs of this article were defrayed in part by page charge payment. This article must therefore be hereby marked "*advertisement*" in accordance with 18 U.S.C. §1734 solely to indicate this fact.

Abbreviations: agm, arithmetic–geometric mean; FFT, fast Fourier transform.

properties: these differential equations are globally nilpotent. (Among other things this means that all solutions of such differential equations with algebraic initial conditions are G-functions.) Moreover, on the basis of our results on the Grothendieck conjecture (11) and extensive computer experiments, we can support the conjecture that "all arithmetically interesting differential equations came from geometry." This means that any G-function satisfying a linear differential equation over $\overline{\mathbf{Q}}(x)$ can be expressed algebraically in terms of solutions of Picard–Fuchs equations—combination of algebraic functions and periods of deformations of algebraic varieties. There is a more precise expression for this conjecture [Dwork–Siegel hypothesis (9, 10)]: all G-functions can be expressed in terms of algebraic combinations of (integrals of) generalized hypergeometric functions ${}_{p+1}F_p$. Though these conjectures are unproved, in many specific cases one finds a variety of expressions of classical constants arising from arithmetic or geometry in terms of rapidly convergent generalized hypergeometric functions that are well suited for high-precision numerical evaluation and for derivation of diophantine approximations to these constants. Generalized hypergeometric functions are usually defined as power series whose consecutive coefficients satisfy rank one linear recurrence with coefficients that are rational functions of indices:

$$ {}_mF_n(a_1, \ldots, a_m; b_1, \ldots, b_n; x) = \sum_{N=0}^{\infty} \frac{\prod_{i=1}^{m}(a_i)_N}{\prod_{j=1}^{n}(b_j)_N} \cdot \frac{x^N}{N!}, $$

$$ (c)_N = (c) \cdots (c + N - 1). $$

Constants expressible in terms of values of generalized hypergeometric functions can be called "rank two" constants. Here is a very simple scheme for computing such constants in terms of truncated generalized hypergeometric series.

Algorithm I. Consider the following scheme of computation of (rational number representations of) truncated (generalized) hypergeometric series:

$$ \begin{pmatrix} a & 0 \\ b & c \end{pmatrix} = \begin{pmatrix} A(0) & 0 \\ B(0) & C(0) \end{pmatrix} \cdots \begin{pmatrix} A(N-1) & 0 \\ B(N-1) & C(N-1) \end{pmatrix} $$

where $A(\cdot), B(\cdot), C(\cdot) \in \mathbf{Z}[\cdot]$ and $f = b/a$ is the rational number representing the N first terms in the generalized hypergeometric series. Here c is the numerator of the Nth coefficient, b is the numerator of the Nth order truncated series, and a is the common denominator. Then the simplest way to compute a, b, and c is the following:

Stage 1 (initialization). Put $M_k = \begin{pmatrix} A(k) & 0 \\ B(k) & C(k) \end{pmatrix}$ for $k = 0, \ldots, N - 1$.

Stage 2 (multiplication). Put $M_k = M_{2\cdot k} \times M_{2k+1}$ for $k = 0, \ldots, [N/2] - 1$, and $M_k = M_{N-1}$ for $k = (N - 1)/2$ for odd N.

Stage 3 (recursion). Put $N = \text{ceiling}(N/2)$. If $N > 1$ go to *Stage 2*, otherwise return $\begin{pmatrix} a & 0 \\ b & c \end{pmatrix} = M_1$.

While we recommend this algorithm for many practical implementations, it is not necessarily the best in complexity. More efficient schemes based on these principles are discussed in ref. 8 for evaluation of solutions of arbitrary linear differential equations. Such schemes generalizing *Algorithm I* often require an $n \times n$ matrix representation and can result in larger storage requirements. Alternatively one can use, instead of a power series, a continued fraction expansion with partial fractions that are rational functions of the index. The first study of these classes of algorithms and of the extensions needed to compute continued fraction expansions of b/a from the same scheme was that of Gosper (12).

Bounds on the complexity of *Algorithm I* and any of its improvements depend considerably on the arithmetic properties of the corresponding generalized hypergeometric series. The worst case bound is the following.

PROPOSITION 1.1 *The cost of computation of* c/a *in the tree implementation of matrix multiplication*

$$ \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \prod_{n=0}^{N-1} \begin{pmatrix} A(n) & B(n) \\ C(n) & D(n) \end{pmatrix} $$

(*for* $A(\cdot), B(\cdot), C(\cdot), D(\cdot)$ *from* $\mathbf{Z}[\cdot]$) *is at most* $O(M(N) \cdot \log^2(N + 1))$. *The cost is similar for the computation of* N *terms of the continued fraction expansion of* c/a.

The proof of *Proposition 1.1* is reduced to application of *Algorithm I*, with summation of complexities of all $O(\log N)$ applications of *Stage 2* of this algorithm.

Complexity bounds on the computation of c/a in *Proposition 1.1* can be decreased if additional arithmetic conditions on generalized hypergeometric functions are imposed. This is the case for generalized hypergeometric E-functions. For such functions the value of c/a of size H can be computed in only $O(M(H) \cdot \log H)$ primitive operations, following the scheme of *Algorithm I*. For G-functions similar reductions in complexity are possible, up to $O(M(N)\log N)$ complexity.

Section 2. Traditional Algorithms of Computation of π

The classical approach to computing π uses the hypergeometric function arctan. The most popular is Machin's formula $\pi = 16 \cdot \arctan(1/5) - 4\arctan(1/239)$. Since simple methods of multiplication were used in pre-fast Fourier transform (FFT) days, the complexity of computation of N digits of π was proportional to N^2. If one substitutes fast multiplication methods and uses *Algorithm I* for classical identities for π, the complexity drops in the worst case to $O(M(N)\log^2 N)$. In the early 1970s faster algorithms were suggested, built around computations on an elliptic curve rather than on a circle. They reduced the complexity of computation of N digits of π to $O(M(N)\log N)$ (3, 4)

Salamin's Algorithm (3). Initialize $a_0 = 1$, $b_0 = 1/\sqrt{2}$, and compute the agm: $a_n = (a_{n-1} + b_{n-1})/2$, $b_n = (a_{n-1} \cdot b_{n-1})^{1/2}$. Put $c_n^2 = a_n^2 - b_n^2$. Then a_n and b_n converge to the same limit: $\lim a_n = \lim b_n = \text{agm}(a_0, b_0)$, and

$$ \pi = 4 \cdot agm(a_0, b_0)^2/(1 - \Sigma 2^{n+1}c_n^2), $$

with partial results $\pi_n = 4a_{n+1}^2/(1 - \Sigma_{j=1}^{n} 2^{j+1}c_j^2)$, approximating π with the order $\exp(-O(2^{n+1}))$.

For practical implementation of this scheme one should add to the bignum package fast division and square roots of arbitrary precision numbers. This is done, typically, with Newton algorithms, and the complexity of these algorithms on numbers with N digits is $O(M(N))$ (see ref. 4).

Salamin's algorithm and its modifications are made of two ingredients: Gauss's agm algorithm of computation of periods and quasi-periods of an elliptic curve, and Legendre's identity for these periods and quasi-periods. In Legendre's notation periods and quasi-periods of the elliptic curve (cubic) $y^2 = x \cdot (x - 1) \cdot (x - \lambda)$ with modulus λ are denoted by $K(\lambda)$, $K(\lambda')$, and $E(\lambda)$, $E(\lambda')$, respectively, for $\lambda' = 1 - \lambda$. They are represented by hypergeometric functions $K(\lambda) = \pi/2 \cdot {}_2F_1(1/2, 1/2; 1; \lambda)$ and $E(\lambda) = \pi/2 \cdot {}_2F_1(-1/2, 1/2; 1; \lambda)$. The Legendre relation between periods and quasi-periods is $K(\lambda) \cdot E(\lambda') + K(\lambda') \cdot E(\lambda) - K(\lambda) \cdot K(\lambda') = \pi/2$. Salamin's algorithm utilizes this identity at $\lambda = \lambda' = 1/2$ and uses Gauss' agm algorithm to compute K and E. Higher order modular relations are used to provide even faster convergent algorithms (5). Unfortunately, a full precision of $O(N)$ digits has to be preserved at every step of the calculation (including initialization) with extra $O(\log N)$ guard digits.

Section 3. Period Relations and Complex Multiplications

As we have seen in *Section 2*, period and quasi-period relations of arbitrary elliptic curves can be used for the fast computation of π. Special elliptic curves can be used to construct fast schemes of computation of π that can be represented as hypergeometric identities. These elliptic curves possess a complex multiplication, expressed in the statement that the ratio of two periods is a quadratic algebraic number. Ramanujan (6) proved that for any elliptic curve with complex multiplication (for any singular modulus λ), there are two linear relations between periods and quasi-periods (between K, K', E, E') with algebraic number coefficients. Substituting these two linear relations into the Legendre identity, Ramanujan arrived at the expression of an algebraic multiple of π as a quadratic function of a period and a quasi-period—$K(\lambda)$ and $E(\lambda)$. Moreover, Ramanujan presented this quadratic period relation in terms of a single ${}_3F_2$ function.

From Legendre representation of periods and quasi-periods, one can derive several other representations in terms of hypergeometric functions. There are four such classes corresponding to four triangle subgroups: $\Gamma(1)$, $\Gamma(2)$, and Hecke groups G_q, $q = 4, 6$ (see ref. 8). Ramanujan's quadratic representations use (*i*) simple fractional transformations of ${}_2F_1$-functions, (*ii*) a single quadratic relation valid for hypergeometric functions

$${}_2F_1(2a, 2b; c-a-b; z) = {}_2F_1(a, b; c-a-b; 4z(1-z)),$$

and (*iii*) a Clausen identity

$${}_2F_1(a, b; a+b+1/2; z)^2 = {}_3F_2(2a, a+b, 2b; a+b+1/2, 2a+2b; z).$$

One can derive (7–9) all classes of quadratic period relations from the most general one for the modular invariant $J = J(\tau)$, using Eisenstein's series

$$E_k(\tau) = 1 - \frac{2k}{B_k} \cdot \sum_{n=1}^{\infty} \sigma_{k-1}(n) \cdot q^n$$

for $\sigma_{k-1}(n) = \Sigma_{d|n} d^{k-1}$ and $q = e^{2\pi i\tau}$. The standard theory of complex multiplication states that for an arbitrary elliptic curve over $\overline{\mathbf{Q}}$ with complex multiplication by $\sqrt{-d}$ and with periods $\omega_1, \omega_2{:}\tau = \omega_1/\omega_2 \in H$, all ratios $E_{2n}(\tau){:}(\omega_2/2\pi i)^{2n}$ for $n > 1$ are algebraic numbers. Ramanujan (6) proved a new algebraicity statement for a nonholomorphic (Kronecker's) version of $E_2(\tau)$ (13):

LEMMA 3.1. *If* $\tau \in \mathbf{Q}(\sqrt{-d})$, *then the* $\Gamma(1)$-*invariant nonholomorphic series*

$$s_2(\tau) \overset{def}{=} \{E_2(\tau) - 3/(\pi Im(\tau))\} \cdot E_4(\tau)/E_6(\tau),$$

has an algebraic value (from a Hilbert class field $\mathbf{Q}(\sqrt{-d}, J(\tau))$).

Using Fricke's hypergeometric function representation of periods (7) in terms of $F(z) = {}_2F_1(1/12, 5/12; 1; z)$, *Lemma 3.1*, and the Legendre identity, we obtain Ramanujan's quadratic relation as the sum of products of rapidly convergent ${}_2F_1$ series representing $1/\pi$

$$F(12^3/J)^2 \frac{1}{12}(1 - s_2(\tau)) + F(12^3/J)F_z(12^3/J)\frac{12^3}{J} = \frac{J^{1/2}}{(J-12^3)^{1/2} 2\pi\sqrt{d}}.$$

Here $J = J(\tau)$, $\tau = (-1 + \sqrt{-d})/2$, $d > 0$, $d \equiv 3(4)$. Here, according to *Lemma 3.1*, $s_2(\tau)$ is an algebraic number from a real subfield of a Hilbert field $\mathbf{Q}(\sqrt{-d}, J(\tau))$. Next, according to the Weber–Heegner result, $J^{1/3}$ and $(J - 12^3)^{1/2}/\sqrt{-d}$ are (real) algebraic integers of degree $h(-d)$. This means that for a class 1 discriminant $-d$, all coefficients on the left-hand side of the identity are rational numbers, while on the right-hand side we have a rational multiplier of $(-J)^{1/6}/\pi$, where $(-J)^{1/6}$ is a quadratic irrationality.

Now it is enough to apply a special case of the Clausen identity

$${}_2F_1(1/12, 5/12; 1; z)^2 = {}_3F_2(1/6, 5/6, 1/2; 1, 1; z).$$

This equation allows us to represent quadratic period relations in the ${}_3F_2$- form

$$\sum_{n=0}^{\infty} \left\{\frac{1}{6}(1 - s_2(\tau)) + n\right\} \cdot \frac{(6n)!}{(3n)!n!^3} \cdot \frac{1}{J(\tau)^n} = \frac{(-J(\tau))^{1/2}}{\pi} \cdot \frac{1}{(d(1728 - J(\tau))^{1/2}};$$

here $\tau = (1 + \sqrt{-d})/2$. The largest one class discriminant $-d = -163$ gives the most rapidly convergent series among those series where all numbers in the left side are *rational*:

$$\sum_{n=0}^{\infty} \{c_1 + n\} \cdot \frac{(6n)!}{(3n)!n!^3} \frac{(-1)^n}{(640{,}320)^{3n}} = \frac{(640{,}320)^{3/2}}{163 \cdot 8 \cdot 27 \cdot 7 \cdot 11 \cdot 19 \cdot 127} \cdot \frac{1}{\pi}. \quad [1]$$

Here from *Lemma 3.1*, $c_1 = 13{,}591{,}409/(163 \cdot 2 \cdot 9 \cdot 7 \cdot 11 \cdot 19 \cdot 127)$ and $J((1 + \sqrt{-163})/2) = -(640{,}320)^3$.

Ramanujan provides instead of this a variety of other formulas connected mainly with the three other triangle groups commensurable with $\Gamma(1)$. The four classes of ${}_3F_2$ hypergeometric functions (that are squares of ${}_2F_1$-representations of complete elliptic integrals) are ${}_3F_2(1/2, 1/6, 5/6; 1, 1; x)$, ${}_3F_2(1/4, 3/4, 1/2; 1, 1; x)$, ${}_3F_2(1/2, 1/2, 1/2; 1, 1; x)$, and ${}_3F_2(1/3, 2/3, 1/2; 1, 1; x)$.

Representations similar to Eq. **1** can be derived for any of these series for any singular modulus $\tau \in \mathbf{Q}(\sqrt{-d})$ and for any class number $h(-d)$, thus extending Ramanujan's list (6) ad infinum. Ramanujan's own favorite was

$$\frac{9801}{2\sqrt{2}\pi} = \sum_{n=0}^{\infty} \{1103 + 26{,}390n\} \frac{(4n)!}{n!^4 \cdot (4 \cdot 99)^{4n}}, \quad [2]$$

which was used by Gosper in 1985 for the computation of more than 17 million terms in the continued fraction (and decimal) expansion of π.

These rapidly convergent series were applied in ref. 7 to obtain new measures of diophantine approximation to algebraic multiples of π. Extensions of the theory of period relations to multidimensional CM-varieties are given in ref. 8.

Section 4. *p*-adic Period Congruences

In addition to archimedean period relations in the complex multiplication case, there are corresponding nonarchimedean (p-adic) relations reflecting the same modular numbers. One such relation is the Koblitz–Gross formula giving a p-adic interpretation of the Selberg–Chowla expression for periods of elliptic curves with complex multiplication.

In applications to congruences satisfied by hypergeometric approximations to multiples of $1/\pi$ we need congruences satisfied by truncated hypergeometric series that can be interpreted through Hasse invariants and traces of Frobenius. We briefly describe the background of congruences, taking the Legendre model of elliptic curves

$$y^2 = x \cdot (x-1) \cdot (x-\lambda). \qquad [3]$$

Over finite fields there is a well-known relation between Hasse invariants and mod p reduction of solutions of the (Picard–Fuchs = Legendre) linear differential equation. Such a relationship is very general and is derived using Serre's duality. For elliptic curves in the Legendre form, mod p interpretation is particularly easy. One reduces all coefficients of the power series expansion of ${}_2F_1(1/2, 1/2; 1; \lambda)$mod p. In this way one arrives at a polynomial mod p known as the Hasse–Deuring polynomial: $H_p(\lambda) = \Sigma_{i=0}^{m}\binom{m}{i}^2\lambda^i$, $m \overset{\text{def}}{=} (p-1)/2$, of degree m in λ.

LEMMA 4.1. *The trace* $a_p(\lambda)$ *of Frobenius of the elliptic curve* **3** *over* $\mathbf{F}_p$ *for* $\lambda \in \mathbf{F}_p$ *satisfies the following congruence:*

$$a_p(\lambda) \equiv (-1)^m \cdot H_p(\lambda) \mathit{mod}\ p.$$

The number $N_p(\lambda)$ *of* $\mathbf{F}_p$*-rational points on the elliptic curve* **3** *is*

$$N_p(\lambda) \equiv 1 - (-1)^m \cdot H_p(\lambda) \mathit{mod}\ p.$$

In the case when the curve **3** has complex multiplication in the imaginary quadratic field K, the trace of Frobenius or the value $H_p(\lambda)$ of the Hasse–Deuring polynomial has a variety of arithmetic interpretations. Let us look at one-class fields K. Half of the primes p are supersingular for the elliptic curve **3**—i.e., $H_p(\lambda) \equiv 0 \bmod p$. These are the primes p that stay prime in K. For other good primes p, split in K, the trace of Frobenius or $H_p(\lambda)$ is explicitly determined from the representation $4p = a^2 + Db^2$ for the discriminant D of K.

With each of the four theories of *Section 3* of hypergeometric series representations of period relations we associate congruences for values of truncated series. Congruences depend on the order of truncation: if a few consecutive coefficients in series are 0 mod M, all higher coefficients are ignored mod M. This way one builds a "p-adic" interpretation of the Ramanujan identities without changing the left-hand side (see ref. 14 for details).

We present the theory corresponding to the absolute invariant $J(\tau)$. The identity **1** can be written as

$$\sum_{n=0}^{\infty} \{c_1 + n\} \frac{(6n)!}{(3n)!n!^3} \frac{1}{J^n} = \frac{\delta_1}{\pi},$$

where $c_1 = (1 - s_2(\tau))/6$, $\delta_1 = \sqrt{-J/(d(12^3 - J))}/2$ for $\tau = (1 + \sqrt{-d})/2$, $J = J(\tau)$.

Now truncations of the ${}_3F_2$-series in $1/J$ can be appropriately determined mod p. We put

$$S_N^{(1)} \overset{\text{def}}{=} \sum_{n=0}^{N} \{c_1 + n\} \cdot \frac{(6n)!}{(3n)! \cdot n!^3} \cdot \frac{1}{J^n}.$$

THEOREM 4.2. *For all good primes* p,

$$S_N^{(1)} \equiv 0 \mathit{mod}\ p$$

for $[p/6] \leq N < p$.

Here p is good if p does not divide the denominator—i.e., $p \nmid J$ or p does not divide the denominator of c_1. In *Theorem 4.2* the bound on p can be replaced by $[p/6] \leq N \pmod p < p$.

These congruences, particularly *Theorem 4.2* for $J = -640{,}320^3$ can be used to verify schemes of computation of π, based on telescoping *Algorithm I*, or they can be used to supplement *Algorithm I* by using the Chinese remainder theorem.

Section 5. Algorithms and Implementation

For those who wish to run their own π calculation, or to do multimillion digit computations, and who have not written a bignum package before, we describe briefly nontrivial parts of the package. Whenever the word size of operands or the result exceeds the word size of the machine, additional programming is needed. Knuth (15) devotes considerable attention to bignum programming. One should have different routines for different ranges of numbers and for different storage modes. Next, one should take full advantage of any inherent parallelism of algorithms; e.g., in *Algorithm I* in the early stages, many simultaneous and independent operations take place. This way on vector machines there is always a long vectorization length (across short-length operations on early stages and inside long-length operations on later stages of *Algorithm I*).

Bignum division and other elementary and nonelementary operations are combinations of basic primitives: addition/subtraction and bignum multiplication.

Most computational time is spent on bignum multiplication. Other than for relatively short numbers, the classical method of multiplication should be avoided. A conventional remedy is the use of FFT algorithms to speed up the convolution of digits of factors from which the true digit of the result is reconstructed. It is better to speak of fast convolution algorithms rather than FFT because often the floating-point FFT is less efficient than its modular versions or new fast convolution algorithms. We refer to refs. 2 and 15 for definitions of bilinear-form representations of fast convolution algorithms. In all these algorithms one looks at the product $C = A \cdot B$ of two bignums written in the radix Rad:

$$A = \sum_{i=0}^{n-1} A_i \cdot \text{Rad}^i, \qquad B = \sum_{j=0}^{m-1} B_j \cdot \text{Rad}^j$$

as the result of convolution of arrays (A_i) and (B_j). This result of convolution $(\vec{A}) * (\vec{B}) = (\vec{C})$, $C_k = \Sigma_{i+j=k} A_i \cdot B_j$, is computed by using the bilinear-form algorithm

$$\vec{x} = H_n \cdot \vec{A}, \qquad \vec{y} = H_m \cdot \vec{B}, \qquad \vec{C} = G \cdot \vec{z}$$

for $H_n \in M_{l \times n}$, $H_m \in M_{l \times m}$, and $G \in M_{(n+m) \times l}$ for $z_\alpha = x_\alpha \cdot y_\alpha$ for $\alpha = 1, \ldots, l$. Here $l \geq n + m - 1$ is the rank of the algorithm, and the whole algorithm consists of three stages: transforming A and B, dot product of the transformed results ($z_\alpha = x_\alpha \cdot y_\alpha$), and retransformation.

In the case of FFT-like algorithms of fast convolution, matrices H_n, H_m, and G are built from primitive roots of unity: $H_n = (w_l^{ij})$, $H_m = (w_l^{ij})$, $G = (w_l^{-ij})$, and the resulting array C is the circular convolution of length l:$C_k = \Sigma_{i+j \equiv k(l)} A_i \cdot B_j$. Usually, l is chosen to be a highly composite number, In this case the complexity of performing FFT of order l is $O(l \log l)$ of primitive operations in any ring $\mathfrak{O}$ where

l is invertible and where w_l is a primitive root of unity of order l. The well-known Cooley–Tukey FFT corresponds to the case $l = 2^r$ (see ref. 15). The condition on the ring $\mathfrak{G}$ where the FFT is performed is the correct reconstruction of the result C_k.

There are three choices for $\mathfrak{G}$ in practice: (*i*) the complex number field (with the precision of operation high enough to determine C_k correctly taking into account the loss of $O(l \log l)$ digits of precision during the FFT computation), (*ii*) products of finite fields having primitive roots of unity of order l (e.g., for $l = 2^r$ finite fields $\mathbf{F}_p$ for primes p are of special form with $p = s \cdot 2^r + 1$), with integers C_k reconstructed by using the Chinese remainder theorem, and (*iii*) surrogate polynomial or special modular rings such as $\mathbf{Z}/\mathbf{Z} \cdot (2^{2^l} + 1)$. A more general approach to fast modular and integer convolution algorithms is described in ref. 2. These algorithms use arbitrary algebraic curves and varieties and represent all fast convolution algorithms as interpolation algorithms of these varieties. Conventional algorithms arise then as special cases corresponding to interpolation on a projective line or circle. For practical implementation all approaches mentioned above should be used, depending on the length of the convolution.

Once a bignum package is in place, implementation of *Algorithm I* for any hypergeometric function identity can proceed. If congruences such as those of *Section 4* are available for verification of intermediate and final results, they should be applied for "random" moduli (so that no bias can occur). If no explicit congruences are found, it is still important to have congruence checks of final results, independent on local verifications. For this we suggest running *Algorithm I* for a prescribed set of prime moduli, preferably without any use of the bignum package. The set of modular results can be used for verification of results in all stages. We used several classes of prime moduli for these local verifications, to bring any local error to a probability below 10^{-290}.

Section 6. Conclusions

The decimal (or other radix) expansion of a classical constant attracts attention because of curiosity in detecting rules or patterns hidden in the sequence of digits. The amount of data generated during the computation of π is barely enough to determine statistical laws distinguishing decimal expansions of classical numbers from random sequences and from each other.

Among other applications of the calculation of π (and other classical constants) are the determination of initial pieces of continued fraction expansions, needed to determine measures of irrationality, and diophantine approximations in the regions where analytic results are inadequate. This is needed to remove large constants from results on measures of irrationality for π, $\pi/\sqrt{3}$, $\pi/\sqrt{2}$, . . . (see ref. 7).

As of July 1989 we had computed more than 1 billion decimal digits of π.

Our own computations started in December 1988. For the computation of π we used the scheme based on Eq. 1 with the matching congruences of *Section 4*. Codes were prepared for Cray 2, IBM 3090-VF, and IBM GF11 supercomputers. The Cray 2 computations proceeded at the Minnesota Supercomputer Center (MSC) in Minneapolis. Computations on the IBM 3090 were conducted at the IBM T. J. Watson Research Center in Yorktown Heights, NY. Computations were conducted in a shared environment over a period of 8 months. All programming for the Cray 2 and IBM 3090 was in FORTRAN, and all nontrivial parts of the code were fully vectorized. The main focus during these computations was on verifiability of results and preservation of correct data on various storage devices. We are indebted to B. Rackner, who over many months fully shared the burden of this computation. Our use of IBM SCRATCHPAD was invaluable in designing algorithms and proving identities. R. Jenks and his group who developed SCRATCHPAD were a great help to us and we thank them for their support. G. Slishman of IBM devoted considerable time and effort to efficient I/O handling and utilization of channels and storage. We thank J. Parnas and M. Donner for their help. These computations were originally discussed and planned with M. Denneau of IBM, the designer of the IBM GF11 machine. We thank the National Science Foundation (NSF) for support and computer time, particularly M. Ciment, I. Lombardo, and J. Lubin. We are deeply thankful to the staff of the MSC for their support and assistance. Part of the computer time on the Cray 2 at MSC, with 4 gigabytes of physical memory, was provided by NSF.

1. Schonhage, A. & Strassen, V. (1971) *Computing* **12**, 281–292.
2. Chudnovsky D. & Chudnovsky, G. (1987) *Proc. Natl. Acad. Sci. USA* **84**, 1739–1743.
3. Salamin, E. (1976) *Math. Comput.* **30**, 565–570.
4. Brent, R. (1976) *J. Assoc. Comput. Mach.* **23**, 242–251.
5. Borwein, J., Borwein, P. & Bailey, D. (1989) *Am. Math. Month* **96**, 201–219.
6. Ramanujan, S. (1914) *Q. J. Math.* **45**, 350–372.
7. Chudnovsky, D. & Chudnovsky, G. (1988) *Computer Algebra* (Dekker, New York), pp. 1–82.
8. Chudnovsky, D. & Chudnovsky, G. (1988) *Ramanujan Revisited* (Academic, New York), pp. 375–472.
9. Chudnovsky, D. & Chudnovsky, G. (1989) *Proceedings of the Symposium on Pure Mathematics* (Am. Math. Soc., Providence, RI), Vol. 49, pp. 167–232.
10. Siegel, C. L. (1949) *Transcendental Numbers* (Princeton Univ. Press, Princeton, NJ).
11. Chudnovsky, D. & Chudnovsky, G. (1985) *Lecture Notes in Mathematics* (Springer, New York), Vol. 1135, pp. 9–51.
12. Gosper, W. (1972) Memorandum 239 (Massachusetts Inst. Technol. AI Lab., Cambridge).
13. Weil, A. (1976) *Elliptic Functions According to Eisenstein and Kronecker* (Springer, New York).
14. Chudnovsky, D. & Chudnovsky, G. (1989) *Lecture Notes in Mathematics* (Springer, New York), Vol. 1383, pp. 12–49.
15. Knuth, D. (1981) *The Art of Computer Programming* (Addison-Wesley, Reading, MA), Vol. 2.