Systematic Computations on Gauss'

Lattice Point Problem (In Commemoration of

Johannes Gualtherus van der Corput, 1890-1975)

J. van de Lune

Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam

E. Wattel

Free University
Department of Mathematics and Information Theory
De Boelelaan 1081, 1081 HV Amsterdam

0. INTRODUCTION

A point $(x,y) \in \mathbb{R}^2$ is called a *lattice point* if both its coordinates x and y are (rational) integers.

For $t \geq 0$ we denote by $P(t)$ the number of lattice points in the closed circular disc in $\mathbb{R}^2$ about the origin $(0,0)$ with radius $r = t^{1/2}$, i.e. $P(t)$ is the number of lattice points (x,y) satisfying

$$x^2 + y^2 \leq t. \tag{0.0}$$

(In the sequel n will always denote a positive integer.)

Since x and y are integral, so is $x^2 + y^2$, from which it is clear that $P(t)$ is a non-decreasing step-function which is constant on all intervals of the form $n - 1 \leq t < n$ with upshot-discontinuities only at those $t = n$ which can, in the usual number theoretical sense, be written as the sum of two squares.

In the early 1800's GAUSS [7, 8] studied the asymptotic behaviour of $P(t)$ and showed, by a simple geometrical argument, that

$$P(t) = \pi t + \mathcal{O}(t^{1/2}), \quad (t \to \infty) \tag{0.1}$$

which may be rephrased by saying that $P(t)$ equals the area of the disc save for an error of (at most) the order of its circumference.

It seems that, for quite a long time after Gauss, the study of the 'true size' of the error term in (0.1) has not attracted much attention. Not until 1906 did SIERPINSKI [34] sharpen (0.1) to

$$P(t) = \pi t + \mathcal{O}(t^{1/3}) \tag{0.2}$$

by applying an analytical method devised by VORONOÏ [37] for the analogous Dirichlet divisor problem.

In 1912 LANDAU [23, 24] also obtained (0.2) after having 'rigorized' a more or

less heuristic geometrical method of PFEIFFER [33] dating back to 1886. Since the proofs of (0.2) are still rather demanding, we mention here that VINOGRADOV [36; p. 169] has shown, in a rather elementary (though sophisticated) way, the slightly weaker result

$$P(t) = \pi t + \mathcal{O}(t^{1/3}\log(t)). \tag{0.3}$$

(By the way, is there any 'really simple' proof of (0.1) with just $\mathcal{O}$ replaced by o ?)

For many years (0.2) was the best result on the asymptotic behaviour of $P(t)$ and it seems that around the 1920's there was a more or less general concensus that (0.2) is the best possible result for Gauss' lattice point problem. Compare LANDAU [27; pp. 183-184]. Also see VAN DER CORPUT & LANDAU [3] and LANDAU [28].

Therefore, it must have come as a great surprise to the number theoretical world that, in 1923, VAN DER CORPUT [2] proved that

$$P(t) = \pi t + \mathcal{O}(t^{33/100}). \tag{0.4}$$
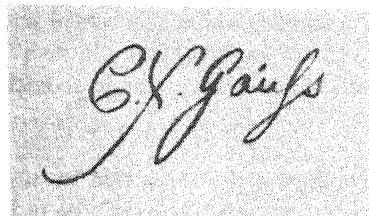
Carl Friedrich Gauss
(1777 - 1855)

(Compare the result of JARNIK [18] that for more general curves the exponent in the equivalent of (0.4) is $\geqslant 1/3$.)

Let from now on

$$E(t) := P(t) - \pi t, \quad (t \geqslant 0) \tag{0.5}$$

be the error function in Gauss' lattice point problem, and let α denote any real number for which

$$E(t) = \mathcal{O}(t^{\alpha}), \quad (t \to \infty) \tag{0.6}$$

is true. Denoting the infimum of all these α's by θ, we have the following time table for the successive results on the order of $E(t)$, i.e. the value of θ. (We must admit that, when consulting various historical accounts, we became mildly confused by the discrepancies between some cross-references.)

$\theta \leqslant$	1 / 2	= 0.5	$\pm$1830, Gauss [7, 8]
$\theta \leqslant$	1 / 3	= 0.333333...	1906, Sierpinski [34]
$\theta \leqslant$	33 / 100	= 0.33	1923, van der Corput [2]
$\theta \leqslant$	37 / 112	= 0.330357...	1924, Landau [26]
			1924, Littlewood & Walfisz [29]
$\theta \leqslant$	163 / 494	= 0.329959...	1927, Walfisz [38, 39]
$\theta \leqslant$	27 / 82	= 0.329268...	1928, Nieland [31]
$\theta \leqslant$	15 / 46	= 0.326086...	1934, Titchmarsh [35]
$\theta \leqslant$	13 / 40	= 0.325	1942, Hua Loo Keng [12]
$\theta \leqslant$	12 / 37	= 0.324324...	1962, Yin Wen Lin [41]
$\theta \leqslant$	12 / 37	= 0.324324...	1963, Chen Jing Run [1]
$\theta \leqslant$	35 / 108	= 0.324074...	1984, Nowak [32]
$\theta \leqslant$	139 / 429	= 0.324009...	1985, Kolesnik [21]
$\theta \leqslant$	7 / 22	= 0.318181...	1988, Iwaniec & Mozzochi [17]

In the opposite direction we have that

$$\theta \geqslant 1/4. \tag{0.7}$$

This was shown in 1915 by LANDAU [25], and, independently, by HARDY [9]. It occurs to us that it is worth mentioning here that, in 1956, ERDÖS & FUCHS [4] proved (0.7) in a different manner by means of a 'very general' theorem.

In 1916 HARDY [10] showed that (0.6) is false for $\alpha = 1/4$. In fact he showed that there exists a positive constant K such that

$$\frac{E(t)}{t^{1/4}} < -K\,(\log(t))^{1/4} \tag{0.8}$$

for infinitely many arbitrarily large values of t. Hence, the left-hand side of (0.8) does not have a finite lower bound as $t \to \infty$. Also see LANDAU [27; pp. 240-249].

In the opposite direction there is a (somewhat less specific) result of INGHAM [15] saying that the left-hand side of (0.8) does not have a finite upper bound as $t \to \infty$.

For further refinements of these results see FRICKER [6] and KRÄTZEL [22]. In view of the best (theoretical) results available at present, we are tempted to

believe that, for large t, the 'extremal negative deviations of $E(t)$ are somewhat larger than those in the positive direction'. Our numerical results (see Sections 4 and 6) corroborate this belief. At present it is not known whether (0.6) is false for any $\alpha > 1/4$. Besides some (minor, though difficult) refinements, we thus only know that

$$1/4 \leq \theta \leq 7/22 \tag{0.9}$$

so that we are left with some uncertainty as to the true value of θ (note that $7/22 - 1/4 = 3/44 = 0.068181818...$).

For more detailed historical accounts of the subject we refer to FRICKER [6], HUA [13; par. 45], IVIC [16], KRÄTZEL [22] and WILTON [40]. Also see H.J.A. DUPARC & J. KOREVAAR, (in Memoriam) Johannes Gualtherus van der Corput, Nieuw Archief voor Wiskunde (3) XXX (1982) pp. 1-39.

The *opinions* on the true value of θ vary. In LANDAU [27; p. 189] we read *ob $\theta > 1/4$ ist, weiss ich nicht. Ich vermute - nichts.* FRICKER [6; p. 87] states that some believe that θ is quite close to $1/3$, that others believe that $\theta = 1/4$, and that there are authors which are explicitly prudent to express any conjecture with respect to θ. HUA [14; p. 134] just mentions that $\theta = 1/4$ is a famous conjecture in number theory. HARDY seems to have been the first to state the conjecture that $\theta = 1/4$.

In 1965 KATAI [19] showed that for some $\lambda > 0$ (compare WILTON [39; par. 5])

$$\int_0^t (E(u))^2 \, du = \lambda \, t^{3/2} + \mathcal{O}(t \log^2(t)) \tag{0.10}$$

from which one may derive that

$$\frac{1}{t} \int_0^t |E(u)| du = \mathcal{O}(t^{1/4}) \tag{0.11}$$

which might be seen as an indication that $\theta = 1/4$.

Since there seems to be a lack of further 'convincing' heuristical arguments supporting any of the opinions, the question about the true value of θ remains unsettled. To this we might add that HEJHAL [11; p. 451, Remark 4.3] states that it is conceivable that the circle problem could be more difficult than the Riemann Hypothesis.

1. NUMERICAL COMPUTATIONS IN THE 1960's

In order to get an impression of the true value of θ one has computed $P(t)$ and hence $E(t)$ for several quite large values of t. The first recorded computations related to this subject may be found in GAUSS [7, 8], from which we infer that, for example, $P(100,000) = 314,197$.

To the best of our knowledge there have been only three more attempts of this kind (all in the 1960's, on IBM computers): FRASER & GOTLIEB [5], MITCHELL [30], and KELLER & SWENSON [20].

Fraser & Gotlieb evaluated $P(t)$ for $t = n^2$ where n runs as follows

$$n = 1 \ (1) \ 50 \ (5) \ 200 \ (100) \ 1000 \ (200) \ 1800 \tag{1.1}$$

and for some (unspecified) values of t in the range $1800^2 < t \leqslant 2000^2$. Their conclusion is that the conjecture that θ is 'arbitrarily close to 1/4 is not inconsistent with the observed results'.

Although we haven't had Mitchell's results in front of us, we mention here that Keller & Swenson found that his results are incorrect for all considered $t \geqslant 9,000,000$ (i.e. radii $\geqslant 3,000$).
Assuming that Mitchell's results for $t < 9,000,000$ are correct, it occurs to us that his results are just a minor extension of those of Fraser & Gotlieb. We do not know Mitchell's conclusion as to the true value of θ.

Keller & Swenson evaluated $P(t)$ for $t = n^2$ where n runs as follows

First n	Step	Last n
1	1	10,000
10,000	250	100,000
100,000	1	100,099
100,000	10,000	150,000
150,000	1	150,099
150,000	250	259,750
200,000	1	200,099
250,000	1	250,099

and for all $t = x^2$ where x runs as follows

First x	Step	Last x
100,000	1/64	100,002
150,000	1/64	150,002
200,000	1/64	200,002
250,000	1/64	250,002

The conclusions of Keller & Swenson may be summarized as follows (almost entirely in their own words): The results clearly suggest that (0.6) is valid for $\alpha = 0.35$ or even perhaps for $\alpha = 0.34$. But since (0.6) is known to be valid for all $\alpha \geqslant 0.325$ no useful quantitative estimates are obtained. However, an extrapolation of the data does suggest that a smaller order should suffice and that computations for larger values of t could indicate this. For example, to obtain a significant improvement, say $\alpha \leqslant 0.30$, a crude extrapolation implies a radius of about 10^8. Unfortunately, calculations by our method for such radii would require at least two hours per case on an IBM 7090. Thus Keller & Swenson.
From the t-listings above we infer that $P(t)$ (and hence $E(t)$) has almost exclusively been evaluated for square integral values of t. In Section 6 we will see that, sadly enough, these t's do not seem to be the most relevant ones for

the purpose they were meant to serve.

2. PRELIMINARY SYSTEMATIC EXPERIMENTS ON THE ORDER OF $E(t)$

In this section we describe some *preliminary systematic* computations concerning the order of $E(t)$.

We begin by confessing that, prior to our numerical approach to Gauss' lattice point problem, we did not have any specific opinion about the true value of θ.

We recall that n denotes a positive integer and that

$$E(t) := P(t) - \pi t, \quad (t \geq 0). \tag{2.1}$$

It is clear that $E(t)$ is a sawtooth function with the following properties.

(E1) $E(t)$ is linear on all intervals of the form $n - 1 \leq t < n$ with slope $-\pi$

(E2) $E(t)$ is continuous from the right for $t \geq 0$

(E3)   all discontinuities of $E(t)$ are 'upshot-discontinuities' at points $t = n$, where n is representable, in the usual number theoretical sense, as the sum of two squares

(E4)   all local maxima of $E(t)$ are assumed at $t = 0$ and certain points of the form $t = n$

(E5) $E(n-0) := \lim_{t \uparrow n} E(t) = E(n-1) - \pi$

(E6)   (by convenient abuse of language) all local infima of $E(t)$ are 'assumed' at certain points of the form $t = n - 0$.

(Note that $E(t)$ is completely determined by its values at the points $t = n - 1$.)

From Section 1 we know that $E(t)$ is unbounded (in the positive as well as in the negative direction). Our aim is to find a 'non trivial, though simple' real function $B(t)$ such that, for $t \geq 1$, say,

(B1) $|E(t)| \leq B(t)$

(B2) $B(t)$ is smooth, positive, and monotonically increasing.

It is clear that in order to study the 'size' of $E(t)$ we will be interested only in those t-values for which $E(t)$ is **extremal,** i.e. t-values (which, by (E4) and (E6) above, are non-negative integers) for which

(M) E has a local maximum at t and $E(u) < E(t)$ for all $u < t$

or

(I) E has a local infimum at $t - 0$ and $E(u) > E(t-0)$ for all $u < t$.

Instead, extremal values might as well be called *champion-extremes*.

Since $E(0) = 1$ and $E(1-0) = 1 - \pi$, all local maxima (infima) of $E(t)$ which

are extremal are positive (negative).

In order to get a first impression as to which $B(t)$ to choose we computed $E(n)$ (and $E(n-0) = E(n-1) - \pi$) for all $n \leqslant 100,000$. This was carried out by means of the following intentionally unsophisticated FORTRAN 2 program GAUSSEXP on an IBM PS / 2 70 386. (As a not irrelevant byproduct, such a robust program yields valuable testvalues for more sophisticated programs.)

Program GAUSSEXP

```
        PROGRAM GAUSSEXP
C    LANGUAGE: IBM FORTRAN / 2
C    COMPUTER: IBM PS / 2 70 386 ( + i 387 math coprocessor )
C    SUBJECT: EXPERIMENT on GAUSS' LATTICE POINT PROBLEM.
C    IMPRESSION of ALL EXTREMALS of E( t ) := P( t ) - pi * t, for t >= 0.
C    TABULATION of t = N , P( N ) , E( N ) , H( N ) := E( N ) / N ** (1 / 4).
        IMPLICIT DOUBLE PRECISION ( D )
        OPEN ( UNIT = 1 , FILE = 'CON' )
        OPEN ( UNIT = 6 , FILE = 'PRN' )
        WRITE ( 6 , 10 )
  10    FORMAT ( / , '                   PROGRAM GAUSSEXP ' , / ,
     $            '                   **************** ' , / )
        ROOTHALF = DSQRT( .5D0 )
        DPI = 4.D0 * DATAN( 1.D0 )
        EMAX = - 1.
        EINF = + 1.
        WRITE( 1 , 20 )
  20    FORMAT( ' INPUT LASTN in I7 Format ' )
        READ( 1 , 30 ) LASTN
  30    FORMAT( I7 )
C
C                       ..... MAIN LOOP .....
C
        DO 70 N = 0 , LASTN
        RN = FLOAT( N )
C  Note that  P( N ) = P( RN ) = P( RN + .5 )            SAFETY FIRST !
        RNPLUSH = RN + .5
        CALL GAUSS( ROOTHALF , RNPLUSH , NROFPTS )
        E = NROFPTS - DPI * RN
        IF( E .LE. EMAX ) GOTO 50
        EMAX = E
        HATMAX = EMAX / SQRT( SQRT( RN ) )
        WRITE( 6 , 40 ) N , NROFPTS , EMAX , HATMAX
  40    FORMAT( ' N = ' , I7 , '        P = ' , I7 ,
     $          '    E = ' , F8.3 , '    H = ' , F6.3 )
  50    E = E - DPI
        IF( E .GE. EINF ) GOTO 70
        EINF = E
        HATINF = EINF / SQRT( SQRT( RN + 1.) )
        WRITE( 6 , 60 ) N + 1 , NROFPTS , EINF , HATINF
  60    FORMAT( ' N = ' , I7 , '(-0)    P = ' , I7 ,
     $          '    E = ' , F8.3 , '    H = ' , F6.3 )
  70    CONTINUE
        END
C
C
        SUBROUTINE GAUSS( ROOTHALF , RNPLUSH , NROFPTS )
        R = SQRT( RNPLUSH )
        K = R * ROOTHALF
        L = R
        NROFPTS = 1 + 4 * ( L + K ** 2 )
        KPLUS1 = K + 1
        IF( KPLUS1 .GT. L ) GOTO 90
        NPTS = 0
        DO 80 I = KPLUS1 , L
        NPTS = NPTS + SQRT( RNPLUSH - I ** 2 )
  80    CONTINUE
        NROFPTS = NROFPTS + 8 * NPTS
  90    RETURN
        END
```
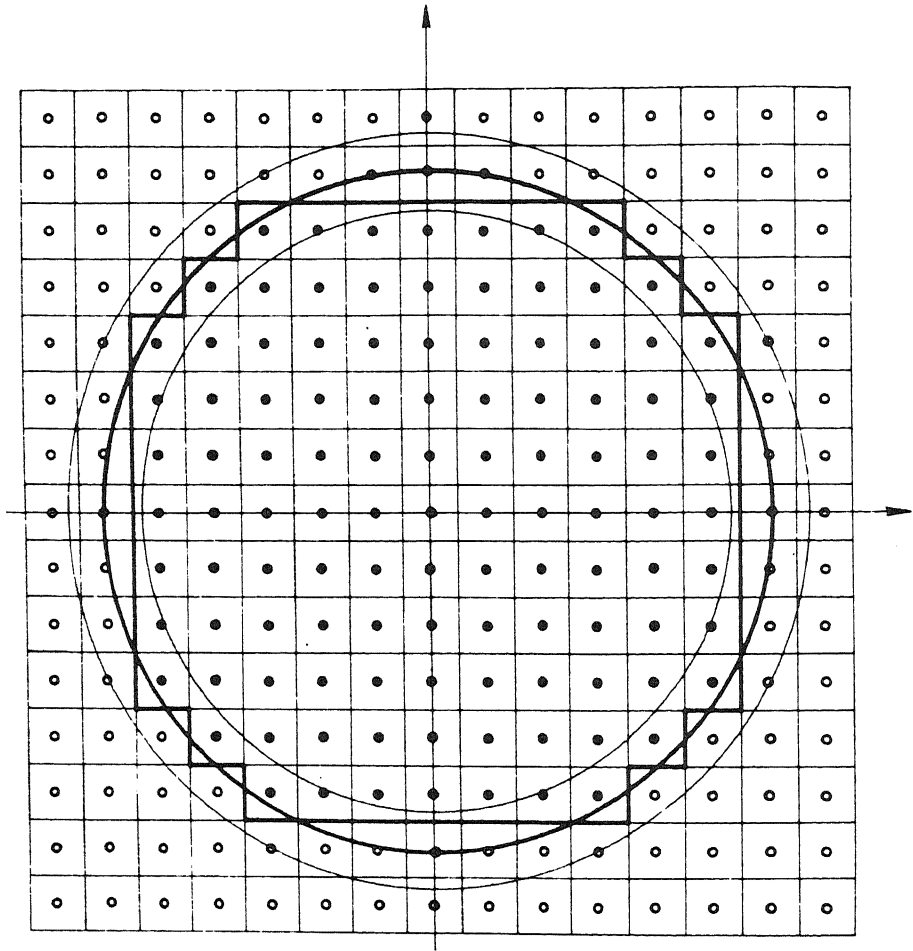
More or less out of curiosity, and to have a 'point of orientation', we also computed $H(t) := E(t)/t^{1/4}$ for all $t = n \leqslant 100.000$ for which $E(t)$ is extremal. The reader may find it interesting to know that, for $t > 0$, the local maxima and infima of $H(t)$ which are most relevant for our purpose coincide with the extremal values of $E(t)$. This may be shown by observing that $t^{1/4}$ is strictly increasing, and that, for $n - 1 < t < n$, the function $H(t)$ is differentiable, the sign of $H'(t)$ being the same as that of

$$-\pi \, t^{1/4} - \frac{1}{4} t^{-3/4} E(t) \tag{2.2}$$

which is trivially negative (for all $t > 0$, so that our claim follows. From the definition of $E(t)$ it is clear that the long and the short of the evaluation of $E(t)$ is the computation of $P(t)$. There are various techniques for computing $P(t)$ (also see Sections 5 and 7) of which we have chosen here (mainly for reasons of running speed when programmed in FORTRAN 2 and implemented on an IBM PS / 2 70 386) the following method due to Gauss (also see KELLER & SWENSON [20])

$$P(n) = 1 + 4(L + K^2) + 8S \tag{2.3}$$

where, writing $r = n^{1/2}$,

$$L = [r] \tag{2.4}$$

$$K = [r/\sqrt{2}] \tag{2.5}$$

and

$$S = \sum_{i = K + 1}^{L} [(n - i^2)^{1/2}] \tag{2.6}$$

the last sum being defined as 0 if $K + 1 > L$ (which is the case only if $n < 5$).

TABEL 1

Main results of program GAUSSEXP (for $n \leqslant 100.000$)

n	$E(n)$	$H(n)$
0	1.000	****
1(− 0)	−2.142	−2.142
1	1.858	1.858
2	2.717	2.285
4(− 0)	−3.566	−2.522
5	5.292	3.593
8(− 0)	−4.133	−2.457
10	5.584	3.140
16(− 0)	−5.265	−2.633
20	6.168	2.917

24(− 0)	− 6.398	− 2.891
25(− 0)	− 9.540	− 4.266
26	7.319	3.241
41	8.195	3.238
53	10.496	3.890
80(− 0)	− 10.327	− 3.453
97(− 0)	− 11.734	− 3.739
130	12.593	3.729
143(− 0)	− 12.248	− 3.542
144(− 0)	− 15.389	− 4.443
149	12.903	3.693
205	12.974	3.429
234	13.867	3.546
287(− 0)	− 16.637	− 4.042
288(− 0)	− 19.779	− 4.801
340	16.858	3.926
410	16.947	3.766
425	17.823	3.925
481(− 0)	− 22.106	− 4.720
586	20.027	4.070
625(− 0)	− 22.495	− 4.499
841(− 0)	− 25.079	− 4.657
850	22.646	4.194
986	27.390	4.888
1152(− 0)	− 26.115	− 4.483
1444(− 0)	− 27.460	− 4.455
1508(− 0)	− 28.522	− 4.577
1680(− 0)	− 28.876	− 4.510
1681(− 0)	− 32.017	− 5.000
1700	28.292	4.406
1844(− 0)	− 32.097	− 4.898
2260	29.001	4.206
2592(− 0)	− 34.008	− 4.766
3024(− 0)	− 35.176	− 4.744
3025(− 0)	− 38.318	− 5.167
3146	33.550	4.480
3400	35.585	4.660
3960(− 0)	− 39.707	− 5.005
3961(− 0)	− 42.849	− 5.401
5183(− 0)	− 45.875	− 5.407
5184(− 0)	− 49.016	− 5.777
5525	43.701	5.069
7921(− 0)	− 51.555	− 5.465
9701	48.410	4.878
9797(− 0)	− 53.183	− 5.346
11234(− 0)	− 55.652	− 5.406

14884(− 0)	−58.465	−5.293
15120(− 0)	−59.881	−5.400
15121(− 0)	−63.023	−5.683
17225	59.067	5.156
19594(− 0)	−63.366	−5.356
21601(− 0)	−64.543	−5.324
21604(− 0)	−65.968	−5.441
21605(− 0)	−69.109	−5.700
22178(− 0)	−69.242	−5.674
28560(− 0)	−70.886	−5.453
28561(− 0)	−74.028	−5.694
31680(− 0)	−76.655	−5.746
31681(− 0)	−79.797	−5.981
32850	67.681	5.027
38016(− 0)	−81.786	−5.857
38017(− 0)	−84.928	−6.082
38018(− 0)	−88.070	−6.307
40321(− 0)	−91.157	−6.433
45994	70.587	4.820
46330	71.012	4.840
52417(− 0)	−91.862	−6.071
52418(− 0)	−95.004	−6.279
58081(− 0)	−97.843	−6.303
61685	71.857	4.560
64181	74.442	4.677
69290	76.045	4.687
80642(− 0)	−99.315	−5.894
83753	79.190	4.655
85264(− 0)	−99.756	−5.838
95459(− 0)	−100.293	−5.706
95460(− 0)	−103.435	−5.885
95461(− 0)	−106.576	−6.063
95464(− 0)	−108.001	−6.144

From this table we get the impression that the relevant values of $|H(t)|$ behave more or less proportional to $\log(t)$. This suggests that we should try, for example,

$$B(t) := 1 + t^{1/4}\log(t), \quad (t \geqslant 1). \tag{2.7}$$

We define

$$Q(t) := \frac{E(t)}{B(t)} = \frac{P(t) - \pi t}{1 + t^{1/4}\log(t)}, \quad (t \geqslant 1) \tag{2.8}$$

and hope that we will obtain a clear indication for boundedness of $Q(t)$.

3. SYSTEMATIC COMPUTATION OF THE EXTREMAL VALUES OF $E(t)$

First observe that the local maxima and infima of $Q(t)$ coincide with those of $E(t)$. This may be shown by considering the derivative of $Q(t)$ for $n < t < n+1$, the sign of which is the same as that of

$$-\pi \,(1+t^{1/4}\log(t))-(\frac{1}{4}t^{-3/4}\log(t)+t^{-3/4})(P(t)-\pi \,t). \qquad (3.1)$$

Since (3.1) is negative if and only if

$$P(t) > \pi \,t \,\frac{4-3\log(t)-4t^{-1/4}}{4+\log(t)} \qquad (3.2)$$

our claim follows since it is a matter of routine to show that

$$4-3\log(t)-4t^{-1/4} < 0, \quad (t>1) \qquad (3.3)$$

whereas $P(t) > 0$.

Hence, since $B(t)$ is monotonically increasing, we need only determine those t's for which $E(t)$ is extremal. From Table 1 we infer that it is to be expected that these values of $E(t)$ occur only very rarely, so that we should try and find a procedure for finding the extremal values of $E(t)$ which avoids 'as many evaluations of $E(t)$ as possible' (note the large 'gaps' in Table 1).

We now give an outline of how this can be achieved. For further details we refer to the full program listing in Section 5.

Suppose that we already know all extremal values of $E(t)$ for $t < n$ and that we have just evaluated $E(n)$. Since $E(t)$ is a rather trivial function for small t, we might, for example, take $n = 1$. Let *Einf* and *Emax* be the most extreme values of $E(t)$ for $t < n+1$. Also assume that the next extremal infimum of $E(t)$ is assumed at $t = n+m(-0)$. (For the moment we do not worry about possible new positive extremal values of $E(t)$.)

Then m is a positive integer satisfying

$$E(n+m-0) < Einf. \qquad (3.4)$$

Since $E(n+1-0) = E(n)-\pi$, we avoid trivialities by assuming $m > 1$.

Since $P(t)$ is non-decreasing we have

$$P(n+v) \geqslant P(v), \quad (v \geqslant 0) \qquad (3.5)$$

whereas

$$P(n+m-0) = P(n+m-1). \qquad (3.6)$$

Hence, m must satisfy

$$Einf > E(n+m-0) = P(n+m-0)-\pi \,(n+m) = \qquad (3.7)$$

$$= P(n+m-1)-\pi \,(n+m) \geqslant P(n)-\pi \,(n+m)$$

or, equivalently,

$$m > \frac{P(n)-\pi \,n-Einf}{\pi} = \frac{E(n)-Einf}{\pi}. \qquad (3.8)$$

We set

$$Exces := E(n) - Einf \tag{3.9}$$

and, in order to avoid trivialities, we assume that $Exces > 0$.
Then, m being an integer, condition (3.8) may also be written as

$$m \geqslant [\frac{Exces}{\pi}] + 1. \tag{3.10}$$

This may also be seen immediately from the graph of $E(t)$ by observing that, when jumping from $t = n$ to $t = n + m$, $E(t)$ cannot decrease more than $m \pi$. More formally this reads

$$E(n + v) \geqslant E(n) - \pi v, \quad (v > 0). \tag{3.11}$$

Hence, if the next evaluation of $E(t)$ actually takes place at $t = n + j$ with

$$j := [\frac{Exces}{\pi}] + 1 \tag{3.12}$$

then we have not rushed past the next low extremal value of $E(t)$. In other words, we can afford a jump of size j. This (rather trivial) fact is, in essence, the central point of our story!

Immediately after the evaluation of $E(t)$ at $t = n + j$ we check whether we have to adjust $Einf$ and record the result accordingly. Note that, even if $E(n + j) > Emax$, we do not know (yet) whether $E(n + j)$ is the first new positive extremal value of $E(t)$.

In practice, (3.12) turns out to be very efficient. Experiments show that a 'random evaluation' of $E(n)$ usually yields a value quite close to 0 so that, based upon (3.12), we can make a reasonably accurate prediction as to the (average) size of the jump j.

For $n = 10^9$, for example, we find $Einf < -1,630$ so that from here on j is, on average, (at least) about 518. For $n = 10^{10}$ we found that $Einf < -2,597$ so that from here on j is, on average, (at least) about 826, a figure which is comparable to a total speed-up factor corresponding to three noteworthy hardware innovations (measured by present day standards). Clearly, the more the computations proceed, the larger the jump j will be (on average).

At first sight it may seem disappointing that the procedure sketched above has the drawback (the more so when j is large) of not yielding any new information about $Emax$. However, things are not as bad as they seem to be. For convenience let us write $k = n + j$ and assume that there is an extremal maximum of $E(t)$ at $k - i$ (strictly) between n and k. Then we must have

$$E(k - i) > Emax. \tag{3.13}$$

Since, by (3.11),

$$E(k) = E((k - i) + i) \geqslant E(k - i) - \pi i \tag{3.14}$$

it follows from (3.13) that

$$E(k)+\pi i > Emax \tag{3.15}$$

or, equivalently,

$$i > \frac{Emax - E(k)}{\pi}. \tag{3.16}$$

Writing

$$Defect := Emax - E(k) \tag{3.17}$$

it follows that i, being an integer, must satisfy

$$i \geqslant [\frac{Defect}{\pi}]+1. \tag{3.18}$$

It should be clear that we apply this analysis only if $Defect > 0$ (the case $Defect \leqslant 0$ being a rather trivial situation).

Hence, if possible at all, when making a backstep i of the size of the right-hand side of (3.18), we are sure (this time argueing from the right to the left) that we do not rush past the sought most extremal maximum of $E(t)$ between n and k.

If necessary, we save the relevant data concerning $E(k-i)$, and repeat this backstep-procedure a number of times in order to find all extremal maxima of $E(t)$ for $n < t \leqslant k$. For more details we refer to subroutine CHECKMAX in the full listing of program GAUSSXTR in Section 5. In practice it turns out that, usually, we need only perform one such backstep (before passing the 'barrier' $t = n$). Similarly as before, dropping $E(k)$ from (3.16) one can make a fairly accurate prediction as to the average size of one backstep.

4. NUMERICAL TABLES FOR (SOME OF) THE MOST RELEVANT EXTREMAL VALUES OF $E(t)$

In this section we present tables of (some of) the most significant t-values for which $E(t)$ is extremal, together with the corresponding values of $Q(t)$.

TABEL 2

Listing of *almost all* extremal values of $E(t)$ for $t \leqslant 25,000,000$.

t	$E(t)$	$Q(t)$
0	1.000	****
1(− 0)	−2.142	−2.142
1	1.858	1.858
2	2.717	1.489
4(− 0)	−3.566	−1.205
5	5.292	1.553
8(− 0)	−4.133	−0.919
10	5.584	1.096
16(− 0)	−5.265	−0.804
20	6.168	0.841

25(− 0)	− 9.540	− 1.164
26	7.319	0.876
41	8.195	0.788
53	10.496	0.896
80(− 0)	− 10.327	− 0.732
97(− 0)	− 11.734	− 0.764
130	12.593	0.722
144(− 0)	− 15.389	− 0.845
149	12.903	0.698
205	12.974	0.614
234	13.867	0.621
288(− 0)	− 19.779	− 0.813
340	16.859	0.648
410	16.947	0.604
425	17.823	0.626
481(− 0)	− 22.106	− 0.739
586	20.027	0.619
625(− 0)	− 22.495	− 0.678
841(− 0)	− 25.079	− 0.673
850	22.646	0.605
986	27.390	0.691
1,152(− 0)	− 26.115	− 0.621
1,444(− 0)	− 27.460	− 0.599
1,508(− 0)	− 28.522	− 0.612
1,681(− 0)	− 32.017	− 0.659
1,700	28.292	0.580
1,844(− 0)	− 32.097	− 0.638
2,260	29.001	0.535
2,592(− 0)	− 34.008	− 0.596
3.025(− 0)	− 38.318	− 0.634
3,146	33.550	0.547
3,400	35.585	0.564
3,961(− 0)	− 42.849	− 0.642
5,184(− 0)	− 49.016	− 0.666
5,525	43.701	0.580
7,921(− 0)	− 51.555	− 0.602
9,701	48.410	0.526
9,797(− 0)	− 53.183	− 0.575
11,234(− 0)	− 55.652	− 0.574
14,884(− 0)	− 58.465	− 0.546
15,121(− 0)	− 63.023	− 0.585
17,225	59.067	0.524
19,594(− 0)	− 63.366	− 0.537
21,605(− 0)	− 69.109	− 0.566
22,178(− 0)	− 69.242	− 0.562
28,561(− 0)	− 74.028	− 0.551

31,681(− 0)	− 79.797	− 0.573
32,850	67.681	0.480
38,018(− 0)	− 88.070	− 0.594
40,321(− 0)	− 91.157	− 0.603
45,994	70.587	0.446
46,330	71.012	0.448
52,418(− 0)	− 95.004	− 0.574
58,081(− 0)	− 97.843	− 0.571
61,685	71.857	0.411
64,181	74.442	0.420
69,290	76.045	0.418
80,642(− 0)	− 99.315	− 0.519
83,753	79.190	0.409
85,264(− 0)	− 99.756	− 0.512
95,464(− 0)	− 108.001	− 0.533
100,053	79.230	0.385
100,058	79.522	0.386
103,241	83.833	0.403
106,250(− 0)	− 117.219	− 0.558
114,244(− 0)	− 123.111	− 0.572
121,252	84.608	0.386
136,490	85.019	0.372
138,581	95.948	0.418
147,652	102.562	0.438
178,345	105.658	0.424
186,624(− 0)	− 135.587	− 0.535
201,601(− 0)	− 151.221	− 0.582
233,546	110.602	0.406
316,205	113.695	0.377
330,986	117.814	0.385
361,201(− 0)	− 161.408	− 0.513
382,330	119.881	0.374
393,121(− 0)	− 165.046	− 0.510
418,609(− 0)	− 165.959	− 0.503
449,530	120.854	0.358
457,317	137.272	0.404
459,649(− 0)	− 172.922	− 0.508
574,544(− 0)	− 182.210	− 0.498
574,561(− 0)	− 203.617	− 0.556
574,925	140.844	0.385
574,930	149.136	0.407
776,161(− 0)	− 212.696	− 0.527
776,529	159.198	0.394
776,533	162.632	0.403
905,130	167.242	0.394
1,067,625	170.143	0.380

1,121,473(− 0)	−218.338	−0.481
1,149,122(− 0)	−256.233	−0.560
1,444,801(− 0)	−263.208	−0.534
1,515,940	171.033	0.342
1,528,250	182.027	0.363
1,764,425	188.382	0.359
1,860,490(− 0)	−280.716	−0.526
2,298,244(− 0)	−297.467	−0.521
2,372,905	208.084	0.361
2,694,250	208.993	0.348
3,104,644(− 0)	−325.782	−0.518
3,520,525	217.523	0.333
3,565,393	224.544	0.342
3,710,538	242.078	0.364
3,823,306	254.958	0.380
4,112,656(− 0)	−336.876	−0.490
4,523,905(− 0)	−341.714	−0.483
4,700,194(− 0)	−341.941	−0.477
5,945,122	257.400	0.334
6,350,400(− 0)	−376.987	−0.479
7,050,325	271.775	0.334
7,441,954(− 0)	−423.015	−0.511
7,602,208	280.196	0.336
7,646,609	318.341	0.381
14,310,920(− 0)	−424.138	−0.418
15,124,082(− 0)	−443.904	−0.430
15,657,472(− 0)	−474.009	−0.454
15,675,545	367.987	0.353
19,368,610(− 0)	−477.886	−0.429
19,368,720(− 0)	−511.461	−0.459
19,547,730	368.038	0.329
24,091,652	372.064	0.312

TABEL 3

A *selection* of extremal values of $E(t)$ for
$25,000,000 \leqslant t \leqslant 60,000,000,000$.

t	$E(t)$	$Q(t)$
26,666,613(− 0)	−514.497	−0.418
27,320,785(− 0)	−532.446	−0.430
29,953,729(− 0)	−561.974	−0.441
30,727,658	432.365	0.337

34,307,381	480.886	0.362
35,069,800(− 0)	− 617.043	− 0.461
40,589,641(− 0)	− 660.977	− 0.472
50,822,641(− 0)	− 674.602	− 0.450
62,204,769(− 0)	− 692.309	− 0.434
67,737,269	490.335	0.300
70,543,201(− 0)	− 737.022	− 0.445
74,192,186	506.509	0.301
85,090,865	516.628	0.294
87,522,761	522.021	0.295
88,328,308	529.483	0.298
91,814,696(− 0)	− 765.445	− 0.426
98,748,329	532.059	0.290
99,605,300	543.261	0.290
101,351,189	550.205	0.297
104,386,801(− 0)	− 814.153	− 0.436
113,005,576	585.624	0.306
121,396,324(− 0)	− 826.651	− 0.423
122,522,401(− 0)	− 841.882	− 0.429
133,055,770(− 0)	− 1,004.550	− 0.500
144,198,053	597.033	0.290
153,494,125	597.531	0.285
158,566,570	661.583	0.312
218,300,480(− 0)	− 1,035.243	− 0.444
233,523,725	662.101	0.278
235,606,625	662.763	0.277
243,062,921	674.026	0.280
253,955,521(− 0)	− 1,058.112	− 0.433
256,616,225	681.748	0.278
257,133,029	682.098	0.278
260,467,205(− 0)	− 1,108.729	− 0.450
269,336,500	715.256	0.288
278,933,890	733.339	0.292
302,074,037	757.521	0.294
311,875,200(− 0)	− 1,228.157	− 0.473
390,664,970	789.233	0.284
446,537,725	792.589	0.274
491,388,368	798.032	0.268
496,731,610	848.218	0.284
541,632,249(− 0)	− 1,341.406	− 0.437
585,410,810	864.972	0.275
606,981,145(− 0)	− 1,385.000	− 0.436
650,785,330	893.208	0.275
666,283,418	893.803	0.274
678,784,770	910.199	0.277
702,687,466	932.045	0.281

766,791,440(− 0)	− 1,425.740	− 0.419
767,469,385	947.229	0.278
815,575,834	972.460	0.280
844,192,753(− 0)	− 1,518.039	− 0.433
933,851,529(− 0)	− 1,630.050	− 0.451
968,670,820	997.141	0.273
1,092,520,490	1,043.720	0.276
1,283,376,866	1,046.987	0.264
1,288,411,133	1,054.764	0.265
1,329,859,250	1,074.892	0.268
1,354,041,098	1,186.865	0.294
1,402,274,381(− 0)	− 1,660.667	− 0.407
1,761,564,481(− 0)	− 1,691.334	− 0.388
1,879,394,212(− 0)	− 1,708.618	− 0.384
1,931,450,706	1,188.260	0.265
2,008,153,888	1,215.181	0.268
2,041,411,684(− 0)	− 1,848.407	− 0.406
2,149,665,572	1,244.330	0.265
2,406,612,881	1,266.016	0.265
2,644,046,005	1,267.939	0.258
2,736,440,713(− 0)	− 1,875.945	− 0.377
2,772,443,605(− 0)	− 2,076.960	− 0.416
2,959,608,836	1,320.323	0.260
2,968,668,450	1,335.536	0.262
3,118,999,104(− 0)	− 2,114.680	− 0.409
3,397,875,892	1,368.883	0.258
3,437,151,605	1,369.458	0.258
3,693,897,829	1,378.302	0.254
3,741,923,026	1,380.220	0.253
3,760,264,705(− 0)	− 2,147.781	− 0.393
3,844,899,194	1,387.336	0.252
3,849,617,125	1,437.967	0.262
3,900,164,266	1,499.141	0.272
4,082,823,361(− 0)	− 2,351.822	− 0.420
4,474,983,258	1,508.730	0.263
4,723,459,364	1,515.528	0.260
5,725,069,549	1,549.571	0.249
5,792,465,045	1,737.452	0.280
6,237,998,005(− 0)	− 2,372.616	− 0.374
6,334,876,800(− 0)	− 2,471.276	− 0.388
7,595,644,004(− 0)	− 2,489.249	− 0.370
7,760,245,844	1,795.439	0.266
7,893,075,529(− 0)	− 2,563.135	− 0.377
8,385,247,125(− 0)	− 2,597.435	− 0.376
8,790,245,320	1,852.436	0.264
9,318,871,985	1,881.180	0.264

9,787,590,649	1,933.757	0.267
10,190,557,141	1,954.847	0.267
10,672,749,305(− 0)	− 2,625.194	− 0.354
10,736,689,625(− 0)	− 2,884.774	− 0.388
12,569,997,245(− 0)	− 2,911.536	− 0.374
13,401,828,392	2,016.022	0.254
15,191,288,109(− 0)	− 2,952.800	− 0.359
15,606,496,945(− 0)	− 2,981.684	− 0.359
15,743,054,441	2,024.090	0.243
15,803,589,556	2,079.522	0.250
16,015,227,425	2,081.050	0.249
17,287,834,888	2,116.386	0.248
17,713,744,117(− 0)	− 3,008.537	− 0.349
18,461,195,545(− 0)	− 3,291.657	− 0.378
19,644,066,922	2,124.219	0.239
20,519,211,530	2,176.897	0.242
22,965,971,474	2,211.729	0.238
23,705,632,197	2,282.203	0.243
24,226,688,548	2,449.795	0.260
24,436,957,514	2,478.932	0.262
25,139,994,610(− 0)	− 3,329.063	− 0.349
27,658,343,752(− 0)	− 3,428.744	− 0.350
28,061,275,133(− 0)	− 3,519.195	− 0.357
29,818,037,041(− 0)	− 3,607.474	− 0.360
29,853,030,425	2,562.427	0.256
31,580,944,265	2,584.647	0.254
31,650,524,809(− 0)	− 3,681.216	− 0.361
34,788,692,045	2,740.430	0.261
35,756,257,745	2,813.445	0.266
36,922,391,089(− 0)	− 3,965.172	− 0.372
40,029,315,730	2,831.406	0.259
44,860,907,521(− 0)	− 4,068.345	− 0.360
47,463,413,689(− 0)	− 4,350.656	− 0.379
52,360,419,101(− 0)	− 4,389.584	− 0.372
55,939,956,749	2,836.211	0.236
56,651,262,026	2,863.528	0.237
58,956,361,256	3,136.765	0.257

In view of these tables the conjecture

$$E(t) = \mathcal{O}(t^{1/4}\log(t))$$

(4.1)

seems to be very plausible. They also indicate that the $\mathcal{O}$-constant in (4.1) is at most 1.

Moreover, the data do not exclude the possibility that even

$$E(t) = o(t^{1/4}\log(t)).$$ (4.2)

More extensive, though less systematic, computations in Section 6 support (4.1) (and (4.2)).

Therefore, it is tempting to replace the original conjecture (4.1) by (4.2).

5. LISTING OF THE MAIN PROGRAM

In this section we present the full listing of a Quick Basic version of the *various programs* that we have actually implemented.

Program GAUSSXTR

```
10   ' Program GAUSSXTR ( in Micro Soft Quick Basic ( QB or QB87 ) )
20   ' Designed for IBM PS / 2 70 386 ( + i 387 math coprocessor )
30   ' FORTRAN versions run CONSIDERABLY FASTER
40   ' EXHAUSTIVE search for ALL EXTREMAL VALUES of E(n) := P(n) - pi * n
50   '                   *******************
60   ' METHOD : " Maximal Slope Principle "
70   ' TABULATION of : n , E , Q = E( n ) / ( 1 + n ^ ( 1 / 4) * LOG( n ) )
80   ' We also print the maxima of D( n ) if E( n - 0 ) is EXTREMAL
90   LPRINT
100  LPRINT " PROGRAM GAUSSXTR ( GAUSS - ROOT - METHOD ) IN DOUBLE PRECISION "
110  LPRINT " ******************************************************************** "
120  LPRINT
130  '
140  '                           ........................
150  '                              INITIALIZATIONS
160  '                           ........................
170  '
180  DIM PRIME1( 200 ) , PRIME3( 200 ) , TEMPMAX#( 200 ) , TEMPN#( 200 )
190  ' TEMPMAX#( . ) and TEMPN#( . ) are used in SUBROUTINE CHECKMAX
200  NPR1% =  100                     ' Number of PRIMES of the form 4k + 1
210  NPR3% =  100                     ' Number of PRIMES of the form 4k + 3
220  ' !      NPR1% and NPR3% may also be REGULATED DYNAMICALLY       !
230  INPUT " INPUT OLDN# >= 1" ; OLDN#          ' From previous output
240  IF OLDN# < 1# THEN GOTO 230 ELSE OLDN# = INT( OLDN# )
250  INPUT " INFIMUM# = " ; INFIMUM#           ' From previous output
260  INPUT " MAXIMUM# = " ; MAXIMUM#           ' From previous output
270  INPUT " PRINTCYCLE " ; PRCY%              ' PRCY% = 10,000 is OK
280  NATINF# = OLDN#
290  NATMAX# = OLDN#
300  INPUT " INPUT previous MAXIMUM of D( n ) "; DMAXIMUM
310  '
320  '                           ....................
330  '                           END of INITIALIZATIONS
340  '                           ....................
```

```
350  '
360  CALL MAKEPR1( NPR1% )
370  CALL MAKEPR3( NPR3% )
380  PI# = 4# * ATN( 1# )
390  EPS# = .5# ^ 16
400  PIEPS# = PI# + EPS#
410  PIEPSINV# = 1# / PIEPS#
420  LPRINT: LPRINT " We START with OLDN = "; OLDN#
430  LPRINT: LPRINT " We TRY to DEFEAT the INFIMUM = "; INFIMUM#
440  LPRINT: LPRINT " We TRY to DEFEAT the MAXIMUM = "; MAXIMUM#
450  LPRINT
460  CALL GAUSS( OLDN# , P# )
470  E# = P# - PI# * OLDN#
480  IF E# - PI# > INFIMUM# AND E# < MAXIMUM# GOTO 540
490  LPRINT
500  LPRINT "                    !  .....  WARNING  .....  !"
510  LPRINT
520  LPRINT " OLDN = "; OLDN#; " E - PI = "; E# - PI#; " E = "; E#
530  LPRINT
540  T0= TIMER
550  FIRSTN# = OLDN#
560  '                    START of the PRINT - LOOP
570  FOR I% = 1 TO PRCY%
580  JUMP# = 1# + INT( ( E# - INFIMUM# ) / PIEPS# )
590  IF JUMP# < 1# THEN JUMP# = 1#
600  NEWN# = OLDN# + JUMP#
610  ' We first determine  NPOINTS#  for NEWN#
620  CALL GAUSS( NEWN# , NPOINTS# )
630  E# = NPOINTS# - PI# * NEWN#
640  EXCES# = E# - INFIMUM#
650  IF EXCES# < 0 GOTO 700        ' ! .....  NEW INFIMUM IN THE MAKE
660  ' We now CHECK for new MAXIMA in [ OLDN# , NEWN# ]
670  CALL CHECKMAX( OLDN# , NEWN# , MAXIMUM# , E# )
680  OLDN# = NEWN#                 ' Note that now E# = E#( OLDN# )
690  GOTO 970
700  ' We ARE GOING TO FIND a NEW INFIMUM  !   ...............
710       INFIMUM# = E#
720       NATINF# = NEWN#
730       NEXTN# = NEWN# + 1#
740       CALL SALTUS( NEXTN# , UPSHOT )
750  ' UPSHOT is either 0 or 1
760       IF UPSHOT > 0 GOTO 830                ' EXTREMAL INFIMUM FOUND
770  ' If UPSHOT < 1 we need NOT EVALUATE at NEWN#
780       NEWN# = NEXTN#
790       NATINF# = NATINF# + 1#
800       INFIMUM# = INFIMUM# - PI#
810       E# = INFIMUM#
820       GOTO 730
830  NAUX# = NATINF# + 1#
840  L# = LOG( NAUX# )
850  QATINF = ( ( INFIMUM# - PI# ) / ( 1# + SQR( SQR( NAUX# ) ) * L# ) )
860  LPRINT " (-) N = "; NAUX#;
870  LPRINT "   E = "; CSNG( INFIMUM# - PI# ); "   Q = "; QATINF;
880  D = ( (1 + INT( SQR(NAUX# - .5#) )) ^ 2 - NAUX# ) / SQR( SQR( NAUX# ) )
890  LPRINT "   D = "; D
900  IF D <= DMAXIMUM THEN GOTO 950 ELSE DMAXIMUM = D
910  LPRINT: LPRINT " N = "; NAUX#, " DMAXIMUM = "; DMAXIMUM
920  ROOT = SQR( NAUX# ): FRACROOT = ROOT - INT( ROOT )
930  LPRINT " Fractional part of SQR("; NAUX#; ") = "; FRACROOT
```

```
940 LPRINT
950 CALL CHECKMAX ( OLDN# , NAUX# , MAXIMUM# , E# )
960 OLDN# = NATINF#
970 NEXT I%
980 '                         END of PRINT - LOOP
990 PROGRESS# = OLDN# - FIRSTN#
1000 REPORT% = REPORT% + 1
1010 LPRINT: LPRINT " ( REPORT "; REPORT%; ", PRCY = "; PRCY%; ")"
1020 LPRINT " CONTINUE with OLDN = "; OLDN#
1030 LPRINT " Extremes : "; CSNG( INFIMUM# - PI# ); " and "; CSNG( MAXIMUM# ),
1040 LPRINT " PROGRESS PER SEC. ="; INT( .5# + PROGRESS# / ( TIMER - T0 ) )
1050 LPRINT
1060 GOTO 540              ' BACK TO PRINT - LOOP
1070 '
1080 '
1090 SUB GAUSS( N# , NPOINTS# ) STATIC
1100 ' THE SQUARES MAY ALSO BE PRECOMPUTED AND STORED IN SQUARE( . ), SAY
1110 ' THE NEXT PRINT SLOWS THE PROCESS DOWN !    ONE MAY JUST DELETE IT
1120 ' PRINT N#,          ' THIS SHOWS THE ZIG - ZAGGING (ON THE SCREEN !)
1130 NPLUSH# = N# + .5#
1140 K# = INT( SQR( NPLUSH# / 2# ) )
1150 L# = INT( SQR( NPLUSH# ) )
1160 NPOINTS# = 1 + 4# * ( L# + K# ^ 2 )
1170 KPLUS1# = K# + 1#
1180 IF KPLUS1# > L# GOTO 1260        ' Only necessary for SMALL N#
1190 P# = 0#
1200 ' We apply NO TRICKS here ! They are, in general, too machine dependent.
1210 ' If FAST memory permits, PRECOMPUTE the SQUARES I#^2 for the next loop
1220 FOR I# = KPLUS1# TO L#
1230 P# = P# + INT( SQR( NPLUSH# - I# ^ 2 ) )
1240 NEXT I#
1250 NPOINTS# = NPOINTS# + 8# * P#
1260 END SUB
1270 '
1280 '
1290 SUB SALTUS( NUMBER# , UPSHOT ) STATIC
1300 SHARED NPR1% , NPR3% , PRIME1( ) , PRIME3( )
1310 INTEGER# = NUMBER#
1320 ' We first REMOVE ALL PRIMES 4K+1 from INTEGER#
1330 CALL CLEAN( INTEGER# , NPR1% )
1340 IF INTEGER# < 2# GOTO 1580
1350 IP1# = INTEGER# + 1#
1360 Q# = INT( IP1# / 4 )
1370 IF Q# * 4# <> IP1# GOTO 1400
1380 UPSHOT = 0                       ' NO UPSHOT; INTEGER = 4k - 1
1390 GOTO 1590
1400 FOR I% = 1 TO NPR3%
1410 POWER% = 0
1420 P# = PRIME3( I% )
1430 Q# = INT( INTEGER# / P# )
1440 IF P# * Q# <> INTEGER# GOTO 1480
1450 POWER% = POWER% + 1
1460 INTEGER# = Q#                    ' TRY to find a HIGHER POWER of P#
1470 GOTO 1430
1480 IF 2 * INT( POWER% / 2 ) = POWER% GOTO 1510
1490 UPSHOT = 0
1500 GOTO 1590
1510 IF INTEGER# < P# GOTO 1580
1520 NEXT I%
```

```
1530 IP1# = INTEGER# + 1#
1540 Q# = INT( IP1# / 4# )
1550 IF Q# * 4# <> IP1# GOTO 1580    ' INTEGER# is NOT of the FORM 4k - 1
1560 UPSHOT = 0
1570 GOTO 1590
1580 UPSHOT = 1                      ' This "GUESS" is SAFE !
1590 END SUB
1600 '
1610 '
1620 SUB MAKEPR3( NPR3% ) STATIC
1630 SHARED PRIME3( )
1640 PRIME3( 1 ) = 3
1650 N = 3
1660 FOR K% = 2 TO NPR3%
1670 N = N + 4
1680 MAXD = INT( SQR( N + .5 ) )
1690 D = 1
1700 D = D + 2
1710 IF D <= MAXD GOTO 1740
1720 PRIME3( K% ) = N
1730 GOTO 1770
1740 Q = INT( N / D )
1750 IF D * Q <> N GOTO 1700
1760 GOTO 1670
1770 NEXT K%
1780 ' LPRINT
1790 ' LPRINT " PRIME3("; NPR3%; ") = "; PRIME3( NPR3% )
1800 END SUB
1810 '
1820 '
1830 SUB MAKEPR1( NPR1% ) STATIC
1840 SHARED PRIME1( )
1850 PRIME1( 1 ) = 2
1860 N = 1
1870 FOR K% = 2 TO NPR1%
1880 N = N + 4
1890 MAXD = INT( SQR( N + .5 ) )
1900 D = 1
1910 D = D + 2
1920 IF D <= MAXD GOTO 1950
1930 PRIME1( K% ) = N
1940 GOTO 1980
1950 Q = INT( N / D )
1960 IF D * Q <> N GOTO 1910
1970 GOTO 1880
1980 NEXT K%
1990 ' LPRINT
2000 ' LPRINT " PRIME1("; NPR1%; ") = "; PRIME1( NPR1% )
2010 END SUB
2020 '
2030 '
2040 SUB CLEAN( I# , NPR1% ) STATIC
2050 ' We REMOVE "ALL" PRIMES of the form 4k + 1 FROM I#
2060 SHARED PRIME1( )
2070 FOR K% = 1 TO NPR1%
2080 PK# = PRIME1( K% )
2090 Q# = INT( I# / PK# )
2100 IF Q# * PK# <> I# GOTO 2130
2110 I# = Q#
2120 GOTO 2090
2130 IF I# < 2# GOTO 2150
2140 NEXT K%
2150 END SUB
2160 '
2170 '
```

```
2180 SUB CHECKMAX( OLDN# , NEWN# , MAXIMUM# , E# ) STATIC
2190 ' E# HAS ALREADY BEEN EVALUATED AT NEWN#.     E# = E#( NEWN# )
2200 SHARED INFIMUM# , PI# , PIEPS# , NPOINTS# , TEMPMAX#( ) , TEMPN#( )
2210 AUXNEWN# = NEWN#
2220 KOUNT = 0
2230 AUXE# = E#                          ' INPUT E# MUST BE SAVED
2240 IF AUXE# < MAXIMUM#  GOTO 2300      ' ( <  for "SAFETY FIRST" )
2250 KOUNT = KOUNT + 1                   ' THE MAXIMUM IS DEFEATED
2260 ' We STORE the DATA corresponding to this (TEMPORARY) MAXIMUM
2270 TEMPN#( KOUNT ) = AUXNEWN#
2280 TEMPMAX#( KOUNT ) = AUXE#
2290 ' We MIGHT CALL SALTUS( , ) HERE.  However, the GAIN will be MINUTE.
2300 I# = 1# + INT( ( MAXIMUM# - AUXE# ) / PIEPS# )
2310 IF I# > 1# GOTO 2330
2320 I# = 1#
2330 AUXNEWN# = AUXNEWN# - I#
2340 IF AUXNEWN# <= OLDN# GOTO 2500
2350 CALL SALTUS( AUXNEWN# , UPSHOT )
2360 IF UPSHOT > 0 GOTO 2430
2370 '        Some SREEN INFORMATION !
2380 '        PRINT
2390 '        PRINT "*......"; AUXNEWN#; ".....*     NO EVALUATION NECESSARY "
2400 AUXNEWN# = AUXNEWN# - 1#
2410 AUXE# = AUXE# + PI#
2420 GOTO 2340
2430 CALL GAUSS( AUXNEWN# , NPOINTS# )
2440 AUXE# = NPOINTS# - PI# * AUXNEWN#
2450 IF AUXE# < MAXIMUM# GOTO 2300       'The INFIMA have already been checked
2460 KOUNT = KOUNT + 1
2470 TEMPN#( KOUNT ) = AUXNEWN#
2480 TEMPMAX#( KOUNT ) = AUXE#
2490 GOTO 2300
2500 IF KOUNT < 1 GOTO 2680
2510 MAXIMUM# = TEMPMAX#( KOUNT )
252C IK# = TEMPN#( KOUNT )
2530 QATMAX = MAXIMUM# / (1# + SQR( SQR( IK# ) ) * LOG( IK# ) )
2540 LPRINT " (+) N = "; IK#;
2550 LPRINT "   E = "; CSNG( MAXIMUM# ); "   Q = "; QATMAX
2560 ' LPRINT " SQR("; IK#; ") ="; SQR( IK# )
2570 IF KOUNT < 2 GOTO 2680
2580 ' Now we UNSCRAMBLE the MAXIMA
2590 FOR K = KOUNT - 1 TO 1 STEP -1
2600 IF TEMPMAX#( K ) < MAXIMUM# GOTO 2670
2610 MAXIMUM# = TEMPMAX#( K )
2620 IK# = TEMPN#( K )
2630 QATMAX = MAXIMUM# / ( 1# + SQR( SQR( IK# ) ) * LOG( IK# ) )
2640 LPRINT " (+) N = "; IK#;
2650 LPRINT "   E = "; CSNG( MAXIMUM# ); "   Q = "; QATMAX
2660 ' LPRINT " SQR("; IK#; ") = "; SQR( IK# )
2670 NEXT K
2680 END SUB
```

There are various methods for the computation of $P(t)$. Without going into details (which are quite machine dependent) we make a few brief comments on some possible approaches (also see Keller & Swenson [20]).

* We call the method (see subroutine GAUSS) in program GAUSSXTR the 'root-method'. If (fast) memory permits, it may be advantageous to precompute the necessary squares. One may also experiment with building these squares by means of a linear recurrence: $(i+1)^2 = i^2 + j$, where $j = 2i + 1$.
   The root-method appears to be fast when implemented on computers equipped with a math coprocessor.
* Keller & Swenson [20] used the 'step-method'. In case this method is faster than the root-method one may speed up this method slightly by subdividing the basis of the half-moon-shaped sector of the circle in intervals I_n such that the slope of the circle is $< -n$ in I_n. Inside such an interval one may apply upward steps of size n instead of 1. The endpoints of these intervals are easily computed by elementary calculus. This method seems to be preferable on computers without a math coprocessor (in our case the Sun 4 SPARC Station 1).

Since, for large t, the negative extremal values of $E(t)$ appear to be definitely more pronounced than the positive ones, one might decide to concentrate entirely on the negative ones by deleting the subroutine CHECKMAX (with all fittings) from the main program.

This yields a speedup factor of about 2 at the cost of 'some' completeness of the results.

6. FURTHER OBSERVATIONS AND EXPERIMENTS

At first sight it seemed to us that the extremal values of $E(t)$ occur rather 'randomly'. However, a closer look revealed that there seems to be a fascinating 'quasi regularity', especially with respect to the negative extremal values. The reader may check for himself that if $E(t)$ is extremal and negative then t 'always lies just below' a perfect square. As a typical simple example we mention $t = 97(-0)$. Since we have not found any exception to the 'rule' just stated, we have ventured to promote this observation to a doctrine and wrote a special program that searches for 'low values' of $E(t)$, only for t's just below perfect squares.

We present a *selection* of the results of this program.

TABEL 4. *Some* low values of $E(t)$ for $t = n^2 - \mathcal{O}(\sqrt{n})$

t	$E(t)$	$Q(t)$
64,407,835,130($-$ 0)	$-4,661.0$	-0.372
66,501,577,825($-$ 0)	$-4,766.2$	-0.377
85,429,935,378($-$ 0)	$-4,779.2$	-0.351

88,815,918,244(− 0)	− 4,804.2	− 0.349
91,021,079,444(− 0)	− 5,026.1	− 0.363
101,405,304,001(− 0)	− 5,327.6	− 0.373
104,720,838,337(− 0)	− 5,468.3	− 0.379
152,521,478,773(− 0)	− 5,746.9	− 0.357
179,934,604,660(− 0)	− 6,153.4	− 0.365
200,220,451,202(− 0)	− 6,393.6	− 0.367
242,696,137,050(− 0)	− 7,357.9	− 0.400
485,392,279,681(− 0)	− 7,614.0	− 0.339
523,604,188,801(− 0)	− 7,841.1	− 0.342
591,033,446,305(− 0)	− 8,312.6	− 0.350
638,954,027,714(− 0)	− 8,522.9	− 0.351
886,358,229,149(− 0)	− 8,634.4	− 0.323
962,429,671,165(− 0)	− 8,799.8	− 0.322
985,195,196,576(− 0)	− 9,070.1	− 0.330
1,097,591,472,016(− 0)	− 9,191.3	− 0.324
1,141,061,921,864(− 0)	− 9,490.0	− 0.331
1,325,441,033,284(− 0)	− 9,790.5	− 0.327
1,465,932,511,466(− 0)	− 10,647.0	− 0.345
2,051,723,921,626(− 0)	− 10,833.7	− 0.319
3,256,974,567,522(− 0)	− 10,978.9	− 0.284
3,435,399,231,265(− 0)	− 11,849.1	− 0.302
3,453,241,721,716(− 0)	− 12,563.8	− 0.319
5,076,464,116,145(− 0)	− 12,948.3	− 0.295
6,271,578,593,284(− 0)	− 13,863.0	− 0.297
7,027,504,080,416(− 0)	− 14,614.2	− 0.303
8,930,695,934,245(− 0)	− 15,763.3	− 0.306
10,155,789,844,210(− 0)	− 16,371.0	− 0.306
18,105,195,196,546(− 0)	− 17,741.1	− 0.282
27,739,182,240,000(− 0)	− 18,315.5	− 0.258
29,218,349,160,000(− 0)	− 18,320.5	− 0.254
36,001,176,008,449(− 0)	− 21,391.5	− 0.280
46,500,397,574,400(− 0)	− 21,602.7	− 0.263
74,798,973,849,553(− 0)	− 23,961.7	− 0.255
100,081,536,563,088(− 0)	− 24,822.5	− 0.243
100,118,735,162,177(− 0)	− 24,932.4	− 0.244
100,209,729,800,645(− 0)	− 26,787.5	− 0.263
100,620,800,502,202(− 0)	− 30,749.0	− 0.301
228,719,465,828,473(− 0)	− 31,863.5	− 0.248
339,804,982,440,000(− 0)	− 34,179.7	− 0.238
825,183,363,254,825(− 0)	− 35,378.8	− 0.192
920,095,256,946,308(− 0)	− 38,393.3	− 0.202
969,150,181,400,018(− 0)	− 39,358.9	− 0.204
998,610,560,638,085(− 0)	− 39,751.5	− 0.205
1,015,043,733,773,154(− 0)	− 40,505.4	− 0.208
1,050,871,873,459,634(− 0)	− 40,623.9	− 0.206

1,095,502,295,247,104(− 0)	− 44,045.4	− 0.221
1,374,644,087,373,226(− 0)	− 45,487.5	− 0.214
1,684,294,158,258,064(− 0)	− 51,193.2	− 0.228
1,777,808,461,651,993(− 0)	− 54,819.6	− 0.240
2,025,014,490,021,601(− 0)	− 57,142.1	− 0.242
2,280,284,160,887,041(− 0)	− 59,432.9	− 0.243

We suggest that in future systematic computations one might gather some statistics concerning the distances of the true low extremal t-values to the corresponding nearest squares.

For t's for which $E(t)$ is negatively extremal we define

$$d(t) := \frac{([(t-1/2)^{1/2}]+1)^2 - t}{t^{1/4}}.$$ (6.1)

Although $d(t)$ is usually quite small (< 2), we nevertheless get the impression that $d(t)$ is unbounded and that significant extremes of $E(t)$ may occur indeed for quite large values of $d(t)$ (>10, for example) so that Table 4 will most probably *not* be complete (nor optimal).

With respect to the positive extremal values of $E(t)$ we did not find such a simple 'rule', although it seems likely that they are usually located close to t's of the form $t = (n + 1/4)^2$. Since the negative extremal values seem to be the most significant ones for our purpose, and in order to save computation time we have not pursued this subject any further.

7. MACHINES, PROGRAMS, AND FUTURE COMPUTATIONS

The first versions of all our programs (in Micro Soft Quick Basic) were developed on an Olivetti M24 and an IBM AT (equipped with a math coprocessor). Soon afterwards the programs were translated into FORTRAN 2 for implementation on an IBM PS / 2 70 386 (equipped with an i387 math coprocessor), and into the language C for a Sun 3/50, a Sun 4/80 SPARC and a Sun 4 SPARC Station 1. Various checks were carried out, by means of a FORTRAN 5 program, on a CDC CYBER 990 system.

REMARK. During the preparation of this lecture, J.T. Tromp (at CWI) notified us that he has already developed a considerably faster procedure for (systematically) finding the extremal values of $E(t)$. See TROMP [42].

SUGGESTION (by J.T. Tromp). Any reader who wishes to perform a check on our tables (or to find better extremes of $E(t)$) is recommended to compute $P(n)$ by means of a subroutine such as

```
        SUB GAUSS( N , P )   STATIC
REM  TROMP's IMPROVED STEP - METHOD
        ROOT = SQR( N + 0.5 )
        K = INT( ROOT / SQR( 2 ) )
        L = INT( ROOT )
        I = 2 * L - 1
        IK = 2 * K + 1
        SUM = 0
        IF I < IK GOTO 246
        J = - 1
        IJ = L * L - N
245    J = J + 2
        IJ = IJ + J
        IF IJ < = 0 GOTO 245
        SUM = SUM + J
        IJ = IJ - I
        I = I - 2
        IF I > = IK GOTO 245
246    P = IK * IK + 4 * SUM
        END SUB
```

The adjustment of the necessary type-declarations of the various variables is left to the reader.

REFERENCES

[1] CHEN JING RUN, *The lattice points in a circle*, Sci. Sinica 12 (1963) pp. 633-649.

[2] J.G. VAN DER CORPUT, *Neue zahlentheoretische Abschätzungen*, Math. Ann. 89 (1923) pp. 215-254.

[3] J.G. VAN DER CORPUT & E. LANDAU, *Über Gitterpunkte in ebenen Bereichen*, Nachr. Ges. Wiss. Göttingen, Math. Phys. Klasse (1920) pp. 135-171.

[4] P. ERDÖS & W.H.J. FUCHS, *On a problem of additive number theory*, J. London Math. Soc. 31 (1956) pp. 67-73.

[5] W. FRASER & C.C. GOTLIEB, *A calculation of the number of lattice points in the circle and sphere*, Math. Comp. (1962) pp. 282-290.

[6] F. FRICKER, *Einführung in die Gitterpunktlehre*, Birkhäuser, 1982.

[7] C.F. GAUSS, *De nexu inter multitudinem classium, in quas formae binariae*

54

secundi gradus distribuuntur, earumque determinantem, Werke (1863), Vol. 2, pp. 269-291 (in particular p. 280).

[8] C.F. GAUSS, *Disquisitiones arithmeticae*, (German edition by H. Maser), 1886, p. 657.

[9] G.H. HARDY, *On the expression of a number as the sum of two squares*, Quart. J. Math., Oxford Ser. 46 (1915) pp. 263-283. Collected Papers, Vol. 2 (1967) pp. 243-283.

[10] G.H. HARDY, *On Dirichlet's divisor problem*, Proc. London Math. Soc. 15 (1916) pp. 1-25.

[11] D.A. HEJHAL, *The Selberg trace formula and the Riemann zeta function*, Duke Math. J. 43 (1976) pp. 441-482.

[12] HUA LOO KENG, *The lattice points in a circle*, Quart. J. Math., Oxford Ser. 13 (1942) pp. 18-29.

[13] HUA LOO KENG, *Die Abschätzung von Exponentialsummen und ihre Anwendung in der Zahlentheorie*, Enzycl. Math. Wiss. Anw., Vol. 1, Part 2, *Analytische Zahlentheorie*, Leipzig, 1959.

[14] HUA LOO KENG, *Introduction to Number Theory*, Springer, 1982.

[15] A.E. INGHAM, *On two classical lattice point problems*, Proc. Cambr. Phil. Soc. 36 (1940) pp. 131-138.

[16] A. IVIC, *The Riemann zeta - function*, Wiley & Sons, New York, 1985, p. 383.

[17] H. IWANIEC & C.J. MOZZOCHI, *On the divisor and circle problems*, J. of Number Th. 29 (1988) pp. 60-93.

[18] V. JARNIK, *Über die Gitterpunkte auf konvexen Kurven*, Math. Z. 24 (1925) pp. 500-518.

[19] I. KATAI, *The number of lattice points in a circle*, (in Russian), Ann. Univ. Sci. Budapest Rolando Eötvös, Sect. Math. 8 (1965) pp. 39-60.

[20] H.B. KELLER & J.R. SWENSON, *Experiments on the lattice problem of Gauss*, Math. Comp. 17 (1963) pp. 223-230.

[21] G. KOLESNIK, *On the method of exponent pairs*, Acta Arith. 45 (1985) pp. 115-143.

[22] E. KRÄTZEL, *Lattice points*, Kluwer Acad. Publ., Dordrecht, 1988.

[23] E. LANDAU, *Die Bedeutung der Pfeiffer'schen Methode für die analytische Zahlentheorie*, Sitzber. K. Akad. Wiss. Wien, Math. Natw. Klasse 121 (1912) pp. 2298-2328.

[24] E. LANDAU, *Über die Zerlegung der Zahlen in zwei Quadrate*, Annali di Matematica, Ser. 3, Vol. 20 (1913) pp. 1-28.

[25] E. LANDAU, *Über die Gitterpunkte in einem Kreise (II)*, Göttinger Nachr. (1915) pp. 161-171.

[26] E. LANDAU, *Note on the preceding paper*, see [28; pp. 110-111] or [29; pp. 487-488].

[27] E. LANDAU, *Vorlesungen über Zahlentheorie*, Vol. 2, Part 8, 1927.

[28] E. LANDAU, (edited by A. Walfisz), *Ausgewählte Abhandlungen zur Gitterpunktlehre*, Deutscher Verl. Wiss., 1962.

[29] J.E. LITTLEWOOD & A. WALFISZ, *The lattice points of a circle*, Proc. Roy. Soc. London, Ser. A 106 (1924) pp. 478-487. Followed by a note by Landau, pp. 487-488.

[30] H.L. MITCHELL, *Numerical experiments on the number of lattice points in the circle*, Techn. Report No. 17, Appl. Math. Stat. Labs., Stanford Univ., Stanford, California.

[31] L.W. NIELAND, *Zum Kreisproblem*, Math. Ann. 98 (1928) pp. 717-736.

[32] W.-G. NOWAK, *Lattice points in a circle and divisors in arithmetic progressions*, Manuscr. Math. 49 (1984) pp. 195-205.

[33] E. PFEIFFER, *Über die Periodicität in der Teilbarkeit der Zahlen und über die Verteilung der Klassen positiver quadratischer Formen auf ihre Determinanten*, Jahresber. der Pfeiffer'schen Lehr- und Erziehungsanstalt zu Jena (1886) pp. 1-21.

[34] W. SIERPINSKI, *O pewnem zagadnieniu z rachunku funkcyj asymptotycznych*, Prace mat. fyz. 17 (1906) pp. 77-118.
Summary in French: *Sur un probleme du calcul des fonctions asymptotiques*, pp. 115-118.

[35] E.C. TITCHMARSH, *The lattice points in a circle*, Proc. London Math. Soc. (2) 38 (1934) pp. 96-115. Corrigendum p. 555.

56

[36] I.M. VINOGRADOV, *Elements of Number Theory*, Dover Publ., 1954.

[37] G. VORONOÏ, *Sur un problème du calcul des fonctions asymptotiques*, J. r. a. M. 126 (1903) pp. 241-282.

[38] A. WALFISZ, *Über zwei Gitterpunktprobleme*, Math. Ann. 95 (1927) pp. 69-83.

[39] A. WALFISZ, *Teilerprobleme*, Math. Z. 26 (1927) pp. 66-88.

[40] J.R. WILTON, *The lattice points of a circle: an historical account of the problem*, Mess. Math. 48 (1928) pp. 67-80.

[41] YIN WEN LIN, *The lattice points in a circle*, Sci. Sinica 11 (1962) pp. 10-15.

[42] J.T. TROMP, *More computations on Gauss' lattice point problem*, Mathematical Centre Report CS-R9017 (1990) 10 p.