Computing the Zeros by Odlyzko-Schönhage Algorithm

*Essay 4 on the course "Numerical Analysis".

1st Chen Yihang
Peking University
1700010780

***Abstract*—In this report, We introduce the history of computing the zeros of the Riemann ζ function, and then we give a brief presentation of Odlyzko-Schönhage algorithm, by which the first 10^{13} zeros are obtained, without requiring much background knowledge.**

The general idea of Odlyzko-Schönhage algorithm is to obtain values of objective function on a regular grid, which untilizes FFT and two algorithms, Chebychev interpolation as well as Greengard-Rokhlin algorithm, to accelerate the algorithm. In sequel, we use such values to perform interpolation. Finer estimates of zeros require convoluted hypothesis on the ζ function, which will not be discussed in this report. In other words, we will focus on the role of numerical analysis in this algorithm.

I. History

The Riemann Zeta function is defined by

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}$$

for complex values of s, which can be be analytically continued to the whole complex plane (with a single pole at $s = 1$). It is known that all non-trivial zeros are located in the band $0 < \mathcal{R}(s) < 1$. The Riemann Hypothesis is the conjecture that all these zeros are located on the critical line $\mathcal{R}(s) = 1/2$.

While numerical computations on zeros of the Zeta function have long been focused on RH verification only (to check the RH, isolating the zeros is sufficient so no precise computations of the zeros are needed) it was Odlyzko who the first, computed precisely large consecutive sets of zeros to observe their distribution.

Odlyzko made some empirical observations of the distribution on the spacing between zeros of $\zeta(s)$ in various zones and checked the correspondence with the GUE hypothesis. In 1987, Odlyzko computed numerically 10^5 zeros of the Riemann Zeta function between index $10^{12}+1$ and $10^{12}+10^5$ to the accuracy of 10^{-8} and was the first to observe a good agreement with the GUE hypothesis . Later, in order to reach much higher heights, Odlyzko with Schönhage developed a fast algorithm for multi-evaluation of $\zeta(s)$. After refinements to make this method efficient for practical purposes, Odlyzko was able to compute 70 million zeros at height 10^{20} in 1989 and then 175 million in 1992 at the same height. Later he reached the height 10^{21}, and in 2001 he computed ten billion zeros at height 10^{22}. In a more recent unpublished work in 2002, Odlyzko computed twenty billion zeros at height 10^{23}. More detailed information can be found at the website Computation of zeros of the Zeta function

Wedeniwski from Böblingen IBM laboratory initiate a system called ZetaGrid, which was at one time the largest distributed computing project designed to explore roots of the Riemann zeta function, checking over one billion roots a day. The project ended in November 2005 due to instability of the hosting provider. Over 10^{13} first zeroes were checked. However, they are not the first group to accomplish this goal. Xavier Gourdon and Patrick Demichel published their calculations of the first 10^{13} zeros of the zeta function, who adopted more efficient algorithm, i.e. "Odlyzko-Schönhage algorithm".[Xavier, 2004] They only use took the equivalent of 525 days of a single modern computer in 2003 (Pentium 4 processor 2.4 Ghz). Not only did they find no exception to Riemann Hypothesis on the first 10^{13} zeros, their results substantiate some other hypothesis (including GUE hypothesis) on the distribution of zeros Riemann zeta function.

A python package mpmath.zetazero is able to calculate n-th zero of ζ function[1], which is not as efficient as the Odlyzko-Schönhage algorithm when trying to compute a large number of consecutive zeros.

II. Odlyzko-Schönhage algorithm

A. Notations and definitions

A consequence of the Zeta functional equation is that the function

$$Z(t) = e^{i\theta(t)}\zeta\left(\frac{1}{2}+it\right)$$
$$\text{where} \quad \theta(t) = \arg(\pi^{-it/2}\Gamma(1/4+it/2))$$

known as the Riemann-Siegel Z-function, is real valued, and satisfies the Riemann-Siegel expansion

$$Z(t) = 2\sum_{n=1}^{m} \frac{\cos(\theta(t) - t\log n)}{\sqrt{n}} + R(t)$$
$$R(t) = (-1)^{m+1}\tau^{-1/2}\sum_{j=0}^{M}(-1)^j\tau^{-j}\Phi_j(z) + R_M(t)$$

where $R_M(t) = \mathcal{O}(t^{-(2M+3)/4})$, and $\tau = \sqrt{\frac{t}{2\pi}}$, $m = \lfloor\tau\rfloor$, $z = 2(t-m)-1$.

B. Presentation of the algorithm

The Odlyzko-Schönhage algorithm permits efficient evaluations of $Z(t)$ in a range of form $T \le t \le T+\Delta$, with $\Delta = \mathcal{O}(\sqrt{T})$. We write

$$Z(t) = \sum_{n=1}^{k_0-1} \frac{\cos(\theta(t) - t\log n)}{\sqrt{n}} + \mathcal{R}(e^{-i\theta(t)}F(k_0-1, k_1; t))$$
$$+ \sum_{k_1+1}^{m} \frac{\cos(\theta(t) - t\log n)}{\sqrt{n}} + R(t)$$

where $k_1 = \lfloor\sqrt{T/2\pi}\rfloor$ and k_0 is a small fixed integer, and $m-k_1$ is bounded. The selection of k_0, k_1 will not be discussed. The

[1]https://kite.com/python/docs/mpmath.zetazero

most time consuming part is in the computation of $Z(t)$ is the computation of $F(t)$. We have

$$F(t) = \sum_{k=k_0}^{k_1} \frac{1}{\sqrt{k}} \exp(it \log k) \tag{II.3}$$

The rest of the technique is mainly dedicated to fast evaluation of this sum. We will not consider other parts of $Z(t)$ in the following of the report.

C. Fast multi-evaluation of $F(t)$ on a regular grid

We are to evaluate $F(t)$ at evenly spaced values

$$t = T_0, T_0 + \delta, \cdots, T_0 + (R-1)\delta.$$

where the selection of δ will not be discussed. The key idea is to compute their Fourier transform by

$$u_k = \sum_{j=0}^{R-1} F(T_0 + j\delta)\omega^{-jk}, \quad \omega = \exp(2\pi i/R).$$

and the inverse Fourier transform gives

$$F(T_0 + j\delta) = \frac{1}{R} \sum_{k=0}^{R-1} u_k \omega^{jk}$$

We set R to be a power of two and use FFT transform. Since this FFT takes a small portion of the total time, the problem is now reduced to computing efficiently the u_k. By equation II.3, we obtain $u_k = \omega^k f(\omega^k)$, where $f(z)$ is defined by

$$f(z) = \sum_{k=k_0}^{k_1} \frac{a_k}{z - b_k}, b_k = \exp(i\delta \log k),$$
$$a_k = \frac{\exp(iT_0 \log k)}{\sqrt{k}} (1 - \exp(iR\delta \log k))$$

D. Fast multi-evaluations of $f(\omega^k)$ by Chebychev interpolation

1) Construction of Chebychev polynomials: We use Chebychev interpolation of the function $G(\theta) = \sum_k a_k/(\exp(i\theta) - b_k)$ of the real variable θ, on the interval $[\theta_0 - L, \theta_0 + L]$, the point-and-derivative interpolation of degree $2N$ is performed on

$$\alpha_j = \theta_0 + L \cos \frac{(2j+1)\pi}{2N}, \quad 0 \le j \le N-1$$

If we assume $R_N(\theta) = \prod_{j=0}^{N-1}(\theta - \alpha_j)$, the interpolation polynomial is defined by

$$Q_{N,\theta_0,L}(\theta) = \sum_{j=0}^{N-1} \left(\frac{G(\alpha_j) R_N(\theta)^2}{R'_N(\alpha_j)^2(\theta - \alpha_j)^2} - \frac{G(\alpha_j) R''_N(\alpha_j) R_N(\theta)^2}{R'_N(\alpha_j)^3(\theta - \alpha_j)} + \frac{G'(\alpha_j) R_N(\theta)^2}{R'_N(\alpha_j)^2(\theta - \alpha_j)} \right)$$

Here, we use point-and-derivative interpolation since it is cheaper in terms of performance to compute G and G' at a given point rather than computing G at two different points.

If

$$\|\beta_k - \theta_0\| := \min_n |\beta_k - \theta_0 + 2n\pi| > \lambda L$$

then for all real values of θ such that $|\theta - \theta_0| < L$, we have the bound

$$|G(\theta) - Q_{N,\theta_0,L}(\theta) < \frac{8 + 2^{2-N}}{(\lambda - 1) L K(\lambda)^{2N}} \sum_k |a_k| \tag{II.5}$$

where $K(\lambda) = \lambda + \sqrt{\lambda^2 - 1}$.

It is convenient to express the interpolation on $[-1, 1]$ instead of $[\theta_0 - L, \theta_0 + L]$, which write in the form $Q_N(t) := G(\theta_0 + Lt)$ with

$$Q_N(t) = \sum_{j=0}^{N-1} \left(\frac{G(\theta_0 + L\gamma_j) T_N(t)^2}{T'_N(\gamma_j)^2 (t - \gamma_j)^2} - \frac{G(\theta_0 + L\gamma_j) T''_N(\gamma_j) T_N(t)^2}{T'_N(\gamma_j)^3 (t - \gamma_j)} + \frac{LG'(\theta_0 + L\gamma_j) T_N(t)^2}{T'_N(\gamma_j)^2 (t - \gamma_j)} \right)$$

where $\gamma_j = \cos \frac{2j+1}{N}\pi$ and $T_N(t) = 2^{N-1} \prod_j (t - \gamma_j)$. We can write $Q_N(t)$ as

$$Q_N(t) = \sum_{j=1}^{N-1} G(\theta_0 + L\gamma_j) V_j(t) + LG'(\theta_0 + L\gamma_j) W_j(t)$$

with

$$A_j(t) = \frac{1}{\prod_{k \neq j} (\gamma_j - \gamma_k)^2} \left(1 - 2(t - \gamma_j) \sum_{k \neq j} \frac{1}{\gamma_j - \gamma_k} \right)$$
$$B_j(t) = \frac{t - \gamma_j}{\prod_{k \neq j} (\gamma_j - \gamma_k)^2}$$
$$V_j(t) = \left(\prod_{k \neq j} (t - \gamma_k)^2 \right) A_j(t), W_j(t) = \left(\prod_{k \neq j} (t - \gamma_k)^2 \right) B_j(t)$$

Since $\gamma_j = -\gamma_{n-j}$, we can first compute

$$L_j(t) = G(\theta_0 + L\gamma_j) A_j(t) + LG'(\theta_0 + L\gamma_j) B_j(t)$$

and then we compute $C_j(t), j < N/2$ by

$$C_j(t) = (t - \gamma_j)^2 L_{n-j}(t) + (t - \gamma_{n-j})^2 L_j(t)$$

and finally, we have

$$Q_N(t) = \sum_{0 \le j < N/2} \left(\prod_{k \neq j 0 \le j < N/2} (t^2 - \gamma_k^2)^2 \right) C_j(t)$$

Such algorithm is more stable in numerical computation.

2) Recursive process for fast computation of $f(\omega^j)$.: Suppose we want to compute approximations of the values $f(\omega^j)$ for consecutive indexes $j = j_0, j_0 + 1, \cdots, j_1$. We define $j_c = (j_0 + j_1)/2$ and $L = (j_1 - j_0)\pi/R$. Then $[2j_0\pi/R, 2j_1\pi/R] = [\theta_0 - L, \theta_0 + L]$, where $\theta_0 = 2j_c\pi/R$.

We want to construct I of indices in $K = \{k_0, k_0 + 1, \cdots, k_1\}$, such that $\|\beta_k - \omega^{j_c}\| > \lambda L, \forall k \in I, \lambda > 1$, which is necessary from II.5 to bound the interpolation error. The contribution $f_I(\omega^j)$, where

$$f_I(z) = \sum_{k \in I} \frac{a_k}{z - b_k}$$

can be approximated efficiently for values $j = j_0, \cdots, j_1$ by using point-and-derivative Chebychev interpolation of $f_I(e^{i\theta})$ on the interval $[\theta_0 - L, \theta_0 + L]$.

For the remaining k indices in $K - I$, the aforementioned process is recursively applied. We split indices j into $j_0, \cdots, j_c$ and $j_c + 1, \cdots, j_1$ and obtain I' from $K - I$ in the same way.

When the number of j indices is small, we directly evaluate $f_I(\omega^j)$ instead of Chebychev interpolation. When ω^j is close to one of the β_k, we choose to use the following stable expansion instead

$$\frac{a_k}{\omega^j - b_k} = -\frac{1}{\sqrt{k}} \exp(-i((R+1)x + \delta \log k - T_0 \log k)) \frac{\sin Rx}{\sin x} \tag{II.7}$$

where $x = \frac{j\pi}{R} - \frac{\delta \log k}{2}$.

In practice, $\lambda = 2$ and $N = 8$ leads to good results, which is efficient since $N \ll j_1 - j_0$.

E. Implementation of the Greengard-Rokhlin algorithm for huge height

When the number of k indices is much bigger than the number of j indices in the evaluations of $f(\omega^j)$, the Greengard-Rokhlin algorithm is much more efficient than the Chebychev interpolation based technique. Since

$$\frac{a_k}{z - b_k} = \sum_{n=0}^{\infty} a_k (b_k - c)^n \frac{1}{(z-c)^{n+1}}$$

we have

$$f_K(z) := \sum_{k \in K} \frac{a_k}{z - b_k} \approx \sum_{n=0}^{V-1} \frac{A_n(K, c)}{(z-c)^{n+1}} \tag{II.8}$$

where

$$A_n(K, c) = \sum_{k \in K} a_k (b_k - c)^n \tag{II.9}$$

Once the value of $A_j(K, c)$ is known, the value of $A_n(K, d)$ can be obtained by

$$A_n(K, d) = \sum_{j=0}^{n} \binom{n}{j} (c - d)^{n-j} A_j(K, c) \tag{II.10}$$

with $\mathcal{O}(n)$ operations, which is much more efficient than directly compute A_n by equation II.9 since $n < V \ll |\#K|$.

The recursive process of Greengard-Rokhlin algorithm For $K \subset \{k_0, k_0 + 1, \cdots, k_1\}, J \subset \{0, 1, \cdots, R-1\}$, the Greengard-Rokhlin procedure, defined by GR(J, K) computes approximations of the contributions $f_K(\omega^j)$, and also returns a pole c and the sequence of coefficients $A_n(K, c)$ for $0 \le n < V$.

1) $c = \exp(\frac{\beta_{\min} + \beta_{\max}}{2} i)$
2) If $\#K = \mathcal{O}(V)$ or $\#J = \mathcal{O}(1)$, apply direct computation, where stable expansion II.7 might be needed. The value $A_n(J, c)$ is also directly computed by equation II.9.
3) Otherwise, define $\beta' = \frac{\beta_{\min} + \beta_{\max}}{2}$ and $L = \frac{\beta_{\max} - \beta_{\min}}{2}$.
   a) Compute $J' \subset J$ such that $|\omega^j - e^{i\beta}| \le \lambda |1 - e^{iL}|, \forall j \in J'$, where $\lambda > 1$.
   b) Divide K into K_1 and K_2 such that $\beta_k < \beta', \forall k \in K_1$.
   c) Call GR(J', K_1), GR(J', K_2), and then obtain $A_n(K, c) = A_n(K_1, c) + A_n(K_2, c)$, where $A_n(K_i, c)$ is obtained from $A_n(K_i, c_i)$ by equation II.10 for $i = 1, 2$. The values of $f_K(\omega^j)$ is obtained as $f_{K_1}(\omega^j) + f_{K_2}(\omega^j)$ for $j \in J'$, or with Taylor approximation fomula II.8 for $j \in J - J'$.

For $j \in J - J'$, the error can be bound by

$$\sum_{n \ge V} \frac{|A_n(K, c)|}{|\omega^j - c|^{n+1}} \le \frac{1}{\lambda |1 - e^{iL}|} \left(\sum_{k \in K} |a_k| \right) \sum_{n \ge V} \frac{1}{\lambda^n}.$$

III. Band limited function interpolation

A. Interpolation of the function $F(t)$

It is known that for a band-limited function of the form

$$G(x) = \int_{-\tau}^{\tau} g(t) e^{ixt} dt$$

then under smooth conditions

$$G(x) = \sum_{n=-\infty}^{+\infty} G(\frac{n\pi}{\tau}) \frac{\sin(\tau x - n\pi)}{\tau x - n\pi}. \tag{III.1}$$

We have the following proposition: Let $G(z)$ be a complex function of the complex variable such that $|G(z)| = \mathcal{O}(e^{\tau \Im(z)}), \tau > 0$. Let $\beta > \tau$, and let a kernel complex function $h(z)$ satisfy $h(0) = 1. |h(z)| = o(e^{\gamma \Im(z)}), \tau + \gamma \le \beta$. Then we have

$$G(x) = \sum_{n=-\infty}^{+\infty} G\left(\frac{n\pi}{\beta}\right) h\left(x - \frac{n\pi}{\beta}\right) \frac{\sin(\beta x - n\pi)}{\beta x - n\pi}. \tag{III.2}$$

Hence, instead of dealing with $F(t)$, we interpolate

$$G(t) = \exp(-i\alpha t) F(t), \ \alpha = \frac{\log k_0 + \log k_1}{2}$$

Then

$$|G(z)| = \mathcal{O}(e^{\tau \Im(z)}), \ \tau = \frac{\log k_1 - \log k_0}{2}$$

Setting $\beta = \lambda\tau, \ \tau > 1, \gamma = \beta - \tau$, and the kernel function

$$h_M(t) = \left(\frac{\sin(\gamma t / M)}{\gamma t / M} \right)^M$$

By formula III.2, and truncate it into a finite sum, we have

$$\widehat{G}(x) = \sum_{n : |x - \frac{n\pi}{\beta}| < \frac{2.55M}{\gamma}} \widehat{G}\left(\frac{n\pi}{\beta}\right) h_M\left(x - \frac{n\pi}{\beta}\right) \frac{\sin(\beta x - n\pi)}{\beta x - n\pi} \tag{III.3}$$

where $\widehat{G}\left(\frac{n\pi}{\beta}\right)$ can be obtained by the Odlyzko-Schönhage algorithm of approximation of $F(t)$ on a regular grid.

B. More accurate approximation of $Z(t)$

In some rare cases (especially between two very close zeros), it may be useful to compute $Z(t)$ more accurately to separate zeros of it. We preferred to use a different technique : when the required precision on Z(t) was higher than the available precision issued from the band-limited function interpolation, we computed $Z(t)$ directly with the classic direct use of the Riemann-Siegel formula.

An Euler-product form of $F(t)$ is obtained by considering the product $P = 2 \times 3 \times \cdots p_h$ of the first h primes (with h small in the practice, say $h \le 4$) and by considering the set Q of integers all of whose prime factors are not larger than p_h. We have

$$F(t) = \sum_{k=1}^{m} k^{-1/2+it} = \sum_{0 < k \le m, \ (k,P)=1} k^{-1/2+it} s(m/k)$$

$$s(a) = \sum_{0 < l \le a, \ l \in Q} l^{-1/2+it}$$

Since the $s(m/k)$ for $0 < k \le m$ are the same for many k the summation above is much more efficient than the direct summation for $F(t)$, and the asymptotically expected factor is at most $\prod_{i=1}^{h} (1 - 1/p_h)$.

References

[Xavier, 2004] Xavier (2004). The 10 13 first zeros of the riemann zeta function , and zeros computation at very large height.