A fast recursive algorithm for computing cyclotomic polynomials

Andrew Arnold (ada26@sfu.ca), Centre for Experimental and Constructive Mathematics **CECM**, Simon Fraser University, 2010

SFU

A motivating problem about $\Phi_n(z)$

Definition 1. *The n_{th}* ***cyclotomic polynomial****, $\Phi_n(z)$, is the monic polynomial whose $\phi(n)$ roots are the n_{th} primitive roots of unity.*

$$\Phi_n(z) = \prod_{\substack{k=0 \\ (k,n)=1}}^{n} (z - e^{\frac{2\pi i}{n}k}) = \sum_{k=0}^{\phi(n)} a_n(k) z^k.$$

Definition 2. *The n_{th}* ***inverse cyclotomic polynomial****, $\Psi_n(z)$, is the monic polynomial whose roots are the n_{th} nonprimitive roots of unity. It is the polynomial satisying $\Phi_n(z)\Psi_n(z) = z^n - 1$ (see [3]).*

Definition 3. *We let $A(n)$ denote the height of $\Phi_n(z)$: the magnitude of its largest coefficient.*

$$A(n) = \max_{0 \le j \le \phi(n)} |a_n(j)|.$$

The first six cyclotomic polynomials are

$$\Phi_1(z) = z - 1, \quad \Phi_2(z) = z + 1, \quad \Phi_3(z) = z^2 + z + 1,$$
$$\Phi_4(z) = z^2 + 1, \quad \Phi_5(z) = z^4 + z^3 + z^2 + z + 1, \quad \Phi_6(z) = z^2 - z + 1.$$

For $n \le 6$, we see that $A(n) = 1$. The following theorems tell us, however, that the cyclotomic polynomial coefficients can become arbitrarily large.

Theorem 4 (Erdos [1]). *Let $c > 0$. Then there exists n such that $A(n) > n^c$.*

Theorem 5 (Maier [2]). *The set of $n \in \mathbb{N}$ satisfying $A(n) > n^c$, for any fixed c, has positive lower density.*

The coefficients of cyclotomic polynomials that are easy to compute, however, are typically very small. For $n < 10^6$, $A(n) < 6 \cdot 10^4$. We were originally motivated by the question: Given $c > 0$, **what is the least n for which $A(n) > n^c$?** To that end we implemented fast algorithms to compute $\Phi_n(z)$. Here are our results:

Table 1: The least n for which $A(n) > n^c$, for $c = 1, 2, 3, 4$

c	n	$A(n)$
1	1181895	14102773
2	43730115	862550638890874931
3	416690995	80103182105128365570406901971
4	1880394945	64540997036010911566826446181523888971563

Lemma 6. *Let $p \nmid n$ be prime. Then $\Phi_{np}(z) = \Phi_n(z^p)/\Phi_n(z)$.*

Lemma 7. *Let $q \mid n$ be prime. Then $\Phi_{nq}(z) = \Phi_n(z^q)$.*

Lemma 8. *Let n be odd. Then $\Phi_{2n}(z) = \Phi_n(-z)$.*

By lemmas 7 and 8, we know if $\bar{n}$ is the greatest squarefree odd divisor of n, then $A(\bar{n}) = A(n)$. As such we only consider $\Phi_n(z)$ of odd, squarefree index. Given $n = p_1 p_2 \cdots p_k$, a product of k odd primes, lemma 6 outlines a way of computing $\Phi_n(z)$ by a sequence of k polynomial divisions. We implemented such a method using the Fast Fourier Transform to perform fast polynomial division (FFT, see timings); however, it was surpassed by the Sparse Power Series (SPS) algorithm.

The palindromic property of $\Phi_n(z)$

To obtain $A(n)$, we need only compute the terms of $\Phi_n(z)$ up to degree $\phi(n)/2$ and not $\phi(n)$. The coefficients of $\Phi_n(z)$, for $n > 1$ are **palindromic**. That is, for $\Phi_n(z) = \sum_{k=0}^{\phi(n)} a_n(k) z^k$, we have that $a(\phi(n) - k) = a(k)$. Thus it is easy to generate the higher-degree terms of $\Phi_n(z)$. Lemma 9 is a more general result which bodes useful in later algorithms.

Lemma 9. *Let*

$$f(z) = \prod_{j=1}^{s} \Phi_{n_j}(z) = \sum_{i=0}^{D} a(i) z^i$$

be a degree D product of cyclotomic polynomials such that n_j is odd for $1 \le j \le s$. Then $a(i) = (-1)^D a(D - i)$. In other words, the coefficients of $f(z)$ are palindromic if D is even, and ***antipalindromic*** *otherwise.*

The sparse power series (SPS) method

The **sparse power series (SPS or SPS1)** method computes $\Phi_n(z)$ as the product

$$\Phi_n(z) = \prod_{d|n} (1 - z^d)^{\mu(n/d)} \tag{1}$$

We call the $(1 - z^d)^{\pm 1}$ comprising $\Phi_n(z)$ in (1) the **subterms** of $\Phi_n(z)$. To compute the product above efficiently, we compute $\Phi_n(z)$ as a truncated power series. The sparse power series earns its name as the power series expansion of nearly all of the subterms appearing in (1) are sparse.

As the power series expansion of $(1 - z^d)^{-1}$ is $(1 + z^d + z^{2d} + \cdots)$, we can multiply a truncated power series of degree D by $(1 - z^d)^{-1}$ in $\mathcal{O}(D)$ arithmetic operations in $\mathbb{Z}$. Multiplying by $(1 - z^d)$ is similarly easy. In addition, the operations on the coefficients are strictly additions and subtractions. Using a dense representation for our truncated power series, these multiplications can be naturally done in memory; we write our resulting product over our previous truncated power series.

Input: n a squarefree, odd integer
Output: $a(0), \ldots, a(\frac{\phi(n)}{2})$, the first half of the coefficients of $\Phi_n(z)$
`// we compute terms up to degree` D
$D \longleftarrow \frac{\phi(n)}{2}$, $(a(0), a(1), \ldots, a(D) \longleftarrow (1, 0, 0, \ldots, 0)$
for $d|n$ *such that* $d < n$ **do**
  if $\mu(\frac{n}{d}) = 1$ **then** `// multiply by` $1 - z^d$
    for $i = D$ **down to** d **by** -1 **do** $a(i) \longleftarrow a(i) - a(i - d)$
  else `// divide by` $1 - z^d$
    for $i = d$ **to** D **do** $a(i) \longleftarrow a(i) + a(i - d)$
return $a(0), a(1), \ldots a(D)$

Procedure SPS(n), computing $\Phi_n(z)$ as a product of sparse power series

The operation cost of the SPS method is $\mathcal{O}(2^k \phi(n))$.

Making SPS faster

Let $n = p_1 p_2 \cdots p_k$ be a product of k distinct odd primes. Let $d_1, d_2, \cdots d_{2^k}$ be the divisors of n in the order by which the SPS algorithm iterates through them all. The SPS algorithm computes the truncated power series of

$$f_s(z) = \prod_{i=1}^{s} (1 - z^{d_i})^{\mu(n/d_i)}$$

for $0 \le s \le 2^k$, all truncated to degree $\phi(n)/2$. If, however, for some s, $f_s(z)$ is a polynomial of degree D_s, then we need only truncate f_t, where $t \le s$, to degree at most D_s. Moreover, if f_s is a polynomial, then $f_s(z)$ is a product of cyclotomic polynomials satisfying lemma 9, hence we need only truncate to degree $\lfloor D/2 \rfloor$.

More generally, if $f_{s_1}, f_{s_2}, \ldots, f_{s_j}$ are polynomials of degree $D_{s_1}, D_{s_2}, \ldots, D_{s_j}$, then for $t \le \min_{1 \le i \le j} s_j$, we need only truncate $f_t(z)$ to degree $\lfloor D/2 \rfloor$, where $D = \min_{1 \le i \le j} D_j$. We call the degree to which we truncate f_t the **degree bound** of f_t.

Aim: Order the divisors $d|n$ in a manner which reduces the degree bound over the computation of $\Phi_n(z)$.

The improved SPS algorithm (SPS2)

Let p be the largest prime divisor of $n = mp$. Then $\Phi_n(z) = \Psi_m(z)\Phi_m(z^p)(z^m - 1)^{-1}$. We can reexpress $\Psi_m(z)$ and $\Phi_m(z)$ as products of subterms of $\Phi_n(z)$:

$$\Phi_n(z) = \left(\prod_{d|m, d<m} (1 - z^d)^{-\mu(\frac{m}{d})} \right) \left(\prod_{d|m} (1 - z^{dp})^{\mu(\frac{m}{d})} \right) (z^m - 1)^{-1}. \tag{2}$$

If compute the product of subterms appearing in $\Psi_m(z)$ first, we can reduce the degree bound to $\lfloor \frac{m - \phi(m)}{2} \rfloor$ when multiplying by these subterms. For n a product of k distinct odd primes, $\Psi_m(z)$ comprises $2^{k-1} - 1$ of the 2^k subterms of $\Psi_m(z)$. We then apply lemma 9 to compute the higher-degree terms of $\Psi_m(z)$. We then truncate to degree $\phi(n)/2$ for the remaining subterms. This method, which we call the **improved SPS method** or SPS2, saves us roughly a factor of 2 time over SPS in practice.

The iterative SPS (SPS3)

Towards further lowering the degree bound, we introduce a cumbersome identity. Let $n = p_1 p_2 \cdots p_k$, a product of k distinct odd primes. For $1 \le i \le k$, let $m_i = p_1 p_2 \cdots p_{i-1}$ and $e_i = p_{i+1} \cdots p_k$. We set $m_1 = e_k = 1$, and let $e_0 = n$. Note that $e_i p_i m_i = n$ for $1 \le i \le k$. By repeated appliation of (2), we can show that

$$\Phi_n(z) = \left(\prod_{j=2}^{k} \Psi_{m_j}(z^{e_j}) \right) \left(\prod_{j=1}^{k} (z^{n/p_j} - 1)^{-1} \right) (z^n - 1) \tag{3}$$

For example, for $n = 105 = 3 \cdot 5 \cdot 7$,

$$\Phi_{105}(z) = \Psi_{15}(z)\Psi_3(z^7)(z^{15} - 1)^{-1}(z^{21} - 1)^{-1}(z^{35} - 1)^{-1}(z^{105} - 1)$$

In the iterative SPS method (SPS3), we first compute the product $\Psi_{m_k}(z^{e_k}) \cdots \Psi_{m_2}(z^{e_2})$ from left to right, raising the degree bound everytime we move onto the next $\Psi_{m_j}(z^{e_j})$ and leveraging lemma 9 to generate higher degree terms of our intermediate polynomial as necessary. For the remaining $k + 1$ subterms we still truncate to degree $\phi(n)/2$ as before.

The gains SPS3 has over SPS2 are more substantial when computing $\Phi_n(z)$ for n with many distinct prime factors. Timings suggest, for n with 6 or more factors, computing $\Phi_n(z)$ using SPS3 is between 2 and 5 times faster than SPS2.

The recursive SPS algorithm (SPS4)

We establish an analagous identity to (3) for $\Psi_n(z)$.

$$\Psi_n(z) = \prod_{j=1}^{k} \Phi_{m_j}(z^{e_j}). \tag{4}$$

(3) and (4) suggest a recursive method of computing $\Phi_n(z)$. Consider the example of $\Phi_n(z)$, for $n = 1155 = 3 \cdot 5 \cdot 7 \cdot 11$. To obtain the coefficients of $\Phi_{1105}(z)$ by way of SPS3, we construct the product

$$\Psi_{105}(z) \cdot \Psi_{15}(z^{11})\Psi_3(z^{77}) \cdot (1 - z^{385})^{-1} \cdot (1 - z^{231})^{-1} \cdot (1 - z^{165})^{-1} \cdot (1 - z^{105})^{-1} \cdot (1 - z^{1155}) \tag{5}$$

from left to right. However, in light of (4), we know this method computes $\Psi_{105}(z)$ in a wasteful manner. We can treat $\Psi_{105}(z)$ as a product of cyclotomic polynomials of smaller index:

$$\Psi_{105}(z) = \Phi_{15}(z)\Phi_5(z^7)\Phi_1(z^{35}) \tag{6}$$

One could apply (3) yet again, now to $\Phi_{15}(z)$, giving us

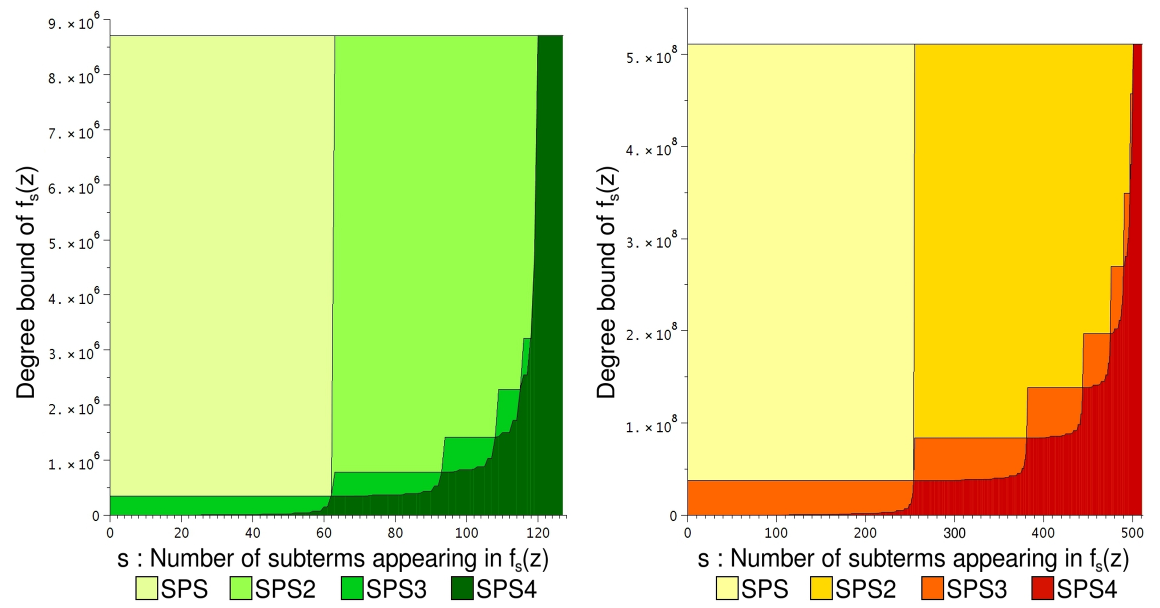
$$\Phi_{15}(z) = \Psi_5(z) \cdot (1 - z^5)^{-1} \cdot (1 - z^3)^{-1} \cdot (1 - z^{15}). \tag{7}$$

Upon computing $\Psi_{105}(z)$, we can break the next term of (5), $\Psi_{15}(z^{11})$ into smaller products in a similar fashion. We effectively compute $\Phi_n(z)$ by recursion into the factors of n. We call this approach the **recursive sparse power series** method or **SPS4**.

A visual comparison of SPS1-4

We show the growth of the degree bound over the computation of $\Phi_{43730115}(z)$ and $\Phi_{3234846615}(z)$. In each version of the SPS2-4 the degree bound is at most that of its predecessor over the computation of $\Phi_n(z)$. The degree bound for SPSk, where $1 \le k \le 4$ is the height of the $4 - (k - 1)$ darkest regions of the plots; moreover, we think of the area of these $4 - (k - 1)$ regions as a heuristic measure of the time cost of SPSk.

Figure 1: Growth of the degree bound of SPS1-4 over the computation of $\Phi_n(z)$ for $n = 43730115$ (left) and $n = 3234846615$ (right)



Timings

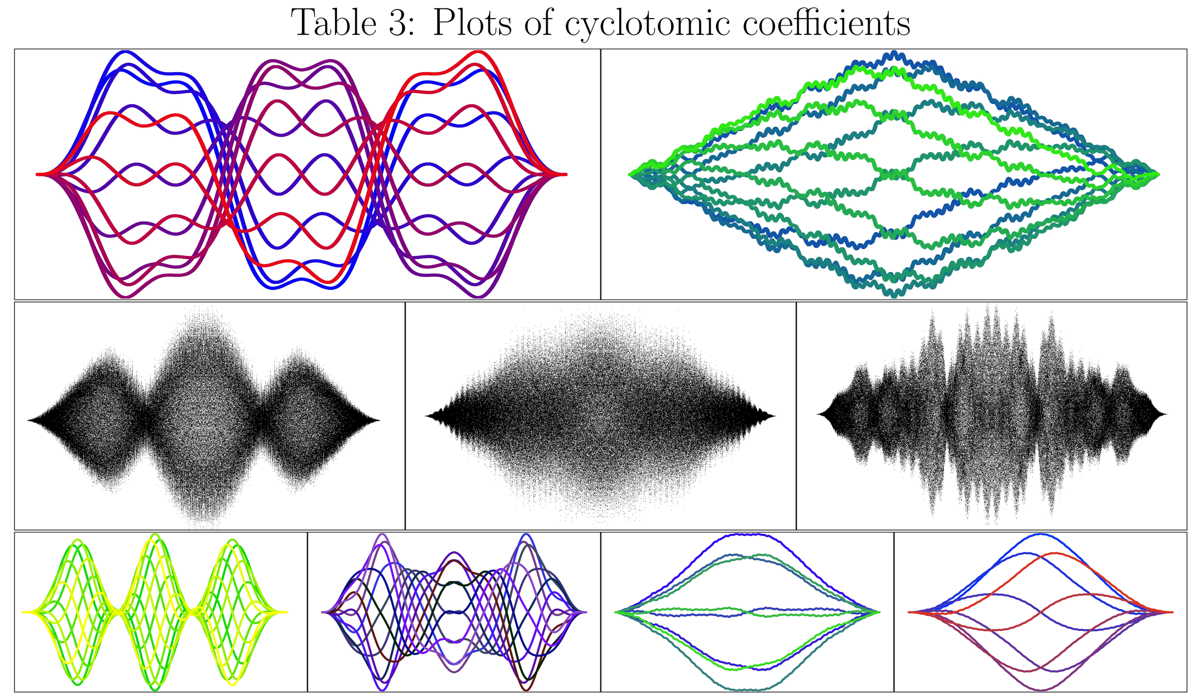
Table 2: Time to calculate $\Phi_n(z)$ (in seconds*)

n	(factorization of n)	FFT	SPS	SPS2	SPS3	SPS4	A(n)
		algorithm					
255255	$= 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17$	0.40	0.00	0.00	0.00	0.00	532
1181895	$= 3 \cdot 5 \cdot 11 \cdot 13 \cdot 19 \cdot 29$	1.76	0.01	0.00	0.00	0.00	14102773
4849845	$= 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19$	7.74	0.12	0.06	0.02	0.01	669606
37182145	$= 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19 \cdot 23$	142.37	1.75	0.95	0.23	0.19	2286541988726
43730115	$= 3 \cdot 5 \cdot 11 \cdot 13 \cdot 19 \cdot 29 \cdot 37$	140.62	1.69	0.93	0.23	0.19	(see table 1)
111546435	$= 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19 \cdot 23$	295.19	6.94	3.88	1.45	0.94	8161018310
1078282205	$= 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19 \cdot 23 \cdot 29$	-	105.61	58.25	12.34	9.29	1558645698271916
3234846615	$= 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19 \cdot 23 \cdot 29$	-	432.28	244.44	81.32	49.18	2888582082500892851

A look at cyclotomic coefficients

The plots in table 3 were produced by plotting a random subset of the terms of $\Phi_n(z)$. We plot degree s on the horizontal axis and the coefficient of the term of degree s on the vertical. The terms of some $\Phi_n(z)$ appear highly structured whereas others exhibit more "noise".

Table 3: Plots of cyclotomic coefficients



Legend: The plots are of coefficients of $\Phi_n(z)$ for select n. 1st row (left to right): $n = 30489595$, $n = 327845$. 2nd row: $n = 4849845$, $n = 111546435$, $n = 3234846615$. 3rd row: $n = 1311052155$, $n = 40324935$, $n = 1181895$, $n = 43730115$.

A challenge problem

The least two integers such that $A(n) > n^4$ only differ by one prime factor. Those are $1880394945 = 43s$ and $2317696095 = 53s$, where $s = 43730115$. We computed $\Phi_n(z)$, where $n = s \cdot 43 \cdot 53 = 99660932085$. A limitation of this problem is memory. Storing the coefficients of $\Phi_n(z)$ as 320-bit integers requires over **750 GB** of space.

To compute $\Phi_n(z)$, we first compute the image of $-\Psi_m(z)$ modulo five 64-bit primes $q_0, \cdots, q_4$, where $m = 1880394945$. We then compute $g_j \bmod q_i$ for $0 \le j < 53, 0 \le i < 5$, where the coefficients of g_j comprise those of terms of $-\Psi_m(z)(1 - z^m)^{-1}$ whose degree

$$\sum_{j=0}^{52} z^j g_j(z^{53}) = \Psi_m(z)(1 - z^m)^{-1} \pmod{z^{\phi(n)/2+1}}, \tag{8}$$

We thus have that, by lemma 6,

$$\sum_{j=0}^{52} z^j g_j(z^{53}) \Phi_m(z^{53}) \equiv \Phi_n(z) \bmod z^{\phi(n)/2+1}. \tag{9}$$

We computed $g_j(z)\Phi_m(z) \bmod q_i$, $0 \le j < 53, 0 \le i < 5$, and reconstructed g_j by Chinese remaindering. We distributed the computation to three desktop computers.

A(99660932085) = 61267208717407836670896202324395260124725254733381530786789617551493787739155364471853 70,

which is roughly $2^{291.6}$ or $n^{7.98}$. The computation took roughly 2 days. Two hard disks valiantly died in previous, naive attempts to compute $\Phi_n(z)$.

References

[1] P. Erdős. On the coefficients of the cyclotomic polynomial. *Bull. Amer. Math. Soc.*, 52:179–184, 1946.

[2] H. Maier. The size of the coefficients of cyclotomic polynomials. In *Analytic number theory, Vol. 2 (Allerton Park, IL, 1995)*, volume 139 of *Progr. Math.*, pages 633–639. Birkhäuser Boston, Boston, MA, 1996.

[3] P. Moree. Inverse cyclotomic polynomials. *J. Number Theory*, 129(3):667–680, 2009.