Faster Homomorphic Function Evaluation using Non-Integral Base Encoding

Charlotte Bonte[1], Carl Bootland[1], Joppe W. Bos[2], Wouter Castryck[1,3], Ilia Iliashenko[1], and Frederik Vercauteren[1,4]

[1] imec-Cosic, Dept. Electrical Engineering, KU Leuven
[2] NXP Semiconductors
[3] Laboratoire Paul Painlevé, Université de Lille-1
[4] Open Security Research

Abstract. In this paper we present an encoding method for fixed-point numbers tailored for homomorphic function evaluation. The choice of the degree of the polynomial modulus used in all popular somewhat homomorphic encryption schemes is dominated by security considerations, while with the current encoding techniques the correctness requirement allows for much smaller values. We introduce a generic encoding method using expansions with respect to a non-integral base, which exploits this large degree at the benefit of reducing the growth of the coefficients when performing homomorphic operations. In practice this allows one to choose a smaller plaintext coefficient modulus which results in a significant reduction of the running time. We illustrate our approach by applying this encoding in the setting of homomorphic electricity load forecasting for the smart grid which results in a speed-up by a factor 13 compared to previous work, where encoding was done using balanced ternary expansions.

1 Introduction

The cryptographic technique which allows an untrusted entity to perform arbitrary computation on encrypted data is known as fully homomorphic encryption. The first such construction was based on ideal lattices and was presented by Gentry in 2009 [19]. When the algorithm applied to the encrypted data is known in advance one can use a *somewhat homomorphic encryption* (SHE) scheme which only allows to perform a limited number of computational steps on the encrypted data. Such schemes are significantly more efficient in practice.

In all popular SHE schemes, the plaintext space is a ring of the form $R_t = \mathbb{Z}_t[X]/(f(X))$, where $t \geq 2$ is a small integer called the coefficient modulus, and $f(X) \in \mathbb{Z}[X]$ is a monic irreducible degree d polynomial called the polynomial

This work was supported by the European Commission under the ICT programme with contract H2020-ICT-2014-1 644209 HEAT, and through the European Research Council under the FP7/2007-2013 programme with ERC Grant Agreement 615722 MOTMELSUM. The second author is also supported by a PhD fellowship of the Research Foundation - Flanders (FWO).

modulus. Usually one lets $f(X)$ be a cyclotomic polynomial, where for reasons of performance the most popular choices are the power-of-two cyclotomics $X^d + 1$ where $d = 2^k$ for some positive integer k, which are maximally sparse. In this case arithmetic in R_t can be performed efficiently using the fast Fourier transform, which is used in many lattice-based constructions (e.g. [6,7,8,30]) and most implementations (e.g. [3,4,5,20,21,25,27]).

One interesting problem relates to the *encoding* of the input data of the algorithm such that it can be represented as elements of R_t and such that one obtains a meaningful outcome after the encrypted result is decrypted and decoded. This means that addition and multiplication of the input data must agree with the corresponding operations in R_t up to the depth of the envisaged SHE computation. An active research area investigates different such encoding techniques, which are often application-specific and dependent on the type of the input data. For the sake of exposition we will concentrate on the particularly interesting and popular setting where the input data consists of finite precision real numbers θ, even though our discussion below is fairly generic. The main idea, going back to Dowlin et al. [16] (see also [17,23,26]) and analyzed in more detail by Costache et al. [14], is to expand θ with respect to a base b

$$\theta = a_r b^r + a_{r-1} b^{r-1} + \cdots + a_1 b + a_0 + a_{-1} b^{-1} + a_{-2} b^{-2} + \cdots + a_{-s} b^{-s} \quad (1)$$

using integer digits a_i, after which one replaces b by X to end up inside the Laurent polynomial ring $\mathbb{Z}[X, X^{-1}]$. One then reduces the digits a_i modulo t and applies the ring homomorphism to R_t defined by

$$\iota : \mathbb{Z}_t[X, X^{-1}] \to R_t : \begin{cases} X & \mapsto X, \\ X^{-1} & \mapsto -g(X) \cdot f(0)^{-1}, \end{cases}$$

where we write $f(X) = Xg(X) + f(0)$ and it is assumed that $f(0)$ is invertible modulo t; this is always true for cyclotomic polynomials, or for factors of them. The quantity $r + s$ will sometimes be referred to as the *degree* of the encoding (where we assume that $a_r, a_{-s} \neq 0$).

Remark 1. For power-of-two-cyclotomics the homomorphism ι amounts to letting $X^{-1} \mapsto -X^{d-1}$, so that the encoding of (1) is given by

$$a_r X^r + a_{r-1} X^{r-1} + \cdots + a_1 X + a_0 - a_{-1} X^{d-1} - a_{-2} X^{d-2} - \cdots - a_{-s} X^{d-s} \ .$$

In fact in [14] it is mentioned that inverting X is only possible in the power-of-two cyclotomic case, but this seems to be overcareful. In particular, contrary to what is claimed there, the above construction is compatible with the SIMD computations described in [16,29].

Decoding is then performed by applying the inverse of the restricted map $\iota|_{\mathbb{Z}_t[X,X^{-1}]_{[-\ell,m]}}$ where

$$\mathbb{Z}_t[X, X^{-1}]_{[-\ell,m]} = \{\, a_m X^m + a_{m-1} X^{m-1} + \ldots + a_{-\ell} X^{-\ell} \,|\, a_i \in \mathbb{Z}_t \text{ for all } i \,\}$$

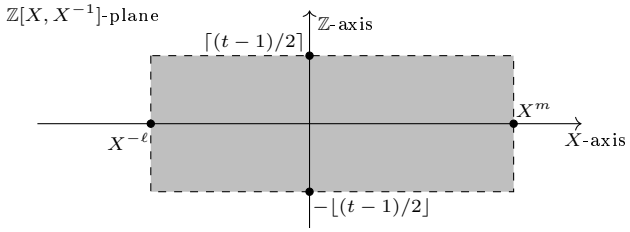


Fig. 1. Box in which to stay during computation, where $\ell + m + 1 = d$.

is a subset of Laurent polynomials whose monomials have bounded exponents. If $\ell+m+1 = d$ then this restriction of ι is indeed invertible as a $\mathbb{Z}_t$-linear map. The precise choice of ℓ, m depends on the data encoded. After applying this inverse, one replaces the coefficients by their representants in $\{-\lfloor (t-1)/2 \rfloor, \ldots, \lceil (t-1)/2 \rceil\}$ to end up with an expression in $\mathbb{Z}[X, X^{-1}]$, and evaluates the result at $X = b$. Ensuring that decoding is correct to a given computational depth places constraints on the parameters t and d, in order to avoid ending up outside the box depicted in Figure 1 if the computation were to be carried out directly in $\mathbb{Z}[X, X^{-1}]$. In terms of R_t we will often refer to this event as the 'wrapping around' of the encoded data modulo t or $f(X)$, although we note that this is an abuse of language. In the case of power-of-two cyclotomics, ending up above or below the box does indeed correspond to wrapping around modulo t, but ending up at the left or the right of the box corresponds to a mix-up of the high degree terms and the low degree terms.

The precise constraints on t and d not only depend on the complexity of the computation, but also on the type of expansion (1) used in the encoding. Dowlin et al. suggest to use balanced b-ary expansions with respect to an odd base $b \in \mathbb{Z}_{\geq 3}$, which means that the digits are taken from $\{-(b-1)/2, \ldots, (b-1)/2\}$. Such expansions have been used for centuries going back at least to Colson (1726) and Cauchy (1840) in the quest for more efficient arithmetic.

If we fix a precision, then for smaller b the balanced b-ary expansions are longer but the coefficients are smaller, this implies the need for a larger d but smaller t. Similarly for larger bases the expansions become shorter but have larger coefficients leading to smaller d but larger t. For the application to somewhat homomorphic encryption considered in [4,14] the security requirements ask for a very large d, so that the best choice is to use as small a base as possible, namely $b = 3$, with digits in $\{\pm 1, 0\}$. Even for this smallest choice the resulting lower bound on t is very large and the bound on d is much smaller than that coming from the cryptographic requirements. To illustrate this, we recall the concrete figures from the paper [4], which uses the Fan-Vercauteren (FV) somewhat homomorphic encryption scheme [18] for privacy-friendly prediction of electricity consumption in the setting of the smart grid. Here the authors use $d = 4096$ for cryptographic reasons, which is an optimistic choice that leads to 80-bit security only (and maybe even slightly less [1]). On the other hand using

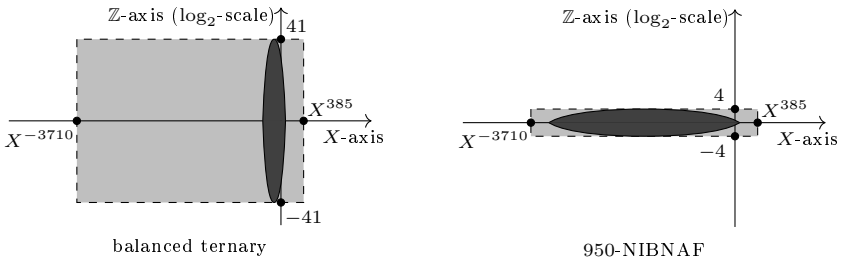


Fig. 2. Comparison between the amount of plaintext space which is actually used in the setting of [4], where $d = 4096$. More precise figures to be found in Section 4.

balanced ternary expansions, correct decoding is guaranteed as soon as $d \geq 368$, which is even a conservative estimate. This eventually leads to the huge bound $t \gtrapprox 2^{107}$, which is overcome by decomposing R_t into 13 factors according to the Chinese Remainder Theorem (CRT). This is then used to homomorphically forecast the electricity usage for the next half hour for a small apartment complex of 10 households in about half a minute, using a sequential implementation.

The discrepancy between the requirements coming from correct decoding and those coming from security considerations suggests that other possible expansions may be better suited for use with SHE. In this paper we introduce a generic encoding technique, using very sparse expansions having digits in $\{\pm 1, 0\}$ with respect to a *non-integral* base $b_w > 1$, where w is a sparseness measure. These expansions will be said to be of 'non-integral base non-adjacent form' with window size w, abbreviated to w-NIBNAF. Increasing w makes the degrees of the resulting Laurent polynomial encodings grow and decreases the growth of the coefficients when performing operations; hence lowering the bound on t. Our encoding technique is especially useful when using fixed-point real numbers, but could also serve in dealing with fixed-point complex numbers or even with integers, despite the fact that b_w is non-integral (this would require a careful precision analysis which is avoided here).

We demonstrate that this technique results in significant performance increases by re-doing the experiments from [4]. Along with a more careful precision analysis which is tailored for this specific use case, using 950-NIBNAF expansions we end up with the dramatically reduced bound $t \geq 33$. It is not entirely honest to compare this to $t \gtrapprox 2^{107}$ because of our better precision analysis; as explained in Section 4 it makes more sense to compare the new bound to $t \gtrapprox 2^{42}$, but the reduction remains huge. As the reader can see in Figure 2 this is explained by the fact that the data is spread more evenly across plaintext space during computation. As a consequence we avoid the need for CRT decomposition and thus reduce the running time by a factor 13, showing that the same homomorphic forecasting can be done in only 2.5 seconds.

Remark 2. An alternative recent proposal for encoding using a non-integral base can be found in [13], which targets efficient evaluation of the discrete Fourier

transform on encrypted data. Here the authors work exclusively in the power-of-two cyclotomic setting $f(X) = X^d + 1$, and the input data consists of complex numbers θ which are expanded with respect to the base $b = \zeta$, where ζ is a primitive $2d$-th root of unity, i.e. a root of $f(X)$. One nice feature of this approach is that the correctness of decoding is not affected by wrapping around modulo $f(X)$. To find a sparse expansion they use the LLL algorithm [24], but unfortunately for arbitrary complex inputs the digits become rather large, at least when compared to w-NIBNAF. An alternative viewpoint on this method is to find an element of R_t having small coefficients which under the canonical embedding has one known component that approximates θ. In this sense the method is very similar to that from [10] where they use ciphertext packing and encode d complex numbers into a single element of R_t which under the canonical embedding returns the given complex numbers, up to a predetermined scalar. But again, to achieve this, the coefficients must be in general quite large.

2 Encoding data using w-NIBNAF

2.1 The non-adjacent form with window size w

One first approach to try to reduce the lower bound on t is by using encodings for which many of the coefficients are zero. One way to achieve this is by using the non-adjacent form (NAF) representation which was introduced by Reitweisner in 1960 for speeding up early multiplication algorithms [28].

Definition 1. *The* non-adjacent form (NAF) representation *of a real number θ is an expansion of θ to the base $b = 2$ with coefficients in $\{-1, 0, 1\}$ such that any two adjacent coefficients are not both non-zero.*

Note that NAF representations always exist and are unique, modulo periodic infinite expansion issues such as

$$2^0 + 2^{-2} + 2^{-4} + 2^{-6} + 2^{-8} + \ldots = 2^1 - 2^{-1} - 2^{-3} - 2^{-5} - \ldots$$

This representation can be generalized, for an integer $w \geq 1$ (called the 'window size') one can ensure that in any window of w consecutive coefficients at most one of them is non-zero. This is possible to base $b = 2$ but for $w > 2$ one requires larger coefficients. This generalization is called w-NAF and was first considered by Cohen et al. [11].

Definition 2. *Let $w \geq 1$ be an integer. A* w-NAF representation *of a real number θ is an expansion of θ with base 2 and whose non-zero coefficients are odd and less that 2^{w-1} in absolute value such that for every set of w consecutive coefficients at most one of them is non-zero.*

Just as for NAF representations, the w-NAF representation is essentially unique. Further, we see that NAF is just the special case of w-NAF for $w = 2$. Unfortunately, due to the fact that the coefficients are taken from a much larger set, using w-NAF encodings in the SHE setting actually gives larger bounds on both t and d for increasing w. This is not useful in the setting of SHE when the goal is to reduce the parameter sizes and the running time of the algorithm.

2.2 The non-integral base when computing a non-adjacent form with window size w

Ideally, we want the coefficients in our expansions to be members of $\{\pm 1, 0\}$ with many equal to 0, as this would lead to the slowest growth in coefficient sizes, allowing us to use smaller values for t. This would come at the expense of using longer encodings, but remember that we have a lot of manoeuvring space on the d side. One way to achieve this is to use a non-integral base $b > 1$ when computing a non-adjacent form. We first give the definition of a non-integral base non-adjacent form with window size w (w-NIBNAF) representation and then explain where this precise formulation comes from.

Definition 3. *A sequence $a_0, a_1, \ldots, a_n, \ldots$ is a* w-balanced ternary sequence *if it has $a_i \in \{-1, 0, 1\}$ for $i \in \mathbb{Z}_{\geq 0}$ and satisfies the property that each set of w consecutive terms has no more than one non-zero term.*

Definition 4. *Let $\theta \in \mathbb{R}$ and $w \in \mathbb{Z}_{>0}$. Define b_w to be the unique positive real root of the polynomial*

$$F_w(x) = x^{w+1} - x^w - x - 1.$$

A w-balanced ternary sequence $a_r, a_{r-1}, \ldots, a_1, a_0, a_{-1}, \ldots$ is a w-NIBNAF representation of θ if

$$\theta = a_r b_w^r + a_{r-1} b_w^{r-1} + \cdots + a_1 b_w + a_0 + a_{-1} b_w^{-1} + \cdots \ .$$

Of course, a priori it may be possible that a given θ has no such w-NIBNAF representation or it may have (infinitely) many of them. We will show that every θ has at least one such w-NIBNAF representation and provide an algorithm to find such a representation. However, let us first state a lemma which shows that b_w is well-defined for $w \geq 1$.

Lemma 1. *For an integer $w \geq 1$ the polynomial $F_w(x) = x^{w+1} - x^w - x - 1$ has a unique positive real root $b_w > 1$. The sequence $b_1, b_2, \ldots$ is strictly decreasing and the limit as w tends to infinity of b_w is 1. Further, $(x^2 + 1) \mid F_w(x)$ for $w \equiv 3 \bmod 4$.*

The proof is straightforward and given in Appendix A. We give the first few values of b_w and note that b_3 is the golden ratio ϕ:

$$\begin{array}{ll} b_1 = 1 + \sqrt{2} \approx 2.414214, & b_2 \approx 1.839287, \\ b_3 = \frac{1}{2}(1 + \sqrt{5}) \approx 1.618034, & b_4 \approx 1.497094, \\ b_5 \approx 1.419633, & b_6 \approx 1.365255. \end{array}$$

Since we are using a non-integral base, a w-NIBNAF representation of a fixed-point number has infinitely many non-zero terms in general. Obviously, this is not practical since one needs to store each non-zero coefficient. In order to overcome this problem one can approximate the fixed-point number by terminating the w-NIBNAF representation after some power of the base. We denote

such a terminated sequence an *approximate w-NIBNAF representation*. There are two straightforward ways of achieving this: either the power of the base used to determine the termination is chosen in advance which gives an easy bound on the maximal possible error created, or we choose a maximal allowed error in advance and terminate after the first power which gives error less than or equal to this pre-determined value.

2.3 Encoding and decoding using w-NIBNAF

The process of encoding works as described in the introduction, i.e. we follow the approach from [14,16] except we use an approximate w-NIBNAF representation instead of the balanced ternary representation. That is, to encode a fixed-point number θ we find an approximate w-NIBNAF representation of θ with small enough error and replace each occurrence of b_w by X, after which we apply the map ι to end up with an element of the plaintext space R_t. Decoding is almost the same as well, only that after inverting ι and lifting the coefficients to $\mathbb{Z}$ we evaluate the resulting Laurent polynomial at $X = b_w$ rather than $X = 3$, computing the value only to the required precision. Rather than evaluating directly it is best to reduce the Laurent polynomial modulo $F_w(X)$ (or modulo the polynomial $F_w(X)/(X^2+1)$ if $w \equiv 3 \bmod 4$) so that we only have to compute powers of b_w up to w (respectively $w-2$). As we encode using approximate representations, there can be many encodings which decode, within a certain precision, to the same value.

Let us prove that every $\theta \in \mathbb{R}$ has a w-NIBNAF representation: Algorithm 1 produces such a representation. Algorithm 1 is a greedy algorithm which chooses the closest signed power of the base to θ and then iteratively finds a representation of the difference. Except when θ can be written as $\theta = h(b_w)/b_w^q$, for some polynomial h with coefficients in $\{\pm 1, 0\}$ and $q \in \mathbb{Z}_{\geq 0}$, any w-NIBNAF representation is infinitely long. Hence, we must terminate Algorithm 1 once the iterative input is smaller than some pre-determined precision $\epsilon > 0$.

We now prove that the algorithm works as required.

Lemma 2. *Algorithm 1 produces an approximate w-NIBNAF representation of θ with an error of at most ϵ.*

Proof. Assuming that the algorithm terminates, the output clearly represents θ to within an error of at most size ϵ. First we show that the output is w-NIBNAF. Suppose that the output, on input θ, b_w, ϵ, has at least two non-zero terms, the first being a_d. This implies either that $b_w^d \leq |\theta| < b_w^{d+1}$ and $b_w^{d+1} - |\theta| > |\theta| - b_w^d$ or $b_w^{d-1} < |\theta| \leq b_w^d$ and $b_w^d - |\theta| \leq |\theta| - b_w^{d-1}$. These conditions can be written as $b_w^d \leq |\theta| < \frac{1}{2} b_w^d (1 + b_w)$ and $\frac{1}{2} b_w^{d-1}(1 + b_w) \leq |\theta| \leq b_w^d$ respectively. This shows that

$$||\theta| - b_w^d| < \max\left\{b_w^d - \tfrac{1}{2} b_w^{d-1}(1 + b_w),\ \tfrac{1}{2} b_w^d (1 + b_w) - b_w^d\right\} = \tfrac{1}{2} b_w^d (b_w - 1)\ .$$

The algorithm subsequently chooses the closest power of b_w to this smaller value, suppose it is b_w^ℓ. By the same argument with θ replaced by $|\theta| - b_w^d$ we have that

Algorithm 1: `GreedyRepresentation`

Input: θ – the fixed-point number to be represented,
b_w – the w-NIBNAF base to be used in the representation,
ϵ – the precision to which the representation is determined.
Output: An approximate w-NIBNAF representation $a_r, a_{r-1}, \ldots$ of θ with error less than ϵ, where $a_i = 0$ if not otherwise specified.

```
while |θ| > ε do
    σ ← sgn(θ)
    t ← σθ
    r ← ⌈log_{b_w}(t)⌉
    if b_w^r − t > t − b_w^{r−1} then
        r ← r − 1
    a_r ← σ
    θ ← θ − σ b_w^r
Return (a_i).
```

either $b_w^\ell \leq \left||\theta| - b_w^d\right|$ or $\frac{1}{2}b_w^{\ell-1}(1+b_w) \leq \left||\theta| - b_w^d\right|$ and since b_w^ℓ is larger than $\frac{1}{2}b_w^{\ell-1}(1+b_w)$ the maximal possible value of ℓ, which we denote by $\ell_w(d)$, satisfies

$$\ell_w(d) = \max\left\{\ell \in \mathbb{Z} \;\middle|\; \tfrac{1}{2}b_w^{\ell-1}(1+b_w) < \tfrac{1}{2}b_w^d(b_w-1)\right\} \quad .$$

The condition on ℓ can be rewritten as $b_w^\ell < b_w^{d+1}(b_w-1)/(b_w+1)$ which implies that $\ell < d+1+\log_{b_w}((b_w-1)/(b_w+1))$ and thus

$$\ell_w(d) = d + \left\lceil \log_{b_w}\left(\frac{b_w-1}{b_w+1}\right)\right\rceil \quad ,$$

so that the smallest possible difference is independent of d and equal to

$$s(w) := d - \ell_w(d) = -\left\lceil \log_{b_w}\left(\frac{b_w-1}{b_w+1}\right)\right\rceil = \left\lfloor \log_{b_w}\left(\frac{b_w+1}{b_w-1}\right)\right\rfloor \quad .$$

We thus need to show that $s(w) \geq w$. As w is an integer this is equivalent to

$$\log_{b_w}\left(\frac{b_w+1}{b_w-1}\right) \geq w \iff b_w^w \leq \frac{b_w+1}{b_w-1} \iff b_w^{w+1} - b_w^w - b_w - 1 \leq 0$$

which holds for all w since $F_w(b_w) = 0$. We point out that our algorithm works correctly and deterministically because when $|\theta|$ is exactly half-way between two powers of b_w we choose the larger power. This shows that the output is of the required form.

Finally, to show that the algorithm terminates we note that the k'th successive difference is bounded above by $\frac{1}{2}b_w^{d-(k-1)s(w)}(b_w-1)$ and this tends to 0 as k tends to infinity. Therefore after a finite number of steps (at most $\left\lceil \left(d - \log_{b_w}\left(2\epsilon/(b_w-1)\right)\right)/s(w)\right\rceil + 1$) the difference is smaller than or equal to ϵ and the algorithm terminates. □

In the limit as ϵ tends to zero we reach a w-NIBNAF representation of θ hence we have proven that any real number indeed admits a w-NIBNAF representation. Clearly we can also use this algorithm to encode θ by instead returning $\sum_i a_i X^i$, this gives an encoding of θ with maximal error ϵ. Since the input θ of the algorithm can get arbitrarily close to but larger than ϵ, the final term in the encoding can be $\pm X^h$ where $h = \lfloor \log_{b_w}(2\epsilon/(1+b_w)) \rfloor + 1$. If we are to ensure that the smallest power of the base to appear in any approximate w-NIBNAF representation is b_w^s then we require that if b_w^{s-1} is the nearest power of b_w to the input θ then $|\theta| \leq \epsilon$ so that we must have $\frac{1}{2} b_w^{s-1}(1+b_w) \leq \epsilon$ which implies the smallest precision we can achieve is $\epsilon = b_w^{s-1}(1+b_w)/2$. In particular if we want 'polynomial' encodings then the best precision possible using the greedy algorithm is $(1+b_w^{-1})/2 < 1$.

Remark 3. If in Algorithm 1 one replaces b_w by a smaller base $b > 1$ then it still produces a w-NIBNAF expansion to the desired precision: this follows easily from the proof of Lemma 2. The distinguishing feature of b_w is that it is maximal with respect to this property, so that the resulting expansions become as short as possible.

3 Analysis of coefficient growth when computing with encodings

After encoding the input data it is ready for homomorphic computations. This increases both the number of non-zero coefficients as well as the size of these coefficients. Since we are working in the ring R_t there is a risk that our data wraps around modulo t as well as modulo $f(X)$, in the sense explained in the introduction, which we should avoid since this leads to erroneous decoding. Therefore we need to understand the coefficient growth more thoroughly. We simplify the analysis in this section by only considering multiplications and what constraint this puts on t, it is then not hard to generalize this to include additions.

3.1 Worst case coefficient growth for w-NIBNAF encodings

Here we analyze the maximal possible size of a coefficient which could occur from computing with w-NIBNAF encodings. As fresh w-NIBNAF encodings are just approximate w-NIBNAF representations written as elements of R_t we consider finite w-balanced ternary sequences and the multiplication endowed on them from R_t or from $\mathbb{Z}_t[X, X^{-1}]$. Further, as we ensure in practice that there is no wrap around modulo $f(X)$ this can be ignored in our analysis.

To start the worst case analysis we have the following lower bound.

Lemma 3. *A lower bound on the maximal absolute size of a term that can be produced by taking the product of p arbitrary w-balanced ternary sequences of length $d+1$ is*

$$B_w(d,p) := \sum_{k=0}^{\lfloor \lfloor p\lfloor d/w \rfloor/2 \rfloor / (\lfloor d/w \rfloor + 1) \rfloor} (-1)^k \binom{p}{k} \binom{p - 1 + \lfloor p\lfloor d/w \rfloor/2 \rfloor - k\lfloor d/w \rfloor - k}{p-1} .$$

A full proof of this lemma is given in Appendix A but the main idea is to look at the largest coefficient of m^p where m has the maximal number of non-zero coefficients, $\lfloor d/w \rfloor + 1$, all being equal to 1 and with exactly $w-1$ zero coefficients between each pair of adjacent non-zero coefficients. We note that the w-NIBNAF encoding, using the greedy algorithm with precision $\frac{1}{2}$, of $b_w^{d+w-(d \bmod w)}(b_w-1)/2$ is m so in practice this lower bound is achievable although unlikely to occur.

We expect that this lower bound is tight, indeed we were able to prove the following lemma, the proof is also given in Appendix A.

Lemma 4. *Suppose w divides d, then $B_w(d,p)$ equals the maximal absolute size of a term that can be produced by taking the product of p arbitrary w-balanced ternary sequences of length $d+1$.*

We thus make the following conjecture which we assume to be true.

Conjecture 1 *The lower bound $B_w(d,p)$ given in Lemma 3 is exact for all d, that is the maximal absolute term size which can occur after multiplying p arbitrary w-balanced ternary sequences of length $d+1$ is $B_w(d,p)$.*

This conjecture seems very plausible since as soon as one multiplicand does not have non-zero coefficients exactly w places apart the non-zero coefficients start to spread out and decrease in value.

3.2 Approximating $B_w(d,p)$

To approximate $B_w(d,p)$ for fixed p define $n := \lfloor d/w \rfloor + 1$, then for suitably large n (so that the variable k varies over a range only dependent on p) we can expand the expression for $B_w(d,p)$ as a 'polynomial' in n of degree $p-1$, see Appendix B for the details. The expressions we find are in fact valid for all n, the first few are:

$$
\begin{aligned}
&B_w(d,1) = 1; && B_w(d,2) = n;\\
&B_w(d,3) = \tfrac{1}{8}(6n^2+1) - \tfrac{(-1)^n}{8}; && B_w(d,4) = \tfrac{1}{3}(2n^3+n);\\
&B_w(d,5) = \tfrac{1}{384}(230n^4+70n^2+27) - \tfrac{(-1)^n}{384}(30n^2+27);\\
&B_w(d,6) = \tfrac{1}{20}(11n^5+5n^3+4n);\\
&B_w(d,7) = \tfrac{1}{23040}(11774n^6+4235n^4+2261n^2+1125)\\
&\qquad\qquad - \tfrac{(-1)^n}{23040}(1155n^4+1365n^2+1125)\\
&B_w(d,8) = \tfrac{1}{315}(151n^7+70n^5+49n^3+45n).
\end{aligned}
$$

Denoting the coefficient of n^{p-1} in these expressions by ℓ_p, it can be shown (see [2] or Appendix B) that $\lim_{p\to\infty}\sqrt{p}\ell_p = \sqrt{6/\pi}$ and hence we have

$$\lim_{p\to\infty} \log_2(B_w(d,p)) - (p-1)\log_2(n) + \tfrac{1}{2}\log_2\left(\tfrac{\pi p}{6}\right) = 0$$

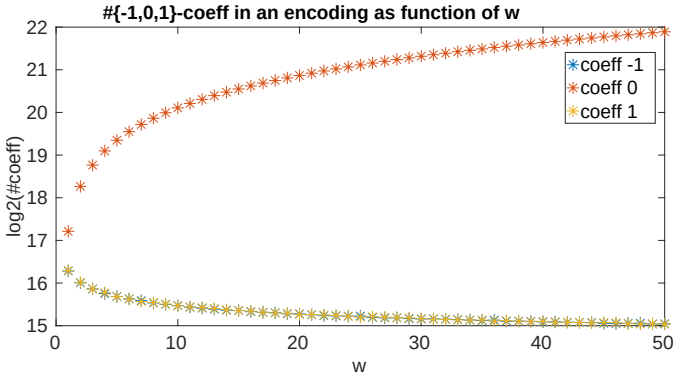


Fig. 3. Plot of $\log_2(\#\text{coeff})$ on the vertical axis against w on the horizontal axis for w-NIBNAF encodings of random integers in $\left[-2^{40}, 2^{40}\right]$.

or equivalently $B_w(d,p) \sim_p \sqrt{6/\pi p} n^{p-1}$. Thus we have the approximation

$$\log_2(B_w(d,p)) \approx (p-1)\log_2(n) - \tfrac{1}{2}\log_2\left(\tfrac{\pi p}{6}\right)$$

which for large enough n (experimentally we found for $n > 1.825\sqrt{p-1/2}$) is an upper bound for $p > 2$. For a guaranteed upper bound when $p > 2$ we have the result $B_w(d,p) \leq \sqrt{6/(\pi p(n^2-1))} n^p$.

3.3 Statistical analysis of the coefficient growth

Based on the w-NIBNAF encodings of random numbers in $N \in \left[-2^{40}, 2^{40}\right]$, we try to get an idea of the amount of -1, 0 and 1 coefficients in a fresh encoding without fractional part, obtained by running Algorithm 1 to precision $(1+b_w^{-1})/2$. We also analyze how these proportions change when we perform multiplications. We plot this for different values of w to illustrate the positive effects of using sparser encodings.

We know from the definition of a w-NIBNAF expansion that at least $w-1$ among each block of w consecutive coefficients of the expansion will be 0, so we expect for big w that the 0 coefficient occurs a lot more than -1 or 1. This is clearly visible in Figure 3. In addition we see an increasing number of 0 coefficients and decreasing number of -1 and 1 coefficients for increasing w. Hence we can conclude that both the absolute and the relative sparseness of our encodings increase as w increases.

Since the balanced ternary encoding of [14,16] and the 2-NAF encoding [28], only have coefficients in $\{-1, 0, 1\}$ it is interesting to compare them to 1-NIBNAF and 2-NIBNAF respectively. We compare them by computing the percentage of coefficients which are equal to -1, 0 and 1 respectively, in 10 000 encodings of random integers N in $\left[-2^{40}, 2^{40}\right]$. We compute this percentage up to an accuracy of 10^{-4} and consider for our counts all coefficients up to and including the

	balanced ternary	1-NIBNAF	2-NAF	2-NIBNAF
percentage of -1s	0.3387	0.2556	0.1739	0.1471
percentage of 0s	0.3225	0.4869	0.6523	0.7046
percentage of 1s	0.3389	0.2575	0.1738	0.1483

Table 1. Comparison between the previous encoding techniques and w-NIBNAF

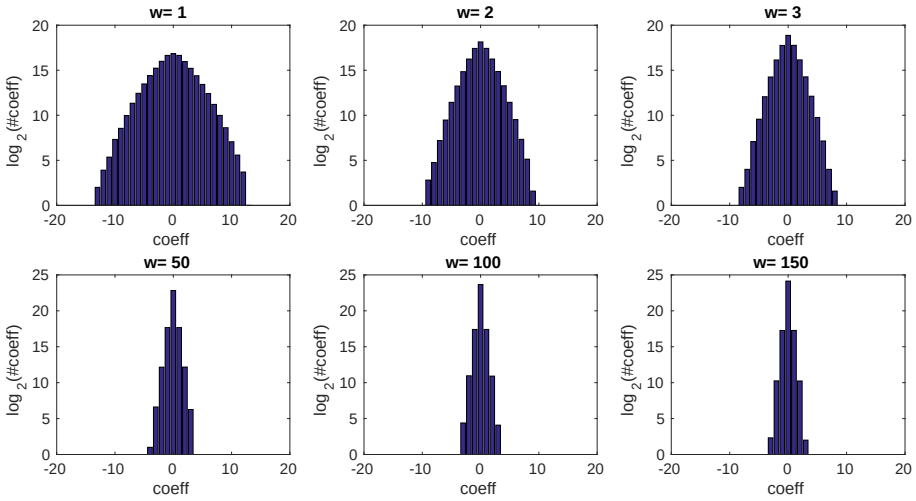


Fig. 4. Plot of $\log_2(\#\text{coeff})$ on the vertical axis against the respective value of the coefficient on the horizontal axis for the result of a multiplication of two w-NIBNAF encodings of random numbers between $\left[-2^{40}, 2^{40}\right]$.

leading coefficient, further zero coefficients are not counted. When we compare the percentages of -1, 0 and 1 coefficients occurring in 1-NIBNAF and balanced ternary in Table 1 we see that for the balanced ternary representation, the occurrences of -1, 0 and 1 coefficients are approximately the same, while for 1-NIBNAF the occurrence of 0 coefficients is bigger than the occurrence of -1 and 1 coefficients. Hence we can conclude that the encodings with this new base will be sparser than the balanced ternary encodings even though the window size is equal. For 2-NIBNAF we also see an improvement in terms of sparseness of the encoding compared to 2-NAF.

The next step is to investigate what happens to the coefficients when we multiply two encodings. From Figure 4 we see that when w increases the maximal size of the resulting coefficients becomes smaller. So the plots confirm the expected result that sparser encodings lead to a reduction in the size of the resulting coefficients after one multiplication.

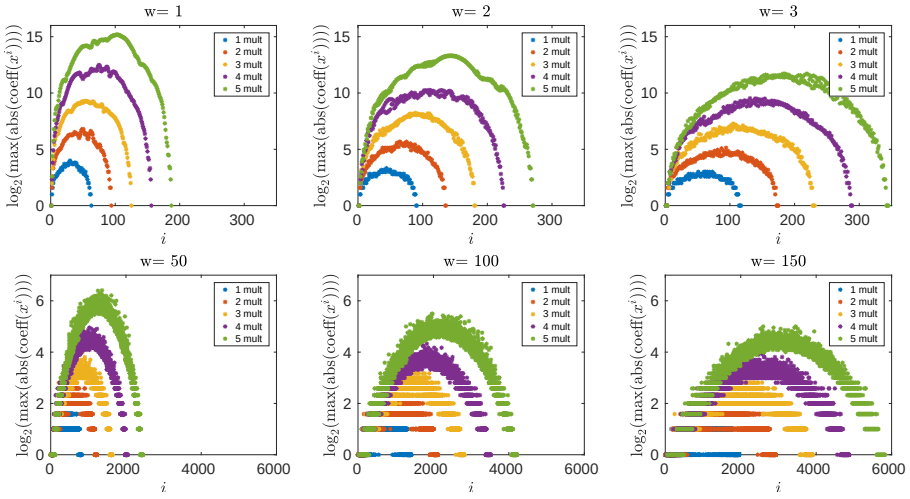


Fig. 5. Plot of the $\log_2$ of the maximum of the absolute value of the coefficient of x^i on the vertical axis against i on the horizontal axis.

Next, we investigate the behaviour for an increasing amount of multiplications. In Figure 5 one observes that for a fixed number of multiplications the maximum coefficient, considering all coefficients in the resulting polynomial, decreases as w increases and the maximum degree of the polynomial increases as w increases. This confirms that increasing the degree of the polynomial, in order to make it more sparse, has the desirable effect of decreasing the size of the coefficients. Figure 5 also shows that based on the result of one multiplication we can even estimate the maximum value of the average coefficients of x^i for a specific number of multiplications by scaling the result for one multiplication.

To summarize, we plot the number of bits of the maximum coefficient of the polynomial that is the result of a certain fixed amount of multiplications as a function of w in Figure 6. From this figure we clearly see that the maximal coefficient decreases when w increases and hence the original encoding polynomial is sparser. In addition we see that the effect of the sparseness of the encoding on the size of the resulting maximal coefficient is bigger when the amount of multiplications increases. However the gain of sparser encodings decreases as w becomes bigger. Furthermore, Figure 6 shows that the bound given in Lemma 3 is much bigger than the average upper bound we get from 10 000 samples.

Remark 4. Since the w-NIBNAF encodings produced by Algorithm 1 applied to $-N$ and N are obtained from one another by changing all the signs, the coefficients -1 and 1 must be distributed evenly, as we indeed observe. This is good, because it typically leads to the maximal amount of cancellation possible during computation. While this does not affect our worst case analysis from Section 3.1, in practice where the worst cases are extremely unlikely, this allows for a considerable reduction of the size of the coefficient modulus t. This is

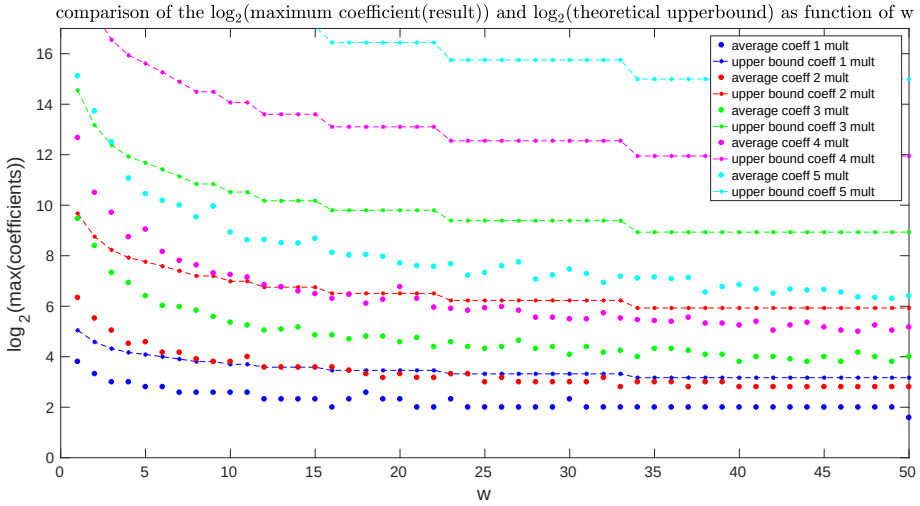


Fig. 6. Plot of the $\log_2$ of the maximum coefficient of the resulting polynomial on the vertical axis against w on the horizontal axis.

implicitly used in the next section. If in some application the input encodings happen to be biased towards 1 or -1 then it might help to work with respect to the *negative* base $-b_w < -1$, by switching the signs of all the digits that appear at an odd index.

4 Practical impact

The size of the plaintext modulus might have a significant impact on the performance of a homomorphic algorithm. In this section we demonstrate that switching to using w-NIBNAF encodings enhances the practical performance of a homomorphic forecasting algorithm by a factor 13.

Being evaluated homomorphically any arithmetic circuit encounters the following constraints while using polynomial encodings of real numbers. The first constraint comes from the correctness requirement of an underlying SHE scheme. Namely, the noise inside the ciphertext should not exceed some level during the computations, otherwise decryption fails. In this context, an increase to the plaintext modulus expands the noise and this places an upper bound on the possible t which can be used. The second constraint does not relate to SHE but to the circuit itself. After any arithmetic operation the polynomial coefficients tend to grow. Given that fact, one should take a big enough plaintext modulus in order to prevent or mitigate possible wrapping around modulo t. This determines the lower bound on range of possible values of t.

In practice, for deep enough circuits these two constraints do not juxtapose, i.e. there is no interval where t can be chosen. However, the plaintext space R_t can be split into smaller rings $R_{t_1}, \ldots, R_{t_k}$ with $t = \prod_{i=1}^{k} t_i$ using the Chinese

Remainder Theorem (CRT). This technique [6] allows us to take the modulus big enough for correct evaluation of the circuit and then perform k threads of the homomorphic algorithm over $\{R_{t_i}\}_i$. As a result, these k output polynomials will be combined into the final output, again by CRT. This approach needs k times more memory and time than the case of a single modulus. Hence, the problem is how to reduce the number of factors of t. The plaintext modulus can be defined for any arithmetic circuit using the worst case scenario in which the final output has the maximal possible coefficient. However, this case occurs in practice with a negligible probability that decreases for circuits of a bigger multiplicative depth.

In this section we show that for practical applications one can take t to be smaller than that given by the worst case. This is based on the fact that for a given t one can approximate the probability of a circuit evaluating incorrectly. This probability becomes negligible for a large enough plaintext modulus. Moreover, we can allow some coefficients to wrap around modulo t with no harm to the final results as long as they are one of the least significant coefficients of the fractional part.

One of the experimental environments recently studied in the SHE setting [4,9,17] is that of artificial neural networks (ANNs). Being a statistical tool, ANNs often deal with real numbers. Thus, for homomorphic evaluation they need to convert real input values and internal parameters into elements of the plaintext space of an underlying SHE scheme. The main obstacle to the SHE-friendly use of ANNs consists in the highly non-linear functions inherent within their structure. One way to overcome this problem is to replace those non-linear functions with quadratic polynomials [17] such that the resulting network will be expressed by a polynomial with a reasonable degree.

Proposed in 1970 [22], the group method of data handling (GMDH) addresses the fitting task as well. In addition, it has a simpler structure than ANNs avoiding many additions during evaluation. Recently this method was applied in the homomorphic setting together with the balanced ternary expansion [4] in order to forecast electricity consumption using smart meters. Due to the fact that 80 percent of electricity meter devices in the European Union should be replaced with smart meters by 2020, this application may mitigate some emerging privacy and efficiency issues.

4.1 The group method of data handling (GMDH)

The basic version of the GMDH algorithm consists in creating a neural network-like structure (see Figure 7) where each node contains a bivariate quadratic polynomial

$$\nu_{ij} : \mathbb{R}^2 \to \mathbb{R} : (x, y) \mapsto b_{ij0} + b_{ij1}x + b_{ij2}y + b_{ij3}xy + b_{ij4}x^2 + b_{ij5}y^2.$$

Indeed, each node has only two input parameters which is the main simplification in comparison with conventional ANNs. The output node is expressed by a polynomial that approximates the target function depending on data points $x_1, \ldots, x_{n_0}$. Henceforth, we refer to such a structure as the GMDH network.

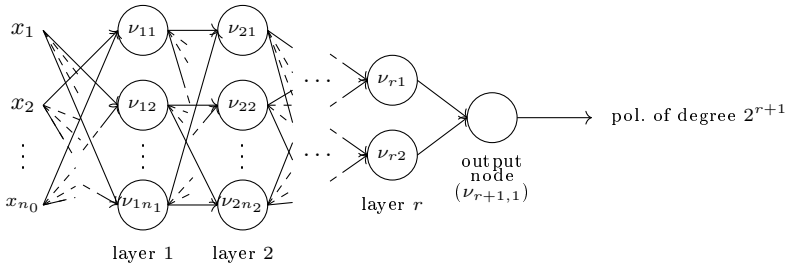


Fig. 7. GMDH network.

The learning algorithm constructs the GMDH network layer by layer in the following way. Before starting the learning process one should set up the number of nodes n_i for each layer and an error function that will help to sort nodes. Those prerequisites are often called *hyperparameters*. Then the learning algorithm looks for polynomial coefficients b_{ijk} of each node of the next layer using the output of the previous one.

For the first layer the algorithm constructs nodes corresponding to all pairs of input values. Each node represents the linear regression problem determined by the equation

$$\mathbf{O} = \mathbf{bA} + \mathbf{e},$$

where $\mathbf{A} = (1, x, y, xy, x^2, y^2)^\intercal$, $\mathbf{b} = (b_{ij0}, b_{ij1}, b_{ij2}, b_{ij3}, b_{ij4}, b_{ij5})$, $\mathbf{O}$ is the expected output and $\mathbf{e}$ is a random noise. The coefficient vector $\mathbf{b}$ can be found with standard statistical tools, e.g. the least squares method. As a result, every node has an assigned output of its polynomial together with the corresponding error estimation. According to this error one excludes the worst $\binom{n_{i-1}}{2} - n_i$ nodes to build the layer. As already stated, this procedure is then repeated for the next layer.

4.2 Experimental setup

To perform experiments we followed the same framework as in [4]. We use real world measurements obtained from the smart meter electricity trials performed in Ireland [12]. This dataset [12] contains observed electricity consumption over 5000 residential and commercial buildings during 30 minute intervals. We used aggregated consumption data of 10 buildings. Given previous consumption data with some additional information, the GMDH network has the goal of predicting electricity demand for the next time period. In particular, it requires 51 input parameters: the 48 previous measurements plus the day of the week, the month and the temperature. The number of hidden layers r is equal to 3 with 8, 4, 2 nodes, respectively (as specified and used in [4]). A single output node provides the electricity consumption prediction for the next half hour.

We encode the input data, given as fixed-point numbers, using approximate w-NIBNAF representations with a fixed number of integer and fractional digits. When increasing the window size w one should take into account that the

precision of the corresponding encodings changes as well. To maintain the same accuracy of the algorithm it is important to keep the precision fixed, hence for bigger w's the smaller base b_w may cause an overflow in the number of integer digits needed for an encoding. Thus, one should increase the number of coefficients used by an encoding.

Starting with the balanced ternary expansion (BTE) and NAF expansions, for any $w > 2$, the numbers $\ell(w)_i$ and $\ell(w)_f$ of integer and fractional digits should be expanded according to the following formula

$$\ell(w)_i = (\ell(\mathrm{BTE})_i - 1) \cdot \log_{b_w} 3 + 1, \qquad \ell(w)_f = -\lfloor \log_{b_w} e_f \rfloor,$$

where e_f is the maximal error of an approximate w-NIBNAF representation such that the prediction algorithm preserves the same accuracy. Empirically we found that the GMDH network demonstrates reasonable absolute and relative errors when $\ell(\mathrm{BTE})_i^{\mathrm{inp}} = 4$ and $e_f^{\mathrm{inp}} = 1$ for the input and $\ell(\mathrm{BTE})_i^{\mathrm{pol}} = 2$ and $e_f^{\mathrm{pol}} = 0.02032$ for polynomial coefficients of ν_{ij}.

Finally, we set the polynomial ring $R_t = \mathbb{Z}_t[X]/(X^{4096} + 1)$ according to the security level 80 of the underlying SHE scheme (in this case the scheme due to Fan and Vercauteren [18] is used). The degree of the ring constrains the multiplicative depth of the algorithm. In particular, the integer and fractional parts may juxtapose because the maximal position of a non-zero integer and fractional coefficients come closer together after each multiplication. Once the integer and fractional parts have started to overlap it is no longer possible to decode correctly.

4.3 Results

The results reported in this section are obtained running the same software and hardware as in [4]: namely, FV-NFLlib software library [15] running on a laptop equipped with an Intel Core i5-3427U CPU (running at 1.80GHz). We performed 8560 runs of the GMDH algorithm with BTE, NAF and 950-NIBNAF. The last expansion is with the maximal possible w such that the resulting output polynomial still has discernible integer and fractional parts. Correct evaluation of the prediction algorithm requires the plaintext modulus to be bigger than the maximal coefficient of the resulting polynomial. This lower bound for t can be deduced either from the maximal coefficient appearing after any run or, in case of known distribution of coefficient values, from the mean and the standard deviation. In both cases increasing window sizes reduce the bound as depicted in Figure 8. Since negative encoding coefficients are used, 950-NIBNAF demands a plaintext modulus of 7 bits which is almost 6 times smaller than for BTE and NAF.

As expected, w-NIBNAF encodings have longer expansions for bigger w's and that disrupts the decoding procedure in [4,14]. Namely, they naively split the resulting polynomial into two parts of equal size. As one can observe in Figure 8, using 950-NIBNAF, decoding in this manner will not give correct

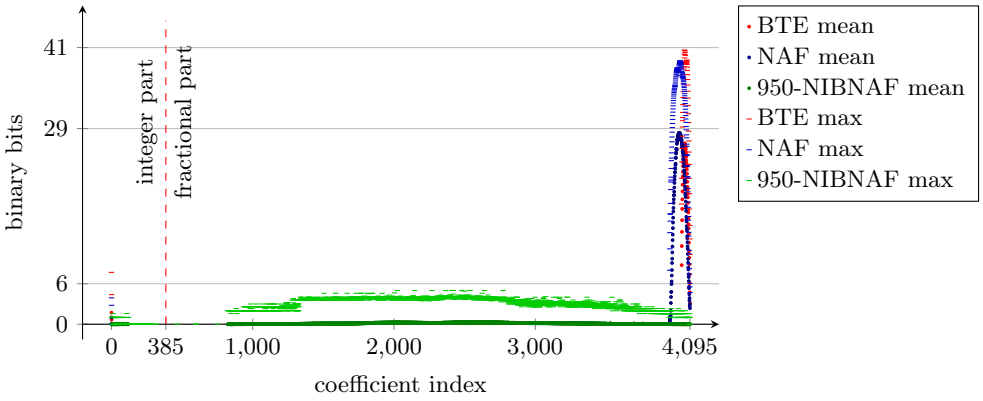


Fig. 8. The mean and the maximal size per coefficient of the resulting polynomial.

results. Instead, the splitting index i_s should be shifted towards zero, i.e. to 385. To be specific, i_s lies in the following interval implied by [4, Lemma 1]

$$(d_i + 1, d - d_f)$$

where $d_i = 2^{r+1}(\ell(w)_i^{\text{inp}} + \ell(w)_i^{\text{pol}}) - \ell(w)_i^{\text{pol}}$ and $d_f = 2^{r+1}(\ell(w)_f^{\text{inp}} + \ell(w)_f^{\text{pol}}) - \ell(w)_f^{\text{pol}}$. Indeed, this is the worst case estimation which results in the maximal $w = 74$ for the current network configuration.

One can notice that the impact of lower coefficients of the fractional part might be much smaller than the precision required by an application. In our use case the prediction value should be precise up to $e_f^{inp} = 1$. We denote the aggregated sum of lower coefficients multiplied by corresponding powers of the w-NIBNAF base as $L(j) = \sum_{i=j-1}^{i_s} a_i b_w^{-i}$. Then the omitted fractional coefficients a_i should satisfy

$$|L(i_c)| < 1,$$

where i_c is the index after which coefficients are ignored.

To find i_c we computed $L(j)$ for every index j of the fractional part and stored those sums for each run of the algorithm. For fixed j the distribution of $L(j)$ is bimodal with mean $\mu_{L(j)}$ and standard deviation $\sigma_{L(j)}$ (see Figure 9). Despite the fact that this unknown distribution is not normal, we naively approximate the prediction interval $[\mu_{L(j)} - 6\sigma_{L(j)}, \mu_{L(j)} + 6\sigma_{L(j)}]$ that will contain the future observation with high probability. It seems to be a plausible guess in this application because all observed $L(j)$ fall into that region with a big overestimate according to Figure 9. Therefore i_c is equal to the maximal j that satisfies $\tau(j) < 1$, where $\tau(j) = \max(|\mu_{L(j)} - 6\sigma_{L(j)}|, |\mu_{L(j)} + 6\sigma_{L(j)}|)$.

As Figure 10 shows, i_c is equal to 3388. Thus, the precision setting allows an overflow in any fractional coefficient a_j for $j < 3388$. The final goal is to provide the bound on t which is bigger than any a_j for $j \geq 3388$. Since the explicit distributions of coefficients are unknown and seem to vary among different

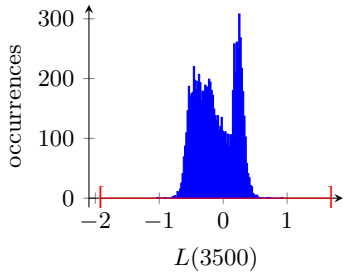


Fig. 9. The distribution of $L(3500)$ over 8560 runs of the GMDH algorithm and an approximation of its prediction interval in red.

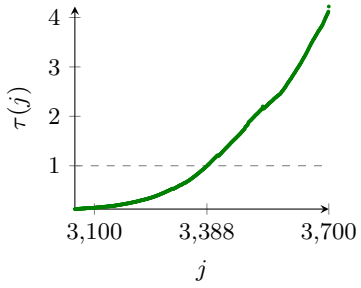


Fig. 10. The expected precision loss after ignoring fractional coefficients less than j.

	t	CRT factors	timing for one run
950-NIBNAF	$2^{5.044}$	1	2.57 s
BTE (this paper)	$2^{41.627}$	5	12.95 s
BTE [4]	$2^{103.787}$	13	32.5 s

Table 2. GMDH implementation with 950-NIBNAF and BTE [4]

indices, we rely in our analysis on the maximal coefficients occurring among all runs. Hence, the plaintext modulus should be bigger than $\max_{j \geq 3388}\{a_j\}$ over all resulting polynomials. Looking back at Figure 8, one can find t.

As mentioned in the beginning of Section 4, t is constrained in two ways: from the circuit and SHE correctness requirements. Now we bound the modulus according to SHE. In our setup, the FV scheme with 80 bits of security, the ring degree 4096 and the standard deviation of noise 102 requires $t \leq 396$ [4]. We compare our approach to the previous GMDH implementation in Table 2. As one can notice, 950-NIBNAF with the chopped fractional part does not need a CRT trick and requires a single modulus which reduces the timings in the sequential mode by 13 times. In the parallel mode it implies a 13 times smaller amount of memory is needed to hold the encrypted results.

Additionally, these plaintext moduli are much smaller than the worst case estimation from Section 3.1. For 950-NIBNAF we take $d \in [542, 821]$ according to the encoding degrees of input data and network coefficients. Any such encoding contains only one non-zero coefficient. Consequently, any product of those encodings has only one non-zero coefficient which is equal to 1. When all monomials of the GMDH polynomial result in an encoding with the same index of a non-zero coefficient, the maximal possible coefficient of the output encoding will occur. In this case the maximal coefficient is equal to the evaluation of the GMDH network with all input data and network coefficients being just 1. It leads to $t = 2 \cdot 6^{15} \simeq 2^{39.775}$.

5 Conclusions

We have presented a generic technique to encode fixed-point numbers using a non-integral base. This encoding technique is especially suitable for use when evaluating homomorphic functions since it utilizes the large degree of the defining polynomial imposed by the security requirements. This leads to a considerably smaller growth of the coefficients and allows one to reduce the size of the plaintext modulus significantly, resulting in faster implementations. We show that in the setting studied in [4], where somewhat homomorphic function evaluation is used to achieve a privacy-preserving electricity forecast algorithm, the plaintext modulus can be reduced from 2^{103} when using a balanced ternary expansion encoding, to $33 \simeq 2^{5.044}$ when using the encoding method introduced in this paper (non-integral base non-adjacent form with window size w), see Table 2. This smaller plaintext modulus means a factor 13 decrease in the running time of this privacy-preserving forecasting algorithm: closing the gap even further to making this approach suitable for industrial applications in the smart grid.

References

1. M. R. Albrecht. On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. Cryptology ePrint Archive, Report 2017/047, 2017. `http://eprint.iacr.org/2017/047`.
2. I. Aliev. Siegel's lemma and sum-distinct sets. *Discrete Comput. Geom.*, 39(1-3):59–66, 2008.
3. E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe. Post-quantum key exchange – a new hope. In *Proceedings of the 25th USENIX Security Symposium*. USENIX Association, 2016.
4. J. W. Bos, W. Castryck, I. Iliashenko, and F. Vercauteren. Privacy-friendly forecasting for the smart grid using homomorphic encryption and the group method of data handling (to appear). In M. Joye and A. Nitaj, editors, *Africacrypt 2017*, LNCS. Springer, 2017.
5. J. W. Bos, C. Costello, M. Naehrig, and D. Stebila. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In *2015 IEEE Symposium on Security and Privacy – SP*, pages 553–570. IEEE Computer Society, 2015.
6. J. W. Bos, K. Lauter, J. Loftus, and M. Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In M. Stam, editor, *Cryptography and Coding 2013*, volume 8308 of *Lecture Notes in Computer Science*, pages 45–64. Springer, 2013.
7. Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In S. Goldwasser, editor, *ITCS 2012*, pages 309–325. ACM, Jan. 2012.
8. Z. Brakerski and V. Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In P. Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 505–524. Springer, Heidelberg, Aug. 2011.
9. H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff. Privacy-preserving classification on deep neural network. Cryptology ePrint Archive, Report 2017/035, 2017. `http://eprint.iacr.org/2017/035`.

10. J. H. Cheon, A. Kim, M. Kim, and Y. Song. Homomorphic encryption for arithmetic of approximate numbers. Cryptology ePrint Archive, Report 2016/421, 2016. http://eprint.iacr.org/2016/421.
11. H. Cohen, A. Miyaji, and T. Ono. Efficient Elliptic Curve Exponentiation Using Mixed Coordinates. In K. Ohta and D. Pei, editors, *Advances in Cryptology – ASIACRYPT '98*, volume 1514 of *LNCS*, pages 51–65. Springer, 1998.
12. Commission for Energy Regulation. Electricity smart metering customer behaviour trials (CBT) findings report. Technical Report CER11080a, 2011. http://www.cer.ie/docs/000340/cer11080(a)(i).pdf.
13. A. Costache, N. P. Smart, and S. Vivek. Faster homomorphic evaluation of Discrete Fourier Transforms. *IACR Cryptology ePrint Archive*, 2016.
14. A. Costache, N. P. Smart, S. Vivek, and A. Waller. Fixed point arithmetic in SHE schemes. In *Selected Areas in Cryptography – SAC 2016*, LNCS. Springer, 2016.
15. CryptoExperts. FV-NFLlib. https://github.com/CryptoExperts/FV-NFLlib, 2016.
16. N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing. Manual for using homomorphic encryption for bioinformatics. Technical report, Technical report MSR-TR-2015-87, Microsoft Research, 2015.
17. N. Dowlin, R. Gilad-Bachrach, K. Laine, K. E. Lauter, M. Naehrig, and J. Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In M. Balcan and K. Q. Weinberger, editors, *International Conference on Machine Learning*, volume 48, pages 201–210. JMLR.org, 2016.
18. J. Fan and F. Vercauteren. Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2012:144, 2012.
19. C. Gentry. Fully homomorphic encryption using ideal lattices. In M. Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.
20. N. Göttert, T. Feller, M. Schneider, J. Buchmann, and S. A. Huss. On the design of hardware building blocks for modern lattice-based encryption schemes. In E. Prouff and P. Schaumont, editors, *CHES 2012*, volume 7428 of *LNCS*, pages 512–529. Springer, Heidelberg, Sept. 2012.
21. T. Güneysu, T. Oder, T. Pöppelmann, and P. Schwabe. Software speed records for lattice-based signatures. In P. Gaborit, editor, *PQCrypto 2013*, volume 7932 of *LNCS*, pages 67–82. Springer, 2013.
22. A. Ivakhnenko. Heuristic self-organization in problems of engineering cybernetics. *Automatica*, 6(2):207 – 219, 1970.
23. K. E. Lauter, A. López-Alt, and M. Naehrig. Private computation on encrypted genomic data. In D. F. Aranha and A. Menezes, editors, *LATINCRYPT 2014*, volume 8895 of *LNCS*, pages 3–27. Springer, Heidelberg, Sept. 2015.
24. A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *MATH. ANN*, 261:515–534, 1982.
25. V. Lyubashevsky, D. Micciancio, C. Peikert, and A. Rosen. SWIFFT: A modest proposal for FFT hashing. In K. Nyberg, editor, *FSE 2008*, volume 5086 of *LNCS*, pages 54–72. Springer, Heidelberg, Feb. 2008.
26. M. Naehrig, K. E. Lauter, and V. Vaikuntanathan. Can homomorphic encryption be practical? In C. Cachin and T. Ristenpart, editors, *ACM Cloud Computing Security Workshop – CCSW*, pages 113–124. ACM, 2011.
27. T. Pöppelmann and T. Güneysu. Towards practical lattice-based public-key encryption on reconfigurable hardware. In T. Lange, K. Lauter, and P. Lisonek, editors, *SAC 2013*, volume 8282 of *LNCS*, pages 68–85. Springer, Heidelberg, Aug. 2014.

28. G. W. Reitwiesner. *Binary Arithmetic*, volume 1 of *Advances in Computers*, pages 231–308. Academic Press, 1960.
29. N. P. Smart and F. Vercauteren. Fully homomorphic SIMD operations. *Designs, Codes, and Cryptography*, 71(1):57–81, Apr. 2014.
30. D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In K. G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 27–47. Springer, Heidelberg, May 2011.

A Proofs

Lemma 1 *For an integer $w \geq 1$ the polynomial $F_w(x) = x^{w+1} - x^w - x - 1$ has a unique positive root $b_w > 1$. The sequence $b_1, b_2, \ldots$ is strictly decreasing and the limit as w tends to infinity of b_w is 1. Further, $(x^2+1) \mid F_w(x)$ for $w \equiv 3 \mod 4$.*

Proof. We have for $w \geq 1$ that

$$F'_w(x) = (w+1)x^w - wx^{w-1} - 1 = (x-1)((w+1)x^{w-1} + x^{w-2} + \cdots + 1)$$

so that for $x \geq 0$ there is only one turning point of $F_w(x)$, at $x = 1$. Further, $F''_w(x) = (w+1)wx^{w-1} - w(w-1)x^{w-2}$, which takes the value $2w > 0$ at $x = 1$, so the turning point is a minimum. Since $F_w(0) = -1$ and $\lim_{x\to\infty} F_w(x) = \infty$ we conclude that there is a unique positive root of $F_w(x)$, $b_w > 1$, for any $w \geq 1$. Further, we have that $F_{w+1}(x) = xF_w(x) + x^2 - 1$ so that $F_{w+1}(b_w) = b_w^2 - 1 > 0$ so that $b_{w+1} < b_w$ and hence the sequence b_w is strictly decreasing and bounded below by 1 so must converge to some limit, say $b_\infty \geq 1$. If $b_\infty > 1$ then as b_w is the positive solution to $x - 1 = (x+1)/x^w$ and, for $x \geq b_\infty > 1$, $\lim_{w\to\infty}(x+1)/x^w = 0$ we see that $b_\infty = \lim_{w\to\infty} b_w = 1$, a contradiction. Hence $b_\infty = 1$ as required. Finally we see that $F_w(x) = x(x-1)(x^{w-1}+1) - (x^2+1)$ and for $w = 4k+3$ that $x^{w-1} + 1 = 1 - (-x^2)^{2k+1} = (x^2+1)\sum_{i=0}^{2k}(-x^2)^i$ and hence $(x^2+1) \mid F_{4k+3}(x)$. □

Recall that to find a lower bound on the maximal absolute coefficient size we consider w-balanced ternary sequences and to each sequence (a_i) we have the corresponding polynomial $\sum_i a_i X^i$ in R_t. As we only look at the coefficients and their relative distances we can simply assume that to each w-balanced ternary sequence $c_0, c_1, \ldots, c_d$ of length $d+1$ we have the associated polynomial $c_0 + c_1 X + \ldots + c_d X^d$ of degree d. Multiplication of polynomials thus gives us a way of multiplying (finite) w-balanced ternary sequences. In the rest of this appendix we use the polynomial and sequence notation interchangeably according to whichever is more convenient.

Lemma 3 *A lower bound on the maximal absolute size of a term that can be produced by taking the product of p arbitrary w-balanced ternary sequences of length $d+1$ is*

$$B_w(d,p) := \sum_{k=0}^{\lfloor \lfloor p\lfloor d/w\rfloor/2\rfloor/(\lfloor d/w\rfloor+1)\rfloor} (-1)^k \binom{p}{k} \binom{p-1+\lfloor p\lfloor d/w\rfloor/2\rfloor - k\lfloor d/w\rfloor - k}{p-1} .$$

Proof. Consider the product of p sequences all of which are equal to $m = 10\cdots010\cdots010\cdots0$ of length $d+1$, having $n := \lfloor d/w \rfloor + 1$ non-zero terms (all being 1) and between each pair of adjacent non-zero terms there are exactly $w-1$ zero terms. Note that n is the maximal number of non-zero terms possible. As polynomials we have that

$$m = \sum_{i=0}^{n-1} X^{iw} = \frac{1-X^{nw}}{1-X^w} \ ,$$

and hence we have

$$\begin{aligned} m^p &= \left(\frac{1-X^{nw}}{1-X^w}\right)^p = (1-X^{nw})^p \cdot (1-X^w)^{-p} \\ &= \left(\sum_{i=0}^{p} (-1)^i \binom{p}{i} X^{inw}\right) \left(\sum_{j=0}^{\infty} \binom{p-1+j}{p-1} X^{jw}\right) \\ &= \sum_{j=0}^{\infty} \sum_{i=0}^{p} (-1)^i \binom{p}{i} \binom{p-1+j}{p-1} X^{(in+j)w} \\ &= \sum_{\ell=0}^{\infty} \left(\sum_{k=0}^{\lfloor \ell/n \rfloor} (-1)^k \binom{p}{k} \binom{p-1+\ell-kn}{p-1} \right) X^{\ell w} \ , \end{aligned}$$

where we have used the substitution $(i,j) \to (k,\ell) = (i, in+j)$. Since we know that m^p has degree $p(n-1)w$ we can in fact change the infinite sum over ℓ to a finite one from $\ell = 0$ to $p(n-1)$. To give the tightest lower bound we look for the maximal coefficient of m^p. We prove in Lemma 5 below that $\ell = \lfloor p(n-1)/2 \rfloor$ does the job and this coefficient is exactly $B_w(d,p)$. □

Lemma 4 *Suppose w divides d, then $B_w(d,p)$ equals the maximal absolute size of a term that can be produced by taking the product of p arbitrary w-balanced ternary sequences of length $d+1$.*

Proof. Let $S_w(d,p)$ be the set of all sequences that are the product of p arbitrary w-balanced ternary sequences of length $d+1$. To prove the lemma we bound all the terms of any sequence in $S_w(d,p)$. For $i = 0, \ldots, pd$ define

$$m_w(d,p,i) = \max\{\, |a_i| \ \mid \ a_i \text{ is the } i\text{'th term of a sequence in } S_w(d,p) \,\} \ .$$

We will prove by induction on p that $m_w(d,p,i) \le B_w(d,p,\lfloor i/w \rfloor)$ where

$$B_w(d,p,\ell) = \sum_{k=0}^{\lfloor \ell/n \rfloor} (-1)^k \binom{p}{k} \binom{p-1+\ell-kn}{p-1}$$

is the coefficient of $X^{\ell w}$ in m^p. We will use the notation $C_i(f)$ for a polynomial f to denote the coefficient of X^i in $f(X)$; this is defined to be zero if $i > \deg(f)$ or $i < 0$. Thus in this notation $B_w(d,p,\ell) = C_{\ell w}\left((1-X^{nw})^p/(1-X^w)^p\right)$.

The base case $p = 1$ is straight forward, all the $m_w(d, p, i)$ are equal to 1 by the definition of a w-balanced ternary sequence. We therefore suppose that $m_w(d, p-1, i) \leq B_w(d, p-1, \lfloor i/w \rfloor)$ for $0 \leq i \leq (p-1)d$.

Consider a product of p w-balanced ternary sequences of length $d+1$. It can be written as $f(X)e(X)$ where $f(X) \in S_w(d, p-1)$ and $e(X) \in S_w(d, 1)$. We know that if $f(X) = \sum_{i=0}^{(p-1)d} a_i X^i$ then $|a_i| \leq m_w(d, p-1, i)$ and if $e(X) = \sum_{j=0}^{d} \alpha_j X^j$ that

$$f(X)e(X) = \sum_{k=0}^{pd} \left(\sum_{i=\max(0,k-d)}^{\min((p-1)d,k)} a_i \alpha_{k-i} \right) X^k \ ,$$

and because of the form of $e(X)$ we see that

$$|C_k(fe)| \leq \sum_{j=1}^{n_k} |a_{i_j}| \leq \sum_{j=1}^{n_k} m_w(d, p-1, i_j)$$

for some $n_k \leq n$, $\max(0, k-d) \leq i_1 < i_2 < \cdots < i_{n_k} \leq \min((p-1)d, k)$ and $i_{j+1} - i_j \geq w$ for $j = 1, \ldots, n_k - 1$.

The final condition on the i_j implies that the $\lfloor i_j/w \rfloor$ are distinct and since $m_w(d, p-1, i)$ is bounded above by $B_w(d, p-1, \lfloor i/w \rfloor)$, which depends only on $\lfloor i/w \rfloor$, we can recast this as

$$|C_k(fe)| \leq \sum_{j=1}^{n_k} B_w(d, p-1, \ell_j) = \sum_{j=1}^{n_k} C_{\ell_j w} \left(\left(\frac{1 - X^{nw}}{1 - X^w} \right)^{p-1} \right)$$

where $\max(0, \lfloor k/w \rfloor - (n-1)) \leq \ell_1 < \ell_2 < \cdots < \ell_{n_k} \leq \min((p-1)(n-1), \lfloor k/w \rfloor)$ where we have used that $d/w = n - 1$ is an integer.

Since $\lfloor k/w \rfloor - (\lfloor k/w \rfloor - (n-1)) + 1 = n$ we see that to make n_k as large as possible the ℓ_j must be the (at most n) consecutive integers in this range subject also to $0 \leq \ell_1$ and $\ell_{n_k} \leq (p-1)(n-1)$. Thus taking a maximum over all possible

f and e we have

$$\begin{aligned}
m_w(d,p,k) &\leq \sum_{\ell=\lfloor k/w\rfloor-(n-1)}^{\lfloor k/w\rfloor} C_{\ell w}\left(\left(\frac{1-X^{nw}}{1-X^w}\right)^{p-1}\right)\\
&= \sum_{j=0}^{n-1} C_{\lfloor k/w\rfloor w}\left(\left(\frac{1-X^{nw}}{1-X^w}\right)^{p-1} X^{w(n-1-j)}\right)\\
&= C_{\lfloor k/w\rfloor w}\left(\left(\frac{1-X^{nw}}{1-X^w}\right)^{p-1}\sum_{j=0}^{n-1} X^{w(n-1-j)}\right)\\
&= C_{\lfloor k/w\rfloor w}\left(\left(\frac{1-X^{nw}}{1-X^w}\right)^{p-1}\sum_{i=0}^{n-1} X^{wi}\right)\\
&= C_{\lfloor k/w\rfloor w}\left(\left(\frac{1-X^{nw}}{1-X^w}\right)^{p-1}\left(\frac{1-X^{nw}}{1-X^w}\right)\right)\\
&= C_{\lfloor k/w\rfloor w}\left(\left(\frac{1-X^{nw}}{1-X^w}\right)^{p}\right) = B_w(d,p,\lfloor k/w\rfloor)\ ,
\end{aligned}$$

which proves the inductive step. To finish the proof we note as before that the maximal value of $B_w(d,p,\lfloor k/w\rfloor)$ for $0\leq k\leq pd$ is reached, for example, when $\lfloor k/w\rfloor = \lfloor p\lfloor d/w\rfloor/2\rfloor$ and in this case we have $B_w(d,p)$ as required. □

Lemma 5. *The maximum coefficient of m^p occurs as the coefficient of X^ℓ for $\ell=\lfloor p(n-1)/2\rfloor$ (although it may also appear as other coefficients).*

Proof. To prove this we prove the following, let a_ℓ be the coefficient of $X^{\ell w}$ then we claim that

$$a_\ell = a_{p(n-1)-\ell} \qquad \text{and} \qquad 1=a_0\leq a_1\leq\cdots\leq a_{\lfloor p(n-1)/2\rfloor}. \tag{2}$$

From these two claims it followed that $a_{\lfloor p(n-1)/2\rfloor}\geq a_{\lfloor p(n-1)/2\rfloor+1}\geq\cdots\geq a_{p(n-1)}$ which is what we require.

We will prove (2) by induction on p. The base case $p=1$ is true as all $a_\ell=1$. Thus, suppose that $p\geq 2$ and that the coefficients (in powers of X^w) of m^{p-1} are c_i for $i=0,\ldots,(p-1)(n-1)$ and satisfy the inductive hypothesis $c_i=c_{(p-1)(n-1)-i}$ and $1=c_0\leq c_1\leq\cdots\leq c_{\lfloor(p-1)(n-1)/2\rfloor}$. We have

$$\begin{aligned}
\sum_{\ell=0}^{p(n-1)} a_\ell X^{\ell w} = m^p = m^{p-1}m &= \left(\sum_{i=0}^{(p-1)(n-1)} c_i X^{iw}\right)\left(\sum_{j=0}^{n-1} X^{jw}\right)\\
&= \sum_{\ell=0}^{p(n-1)}\left(\sum_{k=\max(0,\ell-n+1)}^{\min((p-1)(n-1),\ell)} c_k\right)X^{\ell w}\ ,
\end{aligned}$$

so that

$$a_{p(n-1)-\ell} = \sum_{k=\max(0,(p-1)(n-1)-\ell)}^{\min((p-1)(n-1),p(n-1)-\ell)} c_k = \sum_{k=\max(-(p-1)(n-1),-\ell)}^{\min(0,n-1-\ell)} c_{(p-1)(n-1)+k}$$
$$= \sum_{i=\max(0,\ell-n+1)}^{\min((p-1)(n-1),\ell)} c_{(p-1)(n-1)-i} = \sum_{i=\max(0,\ell-n+1)}^{\min((p-1)(n-1),\ell)} c_i = a_\ell \ ,$$

which is the first part of the inductive step. Further,

$$a_\ell - a_{\ell-1} = \left(\sum_{k=\max(0,\ell-n+1)}^{\min((p-1)(n-1),\ell)} c_k\right) - \left(\sum_{k=\max(0,\ell-n)}^{\max((p-1)(n-1),\ell-1)} c_k\right)$$
$$= \begin{cases} c_\ell & \text{if } 1 \le \ell < n \\ c_\ell - c_{\ell-n} & \text{if } n \le \ell \le (p-1)(n-1)+1 \\ -c_{\ell-n} & \text{if } (p-1)(n-1)+1 < \ell \le p(n-1) \end{cases} .$$

We are interested in the case when $\ell \le p(n-1)/2$ which implies that $\ell - n \le (p-2)(n-1)/2 - 1 < (p-1)(n-1)/2$. If $\ell < n$ then the difference is certainly positive as $c_\ell \ge 1$ so suppose $\ell \ge n$ so that if the difference is negative, $c_\ell - c_{\ell-n} < 0$, then $c_\ell < c_{\ell-n}$. This implies, by the inductive hypothesis, that $\ell > (p-1)(n-1)/2$ in which case $c_{\ell-n} > c_\ell = c_{(p-1)(n-1)-\ell}$ but this implies that $(p-1)(n-1) - \ell < \ell - n$, as both are smaller than $(p-1)(n-1)/2$, which rearranges to give $p(n-1)+1 < 2\ell$. This is a contradiction as we are assuming that $\ell \le p(n-1)/2$. Clearly $a_0 = c_0 = 1$ so this proves that

$$1 = a_0 \le a_1 \le \cdots \le a_{\lfloor p(n-1)/2 \rfloor}$$

as required, completing the inductive step. □

B How we approximated $B_w(d,p)$

In this appendix we describe how to approximate $B_w(d,p)$ and give details and sketch proofs of the results stated in section 3.2.

As previously mentioned, the easiest way to bound $B_w(d,p)$ is by $B_w(d,p) \le (\lfloor d/w \rfloor + 1)^{p-1} = n^{p-1}$ but we can be more precise than this. For p even and fixed, we have $\lfloor p(n-1)/2 \rfloor = p(n-1)/2$ and so

$$\binom{p-1+\left\lfloor \frac{p(n-1)}{2} \right\rfloor - kn}{p-1} = \binom{(\frac{p}{2}-k)n + \frac{p}{2} - 1}{p-1} = \frac{(\frac{p}{2}-k)^{p-1}}{(p-1)!} n^{p-1} + O(n^{p-3})$$

which is a polynomial in n of degree $p-1$ and whose coefficients depend on p and k. The coefficient of n^{p-2} is zero since $\sum_{i=0}^{p-2} (\frac{p}{2} - 1 - i) = 0$, in fact the

coefficient of any even power of n is zero since we have

$$\binom{(\frac{p}{2}-k)n+\frac{p}{2}-1}{p-1} = \frac{1}{(p-1)!}\prod_{j=1}^{p-1}\left((\tfrac{p}{2}-k)n+\tfrac{p}{2}-j\right)$$

$$= \frac{(\frac{p}{2}-k)n}{(p-1)!}\prod_{1\le j<p/2}\left((\tfrac{p}{2}-k)^2n^2-(\tfrac{p}{2}-j)^2\right) = \frac{(\frac{p}{2}-k)n}{(p-1)!}\prod_{1\le j<p/2}\left((\tfrac{p}{2}-k)^2n^2-j^2\right).$$

For $n \ge \frac{p}{2}$ we have that $\lfloor (p(n-1)/2)/n \rfloor = \lfloor \frac{p}{2} - \frac{p}{2n} \rfloor = \frac{p}{2}-1$ which does not depend on n. In this case multiplying by $(-1)^k\binom{p}{k}$ and summing from $k=0$ to $k=\frac{p}{2}-1$ gives a polynomial in n of degree $p-1$ and whose leading coefficient is

$$\ell_p = p\sum_{0\le k<\frac{p}{2}}(-1)^k\frac{(\frac{p}{2}-k)^{p-1}}{k!(p-k)!}\ , \tag{3}$$

the reason for the strange way of writing the upper limit on k will become clear in a moment. Although we do not prove it here it turns out that this polynomial also correctly gives the value of $B_w(d,p)$ for $1 \le n < \frac{p}{2}$ too.

Now suppose that $p>1$ is odd. This time the value of $\lfloor p(n-1)/2 \rfloor$ depends on the parity of n. For n odd it is again $p(n-1)/2$ while for n even it is $(p(n-1)-1)/2$. Thus for n odd the expansion of the binomial coefficient as a polynomial in n follows almost as above except now all odd powers of n disappear:

$$\binom{(\frac{p}{2}-k)n+\frac{p}{2}-1}{p-1} = \frac{1}{(p-1)!}\prod_{1\le j<p/2}\left((\tfrac{p}{2}-k)^2n^2-(j-\tfrac{1}{2})^2\right),$$

while for n even it is

$$\binom{p-1+\left\lfloor\frac{p(n-1)}{2}\right\rfloor-kn}{p-1} = \binom{(\frac{p}{2}-k)n+\frac{p-3}{2}}{p-1} = \frac{(\frac{p}{2}-k)^{p-1}}{(p-1)!}n^{p-1}+O(n^{p-2})$$

and the coefficients of odd powers of n do not disappear. Indeed we have

$$\binom{(\frac{p}{2}-k)n+\frac{p-3}{2}}{p-1} = \frac{1}{(p-1)!}\left(1-\frac{p-1}{(p-2k)n}\right)\prod_{1\le j<p/2}\left((\tfrac{p}{2}-k)^2n^2-(j-1)^2\right). \tag{4}$$

This time we have, for $n \ge p$, that $\lfloor \lfloor p(n-1)/2 \rfloor / n \rfloor = \frac{p-1}{2}$ which is independent of n and we find that again we can write $B_w(d,p)$ as a 'polynomial' in n of degree $p-1$ and whose coefficients depend on p *and* the parity of n. Of course this is not really a polynomial in n since the coefficients depend on the parity of n however we can see that the leading coefficient does not depend on n and is also given by (3). It turns out (although we don't prove it) that this 'polynomial' in fact has no odd powers of n and again gives the correct value of $B_w(d,p)$ even when $1 \le n < p$. That is we have for odd p and even n:

$$B_w(d,p) = \sum_{0\le k<p/2}(-1)^k\binom{p}{k}\frac{1}{(p-1)!}\prod_{1\le j<p/2}\left((\tfrac{p}{2}-k)^2n^2-(j-1)^2\right).$$

Another way of writing this, for odd p, is that $B_w(d,p) = f(n) - (-1)^n g(n)$ for two even polynomials f and g of degree $p-1$ and $p-3$ respectively. This approach is used in section 3.2. It is not hard to see that the constant terms of f and g are equal since $\binom{(p/2-k)n+(p-3)/2}{p-1}$ has zero constant term as a polynomial in n; in fact they take the value $B_w(w,p-2)^2/2^{2p-3} = \binom{p-1}{(p-1)/2}^2/2^{2p-1}$.

What is remarkable about the coefficients of these polynomials is that they can we written as integrals. This fact for the leading coefficient ℓ_p ($r=0$ below) is proven in [2] and the references therein. We also present a simple generalization here for the other coefficients.

Lemma 6. *For p even or n odd,*

$$B_w(d,p) = \sum_{0\le r<p/2} \left(\frac{a_r(p)}{\pi} \int_{-\infty}^{\infty} \frac{\sin^p\theta}{\theta^{p-2r}} \mathrm{d}\theta \right) n^{p-1-2r},$$

and for $p>1$ odd and n even

$$B_w(d,p) = \sum_{0\le r<p/2} \left(\frac{a_r(p-1)(p-2r-1)}{(p-1)\pi} \int_{-\infty}^{\infty} \frac{\sin^p\theta}{\theta^{p-2r}} \mathrm{d}\theta \right) n^{p-1-2r},$$

where $a_r(x)$ is a polynomial of degree r whose coefficients are rational and with leading coefficient $(6^r r!)^{-1}$. More precisely we have $a_0(x)=1$ and for any integer x we have

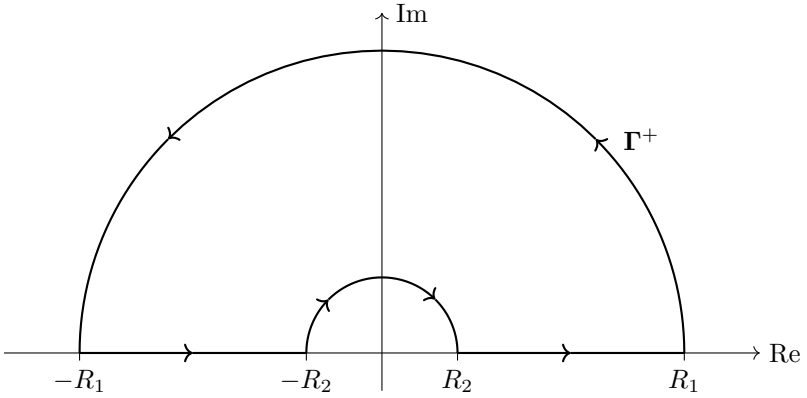
$$a_r(2x) := \frac{\displaystyle\sum_{0<i_1<i_2<\cdots<i_r<x} (i_1 i_2 \cdots i_r)^2}{\displaystyle\prod_{1\le j\le 2r} (x-\tfrac{j}{2})} \qquad \text{for } r\ge 1.$$

Note that the above also encompasses the $p=1$ case if one defines $\frac{p-1}{p-1}=1$.

Proof. To start we use the well known formula for sine:

$$\begin{aligned}\sin^p\theta &= \left(\frac{e^{i\theta}-e^{-i\theta}}{2i}\right)^p = \frac{1}{(2i)^p}\sum_{k=0}^{p}\binom{p}{k} e^{ik\theta}(-e^{-i\theta})^{p-k} \\ &= \left(\frac{i}{2}\right)^p \sum_{k=0}^{p}(-1)^k\binom{p}{k} e^{(2k-p)i\theta}.\end{aligned}$$

We are thus interested in the integral $\int_{-\infty}^{\infty} e^{(2k-p)i\theta}/\theta^q$, if it exists. We will see that the integral exists, for $q\in\mathbb{Z}_{>0}$, as a Cauchy principal value, namely the limit of integrating from $-R_1$ to $-R_2$ plus from R_2 to R_1 as $R_1\to\infty$ and $R_2\to 0$. We start with the case $q=1$, and have three cases, either $2k-p$ is negative, zero or positive. In the case when $2k=p$ then the function is just θ^{-1} which is an odd function. Therefore $\int_{-R_1}^{-R_2}\theta^{-1}\mathrm{d}\theta + \int_{R_2}^{R_1}\theta^{-1}\mathrm{d}\theta = 0$ and so in the limit it is also 0. For the other two cases we use complex analysis, if $2k-p$ is positive then we consider integrating along the curve $\mathbf{\Gamma}^+$ shown below:



Since $e^{(2k-p)iz}/z$ has no poles inside $\mathbf{\Gamma}^+$ we have that $\int_{\mathbf{\Gamma}^+} e^{(2k-p)iz}/z\,\mathrm{d}z = 0$ by Cauchy's Integral Theorem, hence we have

$$\int_{-R_1}^{-R_2} \frac{e^{(2k-p)iz}}{z}\mathrm{d}z + \int_{R_2}^{R_1} \frac{e^{(2k-p)iz}}{z}\mathrm{d}z + \int_0^{\pi} ie^{(2k-p)iR_1e^{it}}\mathrm{d}t - \int_0^{\pi} ie^{(2k-p)iR_2e^{it}}\mathrm{d}t = 0$$

where we have used the substitution $z = Re^{it}$ in the last two integrals. We note that for $0 < t < \pi$ we have $|ie^{(2k-p)iR_1e^{it}}| = e^{(p-2k)R_1\sin t} \to 0$ as $R_1 \to \infty$ since $\sin t$ is positive and $p-2k$ is negative. Thus by Jordan's lemma the third integral tends to 0 as R_1 tends to infinity. For the last integral we note that, as R_2 tends to 0, it tends to $(\pi-0)i\mathrm{Res}(e^{(2k-p)iz}/z, z=0) = \pi i \lim_{z\to 0} e^{(2k-p)iz} = \pi i$. Hence we have, on taking the limit as $R_1 \to \infty$ and $R_2 \to 0$ and rearranging, that

$$\int_{-\infty}^{\infty} \frac{e^{(2k-p)iz}}{z}\mathrm{d}z = \pi i \qquad \text{for } 2k-p > 0.$$

In a similar manner, when $2k-p$ is negative we can integrate around the contour $\mathbf{\Gamma}^-$ which is the same as $\mathbf{\Gamma}^+$ except that the arcs are now in the lower half of the complex plane instead of the upper half, we thus have to change the direction to be clockwise on the outer arc and anti-clockwise on the inner arc. This ensures that as R_1 tends to infinity the integral on the outer arc tends to zero and as R_2 tends to zero the integral on the inner arc tends to πi as before. We can therefore deduce that

$$\int_{-\infty}^{\infty} \frac{e^{(2k-p)iz}}{z}\mathrm{d}z = -\pi i \qquad \text{for } 2k-p < 0.$$

Now assuming the integrals exist we use integration by parts with $u = e^{(2k-p)i\theta}$ and $\mathrm{d}v = \theta^{-q}$ then $\mathrm{d}u = i(2k-p)e^{(2k-p)i\theta}$ and $v = -\theta^{-(q-1)}/(q-1)$ and on noting that $\lim_{R\to\infty}[uv]_{-R}^{R} = 0$ we have

$$\int_{-\infty}^{\infty} \frac{e^{(2k-p)i\theta}}{\theta^q}\mathrm{d}\theta = \frac{i(2k-p)}{q-1}\int_{-\infty}^{\infty} \frac{e^{(2k-p)i\theta}}{\theta^{q-1}}\mathrm{d}\theta,$$

and a simply induction argument shows that for $q \in \mathbb{Z}_{>0}$ the integrals exist as Cauchy principal values and are equal to

$$\int_{-\infty}^{\infty} \frac{e^{(2k-p)i\theta}}{\theta^q} \mathrm{d}\theta = \begin{cases} -\pi i \frac{(i(2k-p))^{q-1}}{(q-1)!} & \text{for } 2k-p < 0 \\ 0 & \text{for } 2k-p = 0 \\ \pi i \frac{(i(2k-p))^{q-1}}{(q-1)!} & \text{for } 2k-p > 0. \end{cases}$$

Therefore

$$\frac{1}{\pi}\int_{-\infty}^{\infty} \frac{\sin^p \theta}{\theta^{p-2r}} \mathrm{d}\theta = \frac{1}{\pi}\left(\frac{i}{2}\right)^p \sum_{k=0}^{p} (-1)^k \binom{p}{k} \int_{-\infty}^{\infty} \frac{e^{(2k-p)i\theta}}{\theta^{p-2r}} \mathrm{d}\theta$$

$$= \left(\frac{-1}{2}\right)^p \left(- \sum_{0 \le k < p/2} (-1)^k \binom{p}{k} \frac{(2k-p)^{p-2r-1}}{i^{2r}(p-2r-1)!} + \sum_{p/2 < k \le p} (-1)^k \binom{p}{k} \frac{(2k-p)^{p-2r-1}}{i^{2r}(p-2r-1)!} \right)$$

$$= \sum_{0 \le k < p/2} \frac{(-1)^{r+k}}{2^{2r+1}} \binom{p}{k} \frac{(\frac{p}{2}-k)^{p-2r-1}}{(p-2r-1)!} + \sum_{0 \le k < p/2} \frac{(-1)^{r+2p-k}}{2^{2r+1}} \binom{p}{p-k} \frac{(\frac{p}{2}-k)^{p-2r-1}}{(p-2r-1)!}$$

$$= \sum_{0 \le k < p/2} \frac{(-1)^{r+k}}{2^{2r}} \binom{p}{k} \frac{(\frac{p}{2}-k)^{p-2r-1}}{(p-2r-1)!}.$$

On the other hand we have for even p that

$$B_w(d,p) = \sum_{0 \le k < p/2} (-1)^k \binom{p}{k} \binom{(\frac{p}{2}-k)n + \frac{p}{2} - 1}{p-1}$$

$$= \sum_{0 \le k < p/2} (-1)^k \binom{p}{k} \frac{(\frac{p}{2}-k)n}{(p-1)!} \prod_{1 \le j < p/2} \left((\tfrac{p}{2}-k)^2 n^2 - j^2\right)$$

$$= \sum_{0 \le k < p/2} (-1)^k \binom{p}{k} \sum_{0 \le r < p/2} \left(\frac{(\frac{p}{2}-k)^{p-1-2r}}{(p-1)!} (-1)^r \sum_{0 < i_1 < \cdots < i_r < p/2} (i_1 i_2 \cdots i_r)^2 \right) n^{p-1-2r}$$

$$= \sum_{0 \le r < p/2} \left(a_r(p) \sum_{0 \le k < p/2} \frac{(-1)^{r+k}}{2^{2r}} \binom{p}{k} \frac{(\frac{p}{2}-k)^{p-2r-1}}{(p-2r-1)!} \right) n^{p-1-2r}$$

$$= \sum_{0 \le r < p/2} \left(\frac{a_r(p)}{\pi} \int_{-\infty}^{\infty} \frac{\sin^p \theta}{\theta^{p-2r}} \mathrm{d}\theta \right) n^{p-1-2r}$$

as stated.

To show the stated properties of $a_r(x)$ we note that

$$b_r(x) := \sum_{0 < i_1 < \cdots < i_r < x} (i_1 i_2 \cdots i_r)^2 = \sum_{0 < i_r < x} \left(\sum_{0 < i_1 < i_2 < \cdots < i_{r-1} < i_r} (i_1 \cdots i_{r-1})^2 \right) i_r^2$$

$$= \sum_{0 < i_r < x} b_{r-1}(i_r) i_r^2$$

so we can use an inductive argument on r. For $r = 1$ and integer x we have $b_1(x) = \frac{1}{6}(x-1)x(2x-1)$ where we have used Faulhaber's formula for the sum of squares. Note that the leading coefficient is $\frac{2}{6} = (3^1 1!)^{-1}$. More generally Faulhaber's formula states that $\sum_{k=1}^{\ell} k^q = \frac{1}{q+1}\sum_{j=0}^{q}(-1)^j\binom{q+1}{j}B_j\ell^{q+1-j}$ where B_j are the Bernoulli numbers with $B_0 = 1$, $B_1 = -\frac{1}{2}, \ldots$. As an inductive hypothesis we have that for integer x, $b_r(x)$ is a polynomial in x of degree $3r$ and with leading coefficient $(3^r r!)^{-1}$. Say

$$b_r(x) = \sum_{j=0}^{3r} \beta_{r,j} x^j$$

where $\beta_{r,3r} = (3^r r!)^{-1}$. Then

$$\begin{aligned} b_{r+1}(x) &= \sum_{i_{r+1}=1}^{x-1} b_r(i_{r+1}) i_{r+1}^2 = \sum_{i_{r+1}=1}^{x-1}\sum_{j=0}^{3r} \beta_{r,j} i_{r+1}^{j+2} = \sum_{j=0}^{3r}\beta_{r,j}\sum_{i_{r+1}=1}^{x-1} i_{r+1}^{j+2} \\ &= \sum_{j=0}^{3r}\beta_{r,j}\frac{1}{j+3}\sum_{k=0}^{j+2}(-1)^k\binom{j+3}{k}B_k(x-1)^{j+3-k} \end{aligned}$$

which is again a polynomial in x of degree $3r+3-0 = 3(r+1)$ and leading coefficient $\beta_{r,3r}\frac{1}{3r+3} = \frac{1}{3^{r+1}(r+1)!}$. Hence we have proved the inductive step and the result holds. In particular for even x, $b_r(x/2)$ has leading coefficient $\beta_{r,3r}/2^{3r} = (2^{2r}6^r r!)^{-1}$.

Further we note that $b_r(x)$ is zero at $x = 0, 1, 2, \ldots, r$ since in the sum we must have $i_j \geq j$ for $j = 1, 2, \ldots, r$ so if $r \leq i_r < x \leq r$ there is no possible value that i_r can take and the sum is empty. To show that $a_r(x)$ is a polynomial we need to also prove that $b_r(x)$ is zero at each half integers between 0 and r, we do not show this as it is not needed in our later approximations. Finally we note that the leading coefficient of $a_r(x)$ is $2^{2r}/(2^{2r}\cdot 6^r r!) = (6^r r!)^{-1}$ as required.

Now when p and n are both odd we have almost exactly the same proof only with a very similar expression in place of $b_r(x)$. Namely we have for an integer x the sum

$$\sum_{0<i_1<i_2<\cdots<i_r<x} \left((i_1-\tfrac{1}{2})(i_2-\tfrac{1}{2})\cdots(i_r-\tfrac{1}{2})\right)^2$$

which can be shown to equal $b_r(x+\frac{1}{2})$ although we do not show it here. Since we are interested in the integer $x = \frac{p+1}{2}$ we note that this gives $b_r(p/2)$ which is the same as in the case when p is even. Hence we again have the same form for $B_w(d,p)$.

Finally when p is odd and n is even we use (4) and the proof follows in almost the same manner as before however this time we note that

$$\sum_{0<i_1<\cdots<i_r<x} ((i_1-1)(i_2-1)\cdots(i_r-1))^2 = b_r(x-1)$$

so that

$$\frac{\displaystyle\sum_{0<i_1<i_2<\cdots<i_r<p/2} ((i_1-1)(i_2-1)\cdots(i_r-1))^2}{\displaystyle\prod_{1\le j\le 2r}(\tfrac{p}{2}-\tfrac{j}{2})} = \frac{b((p-1)/2)}{\displaystyle\prod_{1\le j\le 2r}(\tfrac{p}{2}-\tfrac{j}{2})}$$
$$= a_r(p-1)\frac{\prod_{1\le j\le 2r}(\frac{p-1}{2}-\frac{j}{2})}{\prod_{1\le j\le 2r}(\frac{p}{2}-\frac{j}{2})} = a_r(p-1)\frac{p-1-2r}{p-1}.$$

This is still a polynomial in p since $x \mid a_r(x)$ for all $r \ge 1$ and when $r = 0$ this simplifies to $a_1(p-1) = 1$. □

The first few expressions for the $a_r(x)$ are:

$$a_1(x) = \frac{x}{6}; \qquad a_2(x) = \frac{x}{72}\left(x+\frac{2}{5}\right); \qquad a_3(x) = \frac{x}{1296}\left(x^3+\frac{6}{5}x^2+\frac{16}{35}x\right);$$
$$a_4(x) = \frac{x}{31104}\left(x^3+\frac{12}{5}x^2+\frac{404}{175}x+\frac{144}{175}\right);$$
$$a_5(x) = \frac{x}{933120}\left(x^4+4x^3+\frac{244}{35}x^2+\frac{208}{35}x+\frac{768}{385}\right);$$
$$a_6(x) = \frac{x}{33592320}\left(x^5+6x^4+\frac{572}{35}x^3+\frac{4248}{175}x^2+\frac{255968}{13475}x+\frac{1061376}{175175}\right).$$

We can use this formulation to consider the limit of the coefficients as p tends to infinity. In particular we have the following lemma:

Lemma 7. *For $r \in \mathbb{Z}_{\ge 0}$, let $a_r(x)$ be the polynomials given in Lemma 6. Then we have*

$$\lim_{p\to\infty}\sqrt{p}\frac{a_r(p)}{\pi}\int_{-\infty}^{\infty}\frac{\sin^p\theta}{\theta^{p-2r}}\mathrm{d}\theta = \frac{1}{2^{2r}}\sqrt{\frac{6}{\pi}}\binom{2r}{r}.$$

Proof. Apply the substitution $\theta = t/\sqrt{p}$ to give

$$\sqrt{p}\int_{-\infty}^{\infty}\frac{\sin^p(\theta)}{\theta^{p-2r}}\mathrm{d}\theta = \sqrt{p}\int_{-\infty}^{\infty}\left(\frac{\sqrt{p}}{t}\right)^{p-2r}\sin^p\left(\frac{t}{\sqrt{p}}\right)\frac{\mathrm{d}t}{\sqrt{p}} = \frac{1}{p^r}\int_{-\infty}^{\infty}t^{2r}\left(\frac{\sqrt{p}}{t}\sin\left(\frac{t}{\sqrt{p}}\right)\right)^p\mathrm{d}t$$

and since $a_r(p)$ is a polynomial of degree r with leading coefficient $(6^r r!)^{-1}$ and

$$\frac{\sqrt{p}}{t}\sin\left(\frac{t}{\sqrt{p}}\right) = 1-\frac{t^2}{6p}+O(p^{-2})$$

we have under the assumption that we can interchange the limit and the integral, which we do not prove, that

$$\lim_{p\to\infty}\sqrt{p}\frac{a_r(p)}{\pi}\int_{-\infty}^{\infty}\frac{\sin^p\theta}{\theta^{p-2r}}\mathrm{d}\theta = \frac{1}{\pi 6^r r!}\int_{-\infty}^{\infty}t^{2r}e^{-t^2/6}\mathrm{d}t.$$

The integral can be evaluated for a positive integer r by using integration by parts with $u = t^{2r-2}$ and $v = -3e^{-t^2/6}$ to give

$$\int_{-\infty}^{\infty} t^{2r} e^{-t^2/6} \mathrm{d}t = 3(2r-1)\int_{-\infty}^{\infty} t^{2r-2} e^{-t^2/6} \mathrm{d}t = \frac{3(2r)(2r-1)}{2r}\int_{-\infty}^{\infty} t^{2(r-1)} e^{-t^2/6} \mathrm{d}t$$

which by a simple inductive argument and the integral $\int_{-\infty}^{\infty} e^{-t^2/6} = \sqrt{6\pi}$ gives

$$\int_{-\infty}^{\infty} t^{2r} e^{-t^2/6} \mathrm{d}t = \sqrt{6\pi}\left(\frac{3}{2}\right)^r \frac{(2r)!}{r!}$$

and hence

$$\lim_{p\to\infty} \sqrt{p}\frac{a_r(p)}{\pi}\int_{-\infty}^{\infty} \frac{\sin^p\theta}{\theta^{p-2r}}\mathrm{d}\theta = \frac{\sqrt{6\pi}(\frac{3}{2})^r(2r)!}{\pi 6^r (r!)^2} = \frac{1}{2^{2r}}\sqrt{\frac{6}{\pi}}\binom{2r}{r}$$

as required. □

Taking $r = 0$ shows the leading coefficient ℓ_p satisfies $\lim_{p\to\infty}\sqrt{p}\ell_p = \sqrt{6/\pi}$ as was claimed in section 3.2. Further, replacing $a_r(p)$ by $a_r(p-1)\frac{p-1-2r}{p-1}$ has no effect on the limit. Finally we remark that for positive n

$$\sum_{0\le r<p/2} \frac{1}{2^{2r}}\binom{2r}{r} n^{p-1-2r} \le \sum_{r=0}^{\infty} \frac{1}{2^{2r}}\binom{2r}{r} n^{p-1-2r} = \frac{n^{p-1}}{\sqrt{1-n^{-2}}} = \frac{n^p}{\sqrt{n^2-1}}$$

so that by the fact, which we haven't proved, that $\sqrt{p}$ times the coefficient of n^{p-1-2r} in the expansion of $B_w(d,p)$ is increasing for $p > 2$ as p increases, we can conclude that for $p > 2$

$$B_w(d,p) \le \sqrt{\frac{6}{\pi p(n^2-1)}}n^p.$$