On the failure probability of ThreeBears

Mike Hamburg[*]

March 25, 2019

Abstract

Key exchange algorithms based on Ring and Module Learning With Errors (RLWE and MLWE) trade efficiency against failure probability. Some systems – such as LAC, Round5 and ThreeBears– reduce failure probability by using an error-correcting code. This improves efficiency, but makes the failure probability difficult to evaluate rigorously, and risks dramatically underestimating that failure probability.

In this note, we describe work to bound the failure probability of ThreeBears, rather than estimating it. We did not quite succeed in proving a rigorous bound – our estimates are conservative but include heuristics. We primarily studied the failure probability per message. We also investigated the success probability for straightforward CCA attacks requiring at most $2^{\{128,192,256\}}$ pre-quantum or post-quantum work. Our techniques may also be applicable to Round5 and LAC.

1 Introduction and related work

Several candidates for post-quantum encryption and key exchange are based on the LPR encryption system [LPR10], which derives its security from the Ring Learning With Errors (RLWE) problem. This encryption technique was popularized by the NewHope key exchange algorithm [ADPS15, PAA+17], and is also used in LAC [LLJ+17]. Other LPR-like systems in-

[*] Rambus

clude Kyber [BDK+17] which uses Module Learning With Errors; ThreeBears [Ham17] which uses Integer Module Learning With Errors (I-MLWE); and Round5 [BGL+18] which uses Ring Learning With Rounding (RLWR). Collectively, we will call these systems "LPR variants".

When used for encryption with a long-term key, lattice schemes such as LPR must be protected against chosen-ciphertext attacks [HNP+03], typically using a variant of the Fujisaki-Okamoto transform [FO99]. However, chosen-ciphertext attacks are still possible if the underlying encryption algorithm has imperfect correctness; that is, if it has a nonzero probability of decryption failure.

The correctness of these algorithms depends, roughly, on the convolution of i.i.d. random vectors not exceeding certain thresholds. As such, LPR variants usually have a nonzero probability of decryption failures. The failure probability is related to the amount of noise added, which also influences the security, and this creates a tradeoff of security vs. performance vs. failure probability. Some LPR variants, such as NTRU LPRime [BCLv17], replace the i.i.d. random vectors with fixed-weight ones and use parameters that eliminate decryption failures. This makes analysis simpler at the cost of performance and bandwidth.

In the opposite direction, the ThreeBears, Round5 and LAC algorithms use error-correcting codes to reduce failure probability.[1] This technique improves efficiency, but makes it very difficult to accurately assess failure probability – a problem that threatened CCA attacks on earlier versions of both LAC [AS18, Ham18a] and Round5 [Ham18b]. It is usually easy to explicitly calculate the probability p that a given bit of the message will decrypt incorrectly. The naïve estimate is that an e-error correcting code on n bits would reduce the failure rate to around $\binom{n}{e+1} \cdot p^{e+1}$, since a failure does not occur unless at least $e+1$ bits flip. However, the actual failure rate may be much higher, because failures are correlated.

[1] Arguably, so too does the 2015 version of NewHope: its reconciliation mechanism is equivalent to a 4-bit repetition code with soft-decision decoding.

1.1 Related work: D'Anvers-Vercauteren-Verbauwhede

Two recent papers by D'Anvers, Vercauteren and Verbauwhede [DVV18b, DVV18a] analyze CCA attacks on RLWE schemes, and the dependence of failure rates on the norm of the noise in the ciphertext and private key.

The original failure analysis for ThreeBears used a similar technique to relate the decryption failure rate to the norm of the noise in the ciphertext. Additionally, it takes into account a detail that affects ThreeBears, but not most other RLWE schemes. ThreeBears' modulus has the form $N = x^D - x^{D/2} - 1$, so a single large coefficient can contribute to two large coefficients when reduced mod N. As a result, bit flips are correlated if they are separated by $D/2$ positions. A more serious version of this problem was present in the first version of Round5.

We were not entirely satisfied with this sort of analysis, because there may be other, unknown forms of correlation that contribute to the failure rate. Accordingly, in this work we aimed to strictly and rigorously bound (i.e. overestimate) the failure rate of ThreeBears, rather than estimating it accurately. We were not quite able to accomplish this – we needed a few heuristics and floating-point arithmetic.

Our analysis of CCA attacks is roughly the same as in [DVV18b], but reworked to mesh with our more conservative failure estimation technique. It still analyzes only a particular, relatively straightforward attack. However, we did not attempt to analyze how many decryption failures would be enough to recover a ThreeBears key. [DVV18b] estimates work to produce many failures using a non-adaptive attack, but we were concerned that an adaptive attack would be much more powerful, so we only model the effort required to find the first decryption failure.

2 Preliminaries

2.1 Notation

Let $\mathbb{R}$ be the real numbers, and $\mathbb{R}_{\geq 0}$ be the non-negative real numbers. Let $\langle \vec{x},\ \vec{y} \rangle$ be the inner product of two vectors over $\mathbb{R}^n$.

Let $\left[f(x) : x \leftarrow D \right]$ denote the expectation of a function f over a probability distribution D. For $p > 0$ and $f(x) \geq 0$, let $\left[f(x) : x \leftarrow D \right]_p$ be the p-norm (a.k.a. the weighted p-power mean) of f over D, namely $\sqrt[p]{\left[f(x)^p : x \leftarrow D \right]}$.

We make heavy use of the *weighted power means inequality*, which states that if $p \geq q > 0$ and $f(x) \geq 0$ for all x, then

$$\left[f(x) : x \leftarrow D \right]_p \geq \left[f(x) : x \leftarrow D \right]_q$$

2.2 Encryption from Learning with Errors

The key exchange algorithms we study are derived from the Lyubashevsky-Peikert-Regev (LPR) RLWE encryption system [LPR10], which we summarize as follows. Let R be a ring, and let χ be a distribution on R, producing elements which are in some way "small" with high probability. Let d be a module dimension. Let Encode: $M \to R$ map a message into the ring and Decode: $R \to M$ an inverse map, which is immune to "small" changes. Specifically, suppose there is a subset Safe $\subset R$ of the ring, so that for all messages m and for all elements $\epsilon \in$ Safe,

$$\text{Decode}(\text{Encode}(m) + \epsilon) = m$$

Let Round be an algorithm which rounds a ring element to reduce the number of bits required to represent it. The encryption and decryption algorithms are shown in Figure 1. This work can easily be extended to cases where other elements of the public key or ciphertext are rounded, or where a different χ is used for the ciphertext and secret key or for s and ϵ. The

```
$\mathrm{Keygen}()$ :
    $s \leftarrow \chi^d$;  $\epsilon \overset{R}{\leftarrow} \chi^d$;
    $A \overset{R}{\leftarrow} R^{d \times d}$;
    $X \leftarrow A \cdot s + \epsilon$;
    $\mathrm{sk} \leftarrow s$;  $\mathrm{pk} \leftarrow (A, X)$;

$\mathrm{Dec}(\mathrm{sk}, c)$ :
    $s \leftarrow \mathrm{sk}$;  $(Y, Z') \leftarrow c$;
    $m' \leftarrow \mathrm{Decode}(Z' - Ys)$

$\mathrm{Enc}(\mathrm{pk}, m)$ :
    $(A, X) \leftarrow \mathrm{pk}$;
    $s' \leftarrow \chi^d$;  $\epsilon' \overset{R}{\leftarrow} \chi^d$; $\epsilon'' \leftarrow \chi$;
    $Y \leftarrow {s'}^\top A + {\epsilon'}^\top$;
    $Z \leftarrow {s'}^\top X + \epsilon'' + \mathrm{Encode}(m)$;
    $Z' \leftarrow \mathrm{Round}(Z)$;
    $c \leftarrow (Y, Z')$;
```

Figure 1: LPR variant under study.

above description applies to ThreeBears, Kyber, NewHope and LAC, except that ThreeBears doesn't compute $Z' - Ys$ in the ring, instead extracting m by comparing the digits of Z' with those of Ys. When written as a deterministic algorithm, Enc takes an additional input: a string ρ which is used to choose the $2d + 1$ samples from χ.

Let $r := Z' - Z$ be the information lost by rounding Z. The encryption scheme's correctness stems from the fact that

$$\begin{aligned} Z' - Ys &= (Z' - Z) + {s'}^\top X + \epsilon'' + \mathrm{Encode}(m) - ({s'}^\top A + {\epsilon'}^\top)s \\ &= r + {s'}^\top (A \cdot s + \epsilon) + \epsilon'' + \mathrm{Encode}(m) - ({s'}^\top A + {\epsilon'}^\top)s \\ &= \mathrm{Encode}(m) + r + {s'}^\top \epsilon + \epsilon'' - {\epsilon'}^\top s \end{aligned}$$

Therefore $\mathrm{Decode}(Z' - Ys) = m$, and decryption succeeds, so long as $(r + {s'}^\top \epsilon + \epsilon'' - {\epsilon'}^\top s)$ is sufficiently "small", e.g. the absolute value each of its coefficients is less than some threshold t.

2.3 Details for ThreeBears

For ThreeBears with Z' rounded to $\ell = 4$ bits and a digit $x = 2^{10}$, the coefficients of $\epsilon, \epsilon', \epsilon'', s, s'$ are drawn i.i.d. from the same distribution χ_v of variance v, where if $v \leq 1/2$, then

$$\chi_v := \begin{cases} 0 & \text{with probability } 1 - v \\ \pm 1 & \text{with probability } v/2 \text{ each} \end{cases}$$

and if $1/2 < v \leq 1$, then

$$\chi_v := \begin{cases} 0 & \text{with probability } (5 - 2v)/8 \\ \pm 1 & \text{with probability } 1/4 \text{ each} \\ \pm 2 & \text{with probability } (2v - 1)/16 \text{ each} \end{cases}$$

Furthermore, an error in position i cannot occur unless

$$y_i - z_i + r_i - 1 \geq x/4 = 256 \qquad \text{or} \qquad y_i - z_i + r_i \leq -x/4 = -256$$

where y_i is the ith digit of Ys and z_i is the ith digit of Z', and the rounding component $r_i \overset{R}{\leftarrow} [1 - x/2^{\ell+1}, x/2^{\ell+1}] = [-31, 32]$. See Appendix A for a proof of this.

The rest of this paper calculates $y_i - z_i$ using polynomial arithmetic. The difference between this and integer arithmetic is that carries might propagate into y_i and z_i. However, with overwhelming probability (around $1 - 2^{-1000}$) the carries differ by at most 1. This lowers the threshold to 255 instead of 256.

2.4 Fujisaki-Okamoto transform

We follow [DVV18a] in studying CCA attacks on LWE-based encryption schemes that use some variant of the Fujisaki-Okamoto (FO) transform [FO99]. We give a simplified version this transform as follows.

The FO transform is built on a public-key encryption scheme. Let K and $\hat{K}$ be the sets of public and private keys for this scheme, respectively. Let

```
$\mathrm{Encaps}(\mathrm{pk})$ :
$m \overset{R}{\leftarrow} M$;
$(\rho, \tau, s) \leftarrow H(\mathrm{pk}, m)$;
$c \leftarrow \mathrm{Enc}(\mathrm{pk}, m, \rho)$;
return $((c, \tau), s)$;
```

```
$\mathrm{Decaps}(\mathrm{sk}, \mathrm{pk}, (c, t))$ :
$m' \leftarrow \mathrm{Dec}(\mathrm{sk}, c)$;
$(r', t', s') \leftarrow H(\mathrm{pk}, m')$;
if $(\mathrm{Enc}(\mathrm{pk}, m', \rho'), \tau') = (c, \tau)$ then return $s$;
else return $\perp$;
```

Figure 2: Simplified Fujisaki-Okamoto transform from encryption to key exchange. The LAC, Kyber, Round5 and ThreeBears algorithms use this approach, but with a zero-length tag τ.

Enc : $(K \times M \times R)$ be a deterministic encryption function taking a public key $\mathrm{pk} \in K$, a message $m \in M$, and a random string $\rho \in R$, and returning a ciphertext $c \in C$. Let Dec : $(\hat{K} \times C) \to M$ be the corresponding decryption algorithm. Let S be a space of symmetric keys. Let T be a set of "tags"; these are used in certain proofs of security for FO, and are typically either absent or the same size as the message. Let $H : (K \times M) \to (R \times T \times S)$ be a hash function, which is modeled as a random oracle. Then we can perform a key exchange which takes a public key and returns a ciphertext and a shared symmetric key, as shown in Figure 2.

Decapsulation can fail by returning a special symbol $\perp \notin S$, or succeed by returning an element of S. It can succeed only if (c, τ) is a *well-formed ciphertext*, meaning a possible output of the Encaps routine; but it might also fail on some well-formed ciphertexts. This will happen if $H(\mathrm{pk}, m) = (\rho, \tau, s)$ such that

$$\mathrm{Dec}(\mathrm{sk}, \mathrm{Enc}(\mathrm{pk}, m, \rho)) \neq m$$

for some message m and keypair sk, pk.

The intuition behind the FO transform is that an adversary cannot learn much by asking a would-be victim to decrypt chosen messages. This is because (under suitable assumptions) the adversary cannot create a well-formed ciphertext (c, τ) without knowing that it is well-formed or without knowing m. Therefore the adversary (and the CCA simulator) knows the complete output of Decaps without calling it.

However, the adversary does learn whether a well-formed ciphertext decrypt successfully or not. This information typically weakens the security of the system, and over several failed decryptions might make it feasible or even easy to recover the private key. In this paper, we assume that any failed decryption may result in a successful attack.[2] We believe this conservatism is warranted: while a single decryption failure is unlikely to break the private key outright, it may give enough information that further decryption failures will be significantly easier to cause. Furthermore, if the adversary is querying many different keys, the first failure is likely to come from a weak key, i.e. one with larger-than-normal amounts of noise, and that key will be more prone to further failures.

3 CCA attack model

In existing CCA attacks, the adversary has very limited ability to predict which messages will fail to decrypt with a given key. Nor are existing attacks able to determine which keys are more likely to have decryption failures. But such attacks are have not been proved impossible, so security proofs of the FO transform make use only of the scheme's overall failure probability. With attacks on n public keys, they would instead have a term like

$$\left[\left(\max_i \ \Pr\left(\mathsf{fail} : \text{encrypt to } k_i\right)\right) : k_i \leftarrow \text{Keygen}() \text{ for } i = 0 \text{ to } n-1\right]$$

For provable security, we cannot go beyond these calculations. But we also wish to study attacks that follow the pattern of existing ones, which we instantiate as the following "key-agnostic" attack.

First, the attacker chooses a target public key pk at random from a large set. He then encrypts a random message to that public key. The ciphertext noise

[2] This makes "failure" nearly synonymous with "success".

is $(s', \epsilon', \epsilon'')$, plus rounding noise $r = \text{Round}(Xs' + \epsilon' + \epsilon'') - (Xs' + \epsilon' + \epsilon'')$. Since the public key was chosen at random, we model r as independent of $(s', \epsilon', \epsilon'')$; this is likely true for most LPR variants but especially for ThreeBears[3]. Without recovering any information about the private randomness (s, ϵ), the adversary (and this paper) estimates the probability that decryption will fail as if they were chosen randomly from χ:

$$\Pr(\mathsf{Fail}) \approx \Pr\left(r + {s'}^{\top}\epsilon + \epsilon'' - {\epsilon'}^{\top}s \notin \text{Safe} : (s, \epsilon) \leftarrow \chi^{2d}\right)$$

He then sends the message if the estimated $\Pr(\mathsf{Fail})$ reaches some threshold[4] of the adversary's choice, or otherwise falls into some set F_{pk}. If it does cause a decryption failure, then the adversary somehow breaks that recipient's key.

Note that choosing a new key each time makes the attack successful more often, since by assumption the attacker does not know which keys may be weak. If each key k is queried n_k times, success probability for the attack is

$$\Pr(\text{attack succeeds}) = 1 - \prod_k (1 - \Pr(\mathsf{Fail} : k))^{n_k}$$

By the weighted power means inequality, for $n_k > 1$,

$$\left[(1 - \Pr(\mathsf{Fail} : k))^{n_k} : k \leftarrow \text{Keygen}()\right] \geq \left[1 - \Pr(\mathsf{Fail} : k) : k \leftarrow \text{Keygen}()\right]^{n_k}$$

where the left-hand side is the probability of no failures in n_k queries to the same key, and the right-hand side is the probability of no failures in n_k queries to n_k different randomly-chosen keys. So sending each query to a different key maximizes the success probability of the attack.

[3] Treat X as being chosen after $(s', \epsilon', \epsilon'')$. For a random public key, X is indistinguishable from random. Furthermore ThreeBears' ring has no zero divisors, so Xs' is uniform if $s' \neq 0$, which happens with overwhelming probability.

[4] It might be possible to optimize this threshold, or to analyze systems with an exact threshold of, say, 2^{-64}. Our analysis doesn't do this, and instead analyzes the expected value of $\Pr(\mathsf{Fail})$ and analogous expressions for quantum attack. While we would prefer a tighter analysis, this does have the advantage that it will probably still be an overestimate if black-box Grover's algorithm is not optimal.

Let c abbreviate the ciphertext and its randomness. For q queries, the attack requires work proportional to

$$q/\Pr\left(c \in F_{\mathrm{pk}} : (\mathrm{sk}, \mathrm{pk}) \leftarrow \mathrm{Keygen}(), m \leftarrow M, c \leftarrow \mathrm{Enc}(\mathrm{pk}, m)\right)$$

which we abbreviate as $1/\Pr\left(c \in F_{\mathrm{pk}}\right)$. The attack succeeds with probability at most

$$q \cdot \Pr\left(\mathsf{Fail} : c \in F_{\mathrm{pk}}\right)$$

As with many brute-force attacks, this key-agnostic attack has its work, success probability and number of queries all directly proportional. In the lower limit of 1 query, the attack works with one unit of work, and success probability equal to the failure probability of the encryption scheme. We are more interested in the opposite limit, in which the attack works with high probability. As in [DVV18b], we estimate the number of queries required so that this latter quantity approaches 1, in which case the attack succeeds with probability about $1 - 1/e$.

We use two different approaches to model quantum attacks using Grover's algorithm. Grover's algorithm is depth-bounded, speeding up queries by a factor of about the maximum quantum query depth, which is realistically less than 2^{64}. See e.g. [AHU18], Lemma 2: if P is a random predicate modeled as a quantum random oracle, which obtains for any input with probability at most ϵ, then the probability to find a preimage using q queries at depth d is at most $4(q+1)(d+1)\epsilon$, where the leading 4 is not tight. For simplicity, we model this as simply allowing the adversary a factor of 2^{64} more work.

Alternatively, we could allow the attacker infinite depth. This replaces the amount of time taken per query with the square root of the same. We unify these two cases by calculating the work as $1/\sqrt[g]{\Pr\left(c \in F_{\mathrm{pk}}\right)}$, where $g = 1$ for classical attack and $g = 2$ for quantum.

4 Bounding CCA attack success

Let's begin bounding the success probability of CCA attacks. We may need to analyze more than one way a decryption could fail. In the case of

ThreeBears, we need to break up failures according to the bit positions that decrypted incorrectly. So we will use a set F of failure modes, such that decryption failure cannot occur unless some $\mathsf{f} \in F$ occurs.

Theorem 1 (g, k bounding technique)**.** *Let C be a probability distribution. Suppose a stateless algorithm $\mathcal{A}$ repeatedly chooses $c \leftarrow C$ from an oracle, then chooses to either query c or not. If $\mathcal{A}$ queries c, then the events in some collection F of events may occur. $\mathcal{A}$ repeats this process until it makes q queries. Let δ be the probability that any event in F happens on each query.*

Let $g \in \{1, 2\}$. Let w be the expected number of samples of C per query, and let $\text{work} := q\sqrt[g]{w}$*. Let*

$$\delta_k := \sum_{\mathsf{f} \in F} \left[\Pr(\mathsf{f} : c) : c \leftarrow C\right]_k$$

Then

$$\Pr(\text{an event in } F \text{ occurs}) \quad \leq \quad q\delta \quad \leq \quad q^{1-g/k} \cdot \text{work}^{g/k} \cdot \delta_k$$

Furthermore if $k > g$ and $\delta \geq 1/q$, then

$$q \geq (\text{work}^g \cdot \delta_k^k)^{-1/(k-g)}$$

Finally, if $\mathcal{A}$ queries every sample, then $\delta = \delta_1$.

Proof. See Appendix B. □

4.1 Hoeffding bounds

To estimate δ_k, we use Hoeffding's technique for bounding probabilities. Let D be a probability distribution, and $t \in \mathbb{R}$ be a threshold. Then

$$\Pr(x \geq t : x \leftarrow D) \leq \min_{\lambda \geq 0} \left[e^{\lambda(x-t)} : x \leftarrow D\right]$$

This follows from the observation that $e^{\lambda(x-t)} \geq 1$ whenever $x \geq t$. Likewise, let D be a probability distribution over $\mathbb{R}^n$, and let $\vec{t} \in \mathbb{R}^n$ be a threshold.

Then

$$\Pr\left(\vec{x} \succeq \vec{t} : \vec{x} \leftarrow D\right) \le \min_{\vec{\lambda} \succeq 0} \left[e^{\langle \vec{\lambda},\ \vec{x}-\vec{t}\rangle} : \vec{x} \leftarrow D\right]$$

where $\vec{x} \succeq \vec{t}$ means that each coordinate of $\vec{x}$ is greater than or equal to the same coordinate of $\vec{t}$.

4.2 Applying the Hoeffding technique

Let Q be the rank-3 tensor (rank-2 contravariate and rank-1 covariate) that maps the recipient's noise (s, ϵ) and the adversary's noise (s', ϵ') to the difference $s\epsilon' - s'\epsilon$. Treat the inputs as vectors in $\mathbb{R}^{2dD}$, where D is the ring dimension and d is the module rank. Treat the output as a vector in $\mathbb{R}^b$, covering only the positions which are used to transmit the message. Consider a failure class f consisting of one or more positions i and signs sign_i by which

$$\text{sign}_i \cdot (s\epsilon' - s'\epsilon + \epsilon'' + r)_i \ge t_i$$

Let V_{f} be the set of vectors $\vec{\lambda} \succeq 0$ whose coefficients are nonzero only in those positions. For a vector $\vec{\lambda}$, let $\vec{\lambda}_{\text{signed}}$ be the vector whose ith coefficient is $\text{sign}_i \lambda_i$. Then

$$\begin{aligned}
\Pr\left(\mathsf{f} : s', \epsilon', \epsilon'', r\right) &= \Pr\left(\text{sign}_i \cdot (s\epsilon' - s'\epsilon + \epsilon'' + r) \succeq t_i : s, \epsilon \leftarrow \chi^{2dD}\right) \\
&\le \min_{\vec{\lambda} \in V_{\mathsf{f}}} \left[e^{\langle Q(s,\epsilon;\ s',\epsilon'),\ \vec{\lambda}_{\text{signed}}\rangle + \langle \vec{\lambda}_{\text{signed}},\ \epsilon''+r\rangle - \langle \vec{\lambda},\ \vec{t}\rangle} : s, \epsilon \leftarrow \chi^{2dD}\right] \\
&= \min_{\vec{\lambda} \in V_{\mathsf{f}}} \left[e^{\langle Q(s,\epsilon;\ s',\epsilon'),\ \vec{\lambda}_{\text{signed}}\rangle} : s, \epsilon \leftarrow \chi^{2dD}\right] \cdot e^{\langle \vec{\lambda}_{\text{signed}},\ \epsilon''+r\rangle - \langle \vec{\lambda},\ \vec{t}\rangle}
\end{aligned}$$

under the assumption that r, s' and ϵ' are independent. Therefore

$$\begin{aligned}
&\left[\Pr\left(\mathsf{f} : s', \epsilon', \epsilon'', r\right) : s', \epsilon', \epsilon'', r\right]_k \\
&\quad \le \min_{\vec{\lambda} \in V_{\mathsf{f}}} \left(\begin{array}{c} \left[\left[e^{\langle Q(s,\epsilon;\ s',\epsilon'),\ \vec{\lambda}_{\text{signed}}\rangle} : s, \epsilon \leftarrow \chi^{2dD}\right]^k : s', \epsilon' \leftarrow \chi^{2dD}\right] \\ \cdot \left[e^{k\langle \vec{\lambda}_{\text{signed}},\ \epsilon''+r\rangle - k\langle \vec{\lambda},\ \vec{t}\rangle} : \epsilon'', r\right] \end{array} \right)^{\frac{1}{k}}
\end{aligned}$$

The lower term $\left[e^{k\langle\vec{\lambda}_{\text{signed}},\ \epsilon''+r\rangle-k\langle\vec{\lambda},\ \vec{t}\rangle} : \epsilon'', r\right]$ is easy enough to compute, since the coefficients of ϵ'' and those of r are all independent.

The upper term is more difficult. To simplify it slightly, note that s and ϵ, and likewise s' and ϵ', are $2d$ values chosen i.i.d from the same symmetric distribution. Furthermore, the bilinear form $\langle Q(s,\epsilon;\ s',\epsilon'),\ \lambda_{\text{signed}}\rangle$ is block-diagonal with d copies of some block $M_{\vec{\lambda}}$, and another d copies of its negation (which has the same expectation due to symmetry). So the upper term is equal to

$$\left[\left[e^{\vec{x}^\top M\vec{y}} : \vec{y}\leftarrow\chi\right]^k : \vec{x}\leftarrow\chi\right]^{2d}$$

Therefore let

$$\eta(M,k,\chi) := \left[\left[e^{\vec{x}^\top M_{\vec{\lambda}}\vec{y}} : \vec{y}\leftarrow\chi\right]^k : \vec{x}\leftarrow\chi\right]$$

so that

$$\begin{aligned}&\left[\Pr\left(\mathsf{f} : s',\epsilon',\epsilon'',r\right) : s',\epsilon',\epsilon'',r\right]_k\\ &\qquad\qquad \le \min_{\vec{\lambda}\in V_{\mathsf{f}}}\left(\eta(M_{\vec{\lambda}},k,\chi)^{2d}\cdot\left[e^{k\langle\vec{\lambda}_{\text{signed}},\ \epsilon''+r\rangle-k\langle\vec{\lambda},\ \vec{t}\rangle} : \epsilon'',r\right]\right)^{\frac{1}{k}}\end{aligned}$$

Estimating the ROM proof term is very similar, but in this case the calculation is done with the key randomness (s,ϵ) known and the ciphertext randomness $(s',\epsilon',\epsilon'')$ unknown. Swapping (s,ϵ) for (s',ϵ') has no effect on most schemes due to symmetry, and making (s',ϵ') unknown simply replaces

$$\left[e^{k\langle\vec{\lambda}_{\text{signed}},\ \epsilon''+r\rangle-k\langle\vec{\lambda},\ \vec{t}\rangle} : \epsilon'',r\right]^{\frac{1}{k}} \qquad \text{with} \qquad \left[e^{\langle\vec{\lambda}_{\text{signed}},\ \epsilon''+r\rangle-\langle\vec{\lambda},\ \vec{t}\rangle} : \epsilon'',r\right]$$

It remains to compute, or at least estimate, η. We will show two methods to do this: one exact (up to numerical precision) and one approximate.

4.3 Exact solution: pebbling

Note that $\eta(M, k, \chi)$ is invariant under permutations of the rows and columns of M. Suppose we have rearranged M as block-diagonal; then

$$\eta(M, k, \chi) = \prod_{\text{block } B} \eta(B, k, \chi)$$

While we do not know how to compute $\eta(B, k, \chi)$ in general, the following theorem shows that it can be computed when each block is close enough to diagonal:

Theorem 2 (Pebbling algorithm)**.** *Suppose that for a $b \times b$ matrix B, $B_{i,j} \neq 0$ only when*

$$0 \leq i - j \ mod\ b \leq \ell$$

for some small lag ℓ. Suppose the distribution χ can take n different values in each coordinate. In this case, there is an algorithm which computes $\eta(B, k, \chi)$ numerically in time $O(b \cdot n^{3\ell})$.

Proof. We present this algorithm in Appendix C. □

4.4 Approximate solution: Gaussian heuristic

The pebbling technique takes time exponential in ℓ, so it is practical only when ℓ is small. A simpler but imprecise solution is to approximate χ as Gaussian with variance $v = \text{var}(\chi)$, in which case $\eta(M, k, \chi)$ can be determined exactly:

Theorem 3 (Gaussian heuristic)**.** *If χ is the Gaussian distribution with mean 0 and variance v, and if the matrix $I - kv^2 M^\top M$ is positive-definite, then*

$$\eta(M, k, \chi) = \frac{1}{\sqrt{\det(I - kv^2 M^\top M)}}$$

Proof. See Appendix D. □

This heuristic tends to be dominated by the long tails of the Gaussian distribution, which a binomial distribution lacks. Therefore it generally gives answers larger than the actual η, but we haven't proved this. In particular, while $\eta(M, k, \chi)$ is always finite for bounded distributions χ, the Gaussian approximation can diverge.

4.5 Putting it all together

We recall from Theorem 1 that for all $k \geq g$,

$$\Pr\left(\text{Any failure occurs}\right) \leq q^{(k-g)/k} \cdot \sum_{\mathsf{f} \in F} \left[\Pr\left(\mathsf{f} : c\right) : c \leftarrow C\right]_k \cdot \text{work}^{g/k}$$

and for a failure class f, we can bound:

$$\left[\Pr\left(\mathsf{f} : c\right) : c \leftarrow C\right]_k \leq \min_{\vec{\lambda} \in V_{\mathsf{f}}} \left(\eta(M_{\vec{\lambda}}, k, \chi)^{2d} \cdot \left[e^{\langle \vec{\lambda}_{\text{signed}},\ \epsilon'' + r\rangle - \langle \vec{\lambda},\ \vec{t}\rangle} : \epsilon'', r\right]\right)^{\frac{1}{k}}$$

Furthermore Theorem 2 gives us a way to compute $\eta(M_{\vec{\lambda}}, k, \chi)$ in some cases, and Theorem 3 gives us a way to estimate it in all other cases. For a given k, we can therefore compute or estimate all the inner terms efficiently, and can approximate the minimum for each f using gradient descent on $\vec{\lambda}$. The bound from Theorem 1 then lets us conservatively estimate the overall failure probability and the success probability for the key-agnostic attack.

5 Results for ThreeBears

We ran this technique on the CCA-secure instances of ThreeBears. ThreeBears sends data at the $256 + 18$ positions $\{0, \ldots, 136\}$ and $\{175, \ldots, 311\}$. There are two special classes of failures for ThreeBears:

- "Opposed" failures of the form $\{i, i + D/2, j\}$.
- "Sequential" failures of the form $\{i, (i + j)/2, j\}$.

We calculated for all classes of failures up to the following symmetry classes:

- Toggling the sign of the failure in all positions.

- For opposed failures $\{i, i+D/2, j\}$ and $i-j$ isn't divisible by 4, toggling the sign of the failure at j.
- For sequential failures, toggling the sign of the middle element.
- Reflecting all positions across the midpoint of 156.

For most of our calculations, we used the pebbling method for opposed an sequential failures, and the Gaussian heuristic for the other cases. However, BabyBear's larger variance makes the pebbling method quite slow. Therefore we also used the Gaussian heuristic for sequential failures on some calculations for BabyBear. These calculations are marked †, and are likely to give higher estimates than otherwise.

We tracked at all times the contribution $p_{\max}$ of the most significant failure class, and skipped certain calculations which would contribute more than 2^{24} times less. We observed that if f_1 and f_2 had the same spacing between failing bit positions, but f_2's coefficients were further on average from $D/2$, the class f_1 contributed more to the overall estimates. (This is because ThreeBears' reduction mod N amplifies the noise more in the center digits than at the ends.) Accordingly, if f_1 contributed $p \leq p_{\max}/2^{32}$, we skipped computing all shifts f_2, instead entering $p \cdot 2^8$. The 2^8 is a fudge factor, in case the contribution of f_2 should somehow be greater than p. This elision never contributes more than $p_{\max}/2^{24}$. We also aborted gradient descent either immediately, or after one descent step, upon reaching $p_{\max}/2^{24}$.

We didn't test all values of k for reasons of compute time. Instead we chose $k = 1$ to estimate the total failure probability, and a few values $k > 2$ based on runs on a subset of the failure classes. The results are shown in Table 1.

We also aimed to estimate the expected maximum failure probability in $K = 2^{64}$ keys, which is used in the provable IND-CCA security bound. This is shown in Table 2. We see that the best results are close to $K \cdot \delta$, where δ is the overall failure probability. This suggests, somewhat unsurprisingly, that failures are usually caused by a weak (failure-prone) key combined with a failure-prone ciphertext.

		Failure prob.		Key-agn. CCA “bound”		
System	work	claim	“bound”	maxdepth	queries	k
BabyBear r1 ($v = 5/8,\ d = 2$)	128	-135	-128	classical	∞	1
				64	54	4
				∞	48	4
MamaBear r1 ($v = 1/2,\ d = 3$)	192	-147	-141	classical	65	8
				64	57	8
				∞	45	8
PapaBear r1 ($v = 3/8,\ d = 4$)	256	-188	-188	classical	99	6
				64	86	6
				∞	62	8
BabyBear r2 ($v = 9/16,\ d = 2$)	128	-156	-156	classical	∞	1
				64	$68^\dagger$	2
				∞	$68^\dagger$	2
MamaBear r2 ($v = 13/32,\ d = 3$)	192	-206	-206	classical	∞	1
				64	110	3
				∞	93	3
PapaBear r2 ($v = 5/16,\ d = 4$)	256	-256	-256	classical	∞	1
				64	124	3
				∞	97	3

Table 1: Conservative estimates (“bounds”) of $\log_2$ failure probabilities and $\log_2$ number of decryption queries required for key-agnostic CCA attack. The work column is the allowed work for the system’s claimed NIST class. The maxdepth column indicates the $\log_2$ maximum depth of the quantum computation, which is 0 for a classical computer. Queries=∞ means that the attack is ruled out by the work limit.

System	work	single-key	2^{64} keys	k
BabyBear r2	128	-156	-103†	2
MamaBear r2	192	-206	-150	1.75
PapaBear r2	256	-256	-200	1.5

Table 2: Conservative estimates of the failure probability of the weakest key in 2^{64} keys, as required by the security proof.

6 Conclusions

We have shown a technique to conservatively estimate the failure probability of lattice cryptosystems that use error-correcting codes, in particular ThreeBears. The estimation technique uses a few heuristics: that numerical precision is sufficient; and in some cases the Gaussian heuristic for η. To save computation, we also assumed that failure probability falls (or at least rises by a factor of $\leq 2^8$) as the failing bit positions move away from the center, but in principle this is not needed. We have applied the technique to estimating work and success probabilities for a limited class of chosen-ciphertext attacks on ThreeBears. The same technique should be applicable to Round5.

6.1 Future work

We would like to tighten this analysis and improve its rigor. Possibly the Gaussian heuristic can be proved, perhaps with a variant of the Hanson-Wright inequality. We are also not sure that the straightforward attack is optimal, and would like to prove this for RWLE schemes.

Even if a tight rigorous bound cannot be obtained, we would like to overestimate the failure probability by a smaller margin. Perhaps multivariate Mills ratios are the place to start on this. Finally, hopefully this analysis can be applied to other proposed KEMs, such as Round5 or LAC.

6.2 Acknowledgements

Special thanks to Lydia Hamburg, Shaddin Dughmi, Mark Marson and Eric Chu for helpful discussions.

References

[ADPS15] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange - a new hope. Cryptology ePrint Archive, Report 2015/1092, 2015. `http://eprint.iacr.org/2015/1092`.

[AHU18] Andris Ambainis, Mike Hamburg, and Dominique Unruh. Quantum security proofs using semi-classical oracles. Cryptology ePrint Archive, Report 2018/904, 2018. `https://eprint.iacr.org/2018/904`.

[AS18] Jacob Alperin-Sheriff. Official comment: LAC. NIST PQC forum email list, 2018. `https://csrc.nist.gov/CSRC/.../round-1/official-comments/LAC-official-comment.pdf`.

[BCLv17] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. Ntru prime. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.

[BDK$^+$17] Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, and Damien Stehlé. CRYSTALS – kyber: a CCA-secure module-lattice-based KEM. Cryptology ePrint Archive, Report 2017/634, 2017. `http://eprint.iacr.org/2017/634`.

[BGL$^+$18] Sauvik Bhattacharya, Óscar García-Morchón, Thijs Laarhoven, Ronald Rietman, Markku-Juhani O. Saarinen, Ludo Tolhuizen, and Zhenfei Zhang. Round5: Compact and fast post-quantum public-key encryption. Cryptology ePrint Archive, Report 2018/725, 2018. https://eprint.iacr.org/2018/725.

[DVV18a] Jan-Pieter D'Anvers, Frederik Vercauteren, and Ingrid Verbauwhede. The impact of error dependencies on Ring/Mod-LWE/LWR based schemes. Cryptology ePrint Archive, Report 2018/1172, 2018. https://eprint.iacr.org/2018/1172.

[DVV18b] Jan-Pieter D'Anvers, Frederik Vercauteren, and Ingrid Verbauwhede. On the impact of decryption failures on the security of LWE/LWR based schemes. Cryptology ePrint Archive, Report 2018/1089, 2018. https://eprint.iacr.org/2018/1089.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany.

[Ham17] Mike Hamburg. Three bears. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

[Ham18a] Mike Hamburg. Official comment: LAC. NIST PQC forum email list, 2018. https://csrc.nist.gov/CSRC/.../round-1/official-comments/LAC-official-comment.pdf.

[Ham18b] Mike Hamburg. Official comment: Round5 = round2 + hila5. NIST PQC forum email list, 2018. https://csrc.nist.gov/CSRC/...1/official-comments/Round5-official-comment.pdf.

[HNP$^+$03] Nick Howgrave-Graham, Phong Q. Nguyen, David Pointcheval, John Proos, Joseph H. Silverman, Ari Singer, and William

Whyte. The impact of decryption failures on the security of NTRU encryption. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 226–246, Santa Barbara, CA, USA, August 17–21, 2003. Springer, Heidelberg, Germany.

[LLJ+17] Xianhui Lu, Yamin Liu, Dingding Jia, Haiyang Xue, Jingnan He, and Zhenfei Zhang. Lac. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23, French Riviera, May 30 – June 3, 2010. Springer, Heidelberg, Germany.

[PAA+17] Thomas Poppelmann, Erdem Alkim, Roberto Avanzi, Joppe Bos, Léo Ducas, Antonio de la Piedra, Peter Schwabe, and Douglas Stebila. Newhope. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.

A Details of ThreeBears error bounding

Let $\texttt{extract}_\ell(Z, i) := \left\lfloor \frac{z_i \cdot 2^\ell}{x} \right\rfloor$ be the top ℓ bits of z_i, where z_i is the i'th digit of Z in radix $x = 2^{10}$

ThreeBears sends in the ith position the value

$$\text{encr}_i := \texttt{extract}_4(Z, i) + 8 \cdot m_i \mod 16$$

It calculates

$$\begin{aligned} m'_i &:= \left\lfloor \frac{2 \cdot \mathtt{encr}_i - \mathtt{extract}_5(Ys, i)}{2^4} \right\rceil \mod 2 \\ &= \left\lfloor \frac{2 \cdot \mathtt{encr}_i - \mathtt{extract}_5(Ys, i) + 2^3}{2^4} \right\rfloor \mod 2 \\ &= \left\lfloor \frac{2 \cdot \lfloor z_i/2^6 \rfloor + 2 \cdot m - \lfloor y_i/2^5 \rfloor + 2^3}{2^4} \right\rfloor \mod 2 \end{aligned}$$

Let $r_i := 2^6 \cdot \lfloor z_i/2^6 \rfloor - z_i + 2^5 \in [-31, 32]$ be an almost-zero-centered amount lost by truncating z_i. Then

$$\begin{aligned} m'_i &= \left\lfloor \frac{2 \cdot (z_i + r_i - 2^5)/2^6 + 2 \cdot m_i - \lfloor y_i/2^5 \rfloor + 2^3}{2^4} \right\rfloor \mod 2 \\ &= \left\lfloor \frac{(z_i + r_i)/2^5 - 1 - \lfloor y_i/2^5 \rfloor + 2^3}{2^4} \right\rfloor + m_i \mod 2 \\ &= \left\lfloor \frac{(z_i + r_i)/2^5 - 1 + \lfloor (2^5 - 1 - y_i)/2^5 \rfloor + 2^3}{2^4} \right\rfloor + m_i \mod 2 \end{aligned}$$

Now the numerator is an integer, so the inner floor has no effect. Therefore

$$\begin{aligned} m'_i &= \left\lfloor \frac{(z_i + r_i)/2^5 - 1 + ((2^5 - 1 - y_i)/2^5) + 2^3}{2^4} \right\rfloor + m_i \mod 2 \\ &= \left\lfloor \frac{z_i + r_i - y_i - 1 + 2^8}{2^9} \right\rfloor + m_i \mod 2 \end{aligned}$$

This means that m'_i is the same as m_i unless

$$z_i + r_i - y_i - 1 \geq 2^8 \qquad \text{or} \qquad z_i + r_i - y_i \leq -2^8$$

as claimed.

B Proof of the bounding technique

Theorem 1 (g, k bounding technique)**.** *Let C be a probability distribution. Suppose a stateless algorithm $\mathcal{A}$ repeatedly chooses $c \leftarrow C$ from an oracle,* *then chooses to either query c or not. If $\mathcal{A}$ queries c, then the events in some collection F of events may occur. $\mathcal{A}$ repeats this process until it makes q queries. Let δ be the probability that any event in F happens on each query.*

Let $g \in \{1, 2\}$. Let w be the expected number of samples of C per query, and let $\mathrm{work} := q\sqrt[g]{w}$. *Let*

$$\delta_k := \sum_{\mathsf{f} \in F} \left[\Pr\left(\mathsf{f} : c\right) : c \leftarrow C\right]_k$$

Then

$$\Pr\left(\text{an event in } F \text{ occurs}\right) \quad \leq \quad q\delta \quad \leq \quad q^{1-g/k} \cdot \mathrm{work}^{g/k} \cdot \delta_k$$

Furthermore if $k > g$ and $\delta \geq 1/q$, then

$$q \geq (\mathrm{work}^g \cdot \delta_k^k)^{-1/(k-g)}$$

Finally, if $\mathcal{A}$ queries every sample, then $\delta = \delta_1$.

To prove this theorem, we begin with two lemmas derived from the weighted power means inequality:

Lemma 1. *Let C be a distribution over a space $\mathcal{C}$; $Q \subseteq \mathcal{C}$ a subset of its range, $g \geq 1$ be real, and f a function $\mathcal{C} \to \mathbb{R}_{\geq 0}$ be a non-negative function. Then*

$$\sqrt[g]{\Pr\left(c \in Q : c \leftarrow C\right)} \left[f(c) : c \leftarrow C, c \in Q\right] \leq \left[f(c) : c \leftarrow C\right]_g$$

Proof. By the weighted power means inequality,

$$\left[f(c) : c \leftarrow C, c \in Q\right] \leq \left[f(c) : c \leftarrow C, c \in Q\right]_g$$

Therefore

$$\begin{aligned} &\sqrt[g]{\Pr\left(c \in Q : c \leftarrow C\right)} \cdot \left[f(c) : c \leftarrow C, c \in Q\right] \\ &\qquad \leq \sqrt[g]{\Pr\left(c \in Q : c \leftarrow C\right) \left[f(c)^g : c \leftarrow C, c \in Q\right]} \\ &\qquad = \sqrt[g]{\sum_{c \in Q} \Pr\left(c \leftarrow C\right) \cdot f(c)^g} \\ &\qquad \leq \left[f(c) : c \leftarrow C, c \in Q\right]_g \end{aligned}$$

because $f(c)$ is non-negative. $\square$

Lemma 2. *Let C be a distribution on a space $\mathcal{C}$, and let $f : \mathcal{C} \to \mathbb{R}_{\geq 0}$ be a non-negative function. Suppose that $\delta := \big[f(c) : c \leftarrow C\big] > 0$. Then also for all real $j \geq 1$,*

$$\big[f(c) : c \leftarrow C\big] \geq \delta^{1-j}\Big[f(c)^j : c \leftarrow C\Big] = \delta^{1-j}\big[f(c) : c \leftarrow C\big]_j^j$$

Proof. By weighted power means,

$$\big[f(c) : c \leftarrow C\big]^j \leq \Big[f(c)^j : c \leftarrow C\Big]$$

By assumption, also

$$\big[f(c) : c \leftarrow C\big] \cdot \delta^{j-1} = \big[f(c) : c \leftarrow C\big]^j$$

Combining these gives the claimed lemma. $\square$

In the above lemmas, we set $j := k/g$ and

$$f(c) := \Pr\left(\text{an event in } F \text{ occurs when querying } c\right) \leq \sum_{\mathsf{f} \in F} \Pr\left(\mathsf{f} : c\right)$$

We also have work $= q/\sqrt[g]{\Pr\left(c \in Q : c \leftarrow C\right)}$ and $\delta = \big[f(c) : c \leftarrow C, c \in Q\big]$, so that

$$\begin{aligned}
q\delta/\text{work} &= \sqrt[g]{\Pr\left(c \in Q : c \leftarrow C\right)}\,\big[f(c) : c \leftarrow C, c \in Q\big] \\
&\leq \big[f(c) : c \leftarrow C, c \in Q\big]_g \\
&\leq \delta^{1-j} \cdot \big[f(c) : c \leftarrow C, c \in Q\big]_{gj}^j \\
&\leq \delta^{1-k/g} \cdot \big[f(c) : c \leftarrow C, c \in Q\big]_k^{k/g}
\end{aligned}$$

Rearranging,

$$\begin{aligned}
\delta^{k/g} &\leq \text{work}/q \cdot \big[f(c) : c \leftarrow C, c \in Q\big]_k^{k/g} \\
\delta &\leq \text{work}^{g/k}/q^{g/k} \cdot \big[f(c) : c \leftarrow C, c \in Q\big]_k \\
q\delta &\leq \text{work}^{g/k} \cdot q^{1-g/k} \cdot \big[f(c) : c \leftarrow C, c \in Q\big]_k
\end{aligned}$$

Setting $f(c) \leq \sum_{\mathsf{f} \in F} \Pr(\mathsf{f} : c)$ gives

$$\begin{aligned} q\delta &\leq \text{work}^{g/k} \cdot q^{1-g/k} \cdot \left[\sum_{\mathsf{f} \in F} \Pr(\mathsf{f} : c) : c \leftarrow C, c \in Q \right]_k \\ &\leq \text{work}^{g/k} \cdot q^{1-g/k} \cdot \sum_{\mathsf{f} \in F} \big[\Pr(\mathsf{f} : c) : c \leftarrow C, c \in Q \big]_k \\ &= \text{work}^{g/k} \cdot q^{1-g/k} \cdot \delta_k \end{aligned}$$

by the Minkowski inequality. For $k = g = 1$ and work = q, this expression simplifies to

$$\delta \leq \sum_{\mathsf{f} \in F} \big[\Pr(\mathsf{f} : c) : c \leftarrow C \big]$$

which is the union bound on the overall probability that any event in F occurs. On the other hand, requiring $\delta \geq 1/q$ gives

$$q^{g/k-1} \leq \text{work}^{g/k} \cdot \sum_{\mathsf{f} \in F} \delta_k$$

so that

$$\begin{aligned} q &\geq \text{work}^{g/(g-k)} \cdot \delta_k^{k/(g-k)} \\ &= \Big(\text{work}^g \cdot \delta^k \Big)^{-1/(k-g)} \end{aligned}$$

This completes the proof of Theorem 1.

C Pebbling algorithm

Theorem 2 (Pebbling algorithm)**.** *Suppose that for a $b \times b$ matrix B, $B_{i,j} \neq 0$ only when*

$$0 \leq i - j \ \mathit{mod}\ b \leq \ell$$

for some small lag ℓ. Suppose the distribution χ can take n different values in each coordinate. In this case, there is an algorithm which computes $\eta(B, k, \chi)$ numerically in time $O(b \cdot n^{3\ell})$.

Proof. We separate the x variables into four sets:

- Outer variables: x_0 thorough $x_{\ell-1}$ are treated as an outer loop: all possibilities are exhaustively considered.

- Frontier variables: $x_{a-\ell}$ through x_{a-1} are factored out of the inner expression $e^{\vec{x}^\top M \vec{y}}$. This set may overlap with the outer variables. At the beginning, it is the same as the outer variables.

- Factored variables: x_ℓ through $x_{a-\ell-1}$ have been factored out of the expression. This set starts empty.

- Inner variables: The rest of the variables, x_a through x_{b-1}, remain in the inner set.

Let $\vec{y}_{a\ldots b-1}$ be a vector which is 0 for the first a positions, and equal to y in the remaining $b-a$ positions. The pebbling algorithm manipulates the equation

$$\eta(B,k,\chi) = \left[\left[c_{OF} \cdot \left[\left[e^{\vec{x}^\top M \vec{y}_{a\ldots b-1}} : \vec{y}\right]^k : \underbrace{x_a \ldots x_{b-1}}_{\text{inner}}\right] : \underbrace{x_{a-\ell} \ldots x_{a-1}}_{\text{frontier}}\right] : \underbrace{x_0 \ldots x_{\ell-1}}_{\text{outer}}\right]$$

Here c_{OF} depends on the parameters of η, and on all the x's in the outer two expectations; its two indices OF remind that it depends in particular on the values of the Outer and Frontier variables. The Factored variables are not present in the expectation: they are captured in the c_{OF} coefficients.

Consider what happens when we increment a, looking only at the frontier and inner expectations:

$$\begin{aligned}\left[c_{OF} \cdot \left[\left[e^{\vec{x}^\top M \vec{y}_{a\ldots b-1}} : \vec{y}\right]^k : x_a \ldots x_{b-1}\right] : x_{a-\ell} \ldots x_{a-1}\right] \\ = \left[c_{OF} \cdot \left[\left[e^{\vec{x}^\top M \vec{y}_{a\ldots b-1}} : \vec{y}\right]^k : x_{a+1} \ldots x_{b-1}\right] : x_{a-\ell} \ldots x_a\right]\end{aligned}$$

Now

$$e^{\vec{x}^\top M \vec{y}_{a\ldots b-1}} = e^{\vec{x}^\top M \vec{y}_{a+1\ldots b-1}} \cdot e^{\vec{x}^\top M_a y_a}$$

Furthermore, in terms of y's components these are independent of each other, so that

$$\left[e^{\vec{x}^\top M \vec{y}_{a\ldots b-1}} : \vec{y}_{a\ldots b-1}\right]^k = \left[e^{\vec{x}^\top M \vec{y}_{a+1\ldots b-1}} : \vec{y}_{a+1\ldots b-1}\right]^k \cdot \left[e^{\vec{x}^\top M_a y_a} : y_a\right]^k$$

By definition of ℓ, all the variables x_i that appear in $e^{\vec{x}^\top M_a y_a}$ are in the set $\{x_{a-\ell}\ldots x_a\}$, so they are determined by the frontier variables plus x_a. Therefore $\left[e^{\vec{x}^\top M_a y_a} : y_a\right]$ is easily computed based on the distribution χ. Except in this expression, the outermost frontier variable $x_{a-\ell}$ no longer appears with nonzero coefficient in $\vec{x}^\top M \vec{y}_{a+1\ldots b-1}$ (unless it's an outer variable), so it moves to the factored set. This gives us the update expression

$$c_{O\ (x_{a-\ell+1}\ldots x_a)} = \Pr\left(x_a \leftarrow \chi\right) \cdot \sum_{x_{a-\ell}} c_{O\ (x_{a-\ell}\ldots x_{a-1})} \cdot \left[e^{\vec{x}^\top M_a y_a} : y_a\right]^k$$

where the expectation is well-defined and easily computed because it depends only on $x_{a-\ell}$ through x_a. The state of the algorithm is therefore the current position a, and the matrix of coefficients c_{OF}. Since the update matrix depends only on the column M_a of M, and those columns tend to repeat, when implementing this we also keep a cache of recently used update matrices.

After b steps of this update procedure, the frontier has reached the end of the block, wrapped around and returned to be equal to the outer variables, so all variables are either outer or factored and the inner expression is $e^{\vec{x}^\top M \vec{0}} = 1$. At that point,

$$\eta(B, k, \chi) = \sum_O c_{OO}$$

is just the trace of the state matrix. The final answer $\eta(M, k, \chi)$ can be found as a product of $\prod_B \eta(B, k, \chi)$ where B ranges over the blocks in the matrix. □

D Proof of the Gaussian heuristic

Theorem 3 (Gaussian heuristic)**.** *If χ is the Gaussian distribution with mean 0 and variance v, and if the matrix $I - kv^2 M^\top M$ is positive-definite,* *then*

$$\eta(M,k,\chi) = \frac{1}{\sqrt{\det(I - kv^2 M^\top M)}}$$

Proof. Consider a singular-value decomposition $M = S^*\Lambda T$, where S, T are unitary. Then the distributions $\vec{x}^\top M\vec{y} = \vec{x}^\top S^*\Lambda T$ and $\vec{x}^\top \Lambda\vec{y}$ have the same distribution, because multi-variate i.i.d. Gaussians are spherically symmetric. This means that when χ is Gaussian with variance v:

$$\eta(M,k,\chi) = \prod_\lambda \frac{1}{\sqrt{2\pi v}} \int_{-\infty}^{\infty} \left(\frac{1}{\sqrt{2\pi v}} \int_{-\infty}^{\infty} e^{\lambda x_i y_i - \frac{x^2}{2v}}\ dx \right)^k \cdot e^{\frac{y^2}{2v}}\ dy$$

where λ iterates over the singular values of M. We calculate that for all real α,

$$\frac{1}{\sqrt{2\pi v}} \int_{-\infty}^{\infty} e^{\alpha x - \frac{x^2}{2v}}\ dx = \frac{1}{\sqrt{2\pi v}} \int_{-\infty}^{\infty} e^{\frac{(x-v\alpha)^2}{2v} - \frac{v\alpha^2}{2}}\ dx = e^{-\frac{v\alpha^2}{2}}$$

so that

$$\begin{aligned} \frac{1}{\sqrt{2\pi v}} \int_{-\infty}^{\infty} \left(\frac{1}{\sqrt{2\pi v}} \int_{-\infty}^{\infty} e^{\lambda x_i y_i - \frac{x^2}{2v}}\ dx \right)^k \cdot e^{\frac{y^2}{2v}}\ dy &= \frac{1}{\sqrt{2\pi v}} \int_{-\infty}^{\infty} e^{\frac{k(v^2\lambda y_i)^2}{2v} - \frac{y^2}{2v}}\ dy \\ &= \frac{1}{\sqrt{1 - kv^2\lambda^2}} \end{aligned}$$

if the radicand is positive, and divergent if the radicand is negative. Now as long as $I - kv^2 M^\top M$ is positive-definite, all radicands are positive and

$$\eta(M,k,\chi) = \prod_\lambda \frac{1}{\sqrt{1 - kv^2\lambda^2}} = \frac{1}{\sqrt{\det(I - kv^2 M^\top M)}}$$

as claimed.

To implement this numerically, we used Cholesky decomposition to determine both positive-definiteness and the determinant. □