Visualizing size-security tradeoffs for lattice-based encryption

Daniel J. Bernstein[1,2]

[1] Department of Computer Science, University of Illinois at Chicago, Chicago, IL 60607–7045, USA
[2] Horst Görtz Institute for IT Security, Ruhr University Bochum, Germany
djb@cr.yp.to

Abstract. There are many proposed lattice-based encryption systems. How do these systems compare in the security that they provide against known attacks, under various limits on communication volume? There are several reasons to be skeptical of graphs that claim to answer this question. Part of the problem is with the underlying data points, and part of the problem is with how the data points are converted into graphs.

1 Introduction

NIST's Post-Quantum Cryptography Standardization Project has selected 26 proposals for round 2. This paper focuses on the 8 proposals for lattice-based encryption—Frodo, Kyber, LAC, NewHope, NTRU, NTRU Prime, Round5, and Saber—plus ThreeBears, which is sometimes counted as a lattice proposal.

It is not meaningful to ask which proposal provides the maximum security against known attacks. Any claimed answer can be disproven as follows: take another proposal and scale that proposal up to provide a higher security level. It is, however, meaningful to impose a cost limit—e.g., a limit on ciphertext size—and then ask how the proposals *within this cost limit* compare in the security that they provide against known attacks.

Graphs comparing size-security tradeoffs for various lattice proposals began appearing during round 1. Figure 1.1 is an illustrative example of a similar graph for round 2. The graph has 11 curves rather than 9 curves, since NTRU Prime has two different options (`sntrup` for Streamlined NTRU Prime and `ntrulpr` for NTRU LPRime) and Round5 also has two different options (`round5nd` and `round5n1`) beyond the basic size parameters. There is some zig-zagging in the graph, suggesting that it might also make sense to split `ntru` into `ntruhps` and `ntruhrss`, and to split `round5nd` into `round5nd0d` and `round5nd5d`.

This work was supported by the National Institute of Standards and Technology under grant 60NANB12D261, by DFG Cluster of Excellence 2092 "CASA: Cyber Security in the Age of Large-Scale Adversaries", by the Cisco University Research Program, and by the National Science Foundation under grant 1314919. "Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation" (or other funding agencies). Permanent ID of this document: `da0f0331c34c346771e3d0d57e083677f54892a0`. Date: 2019.06.03.

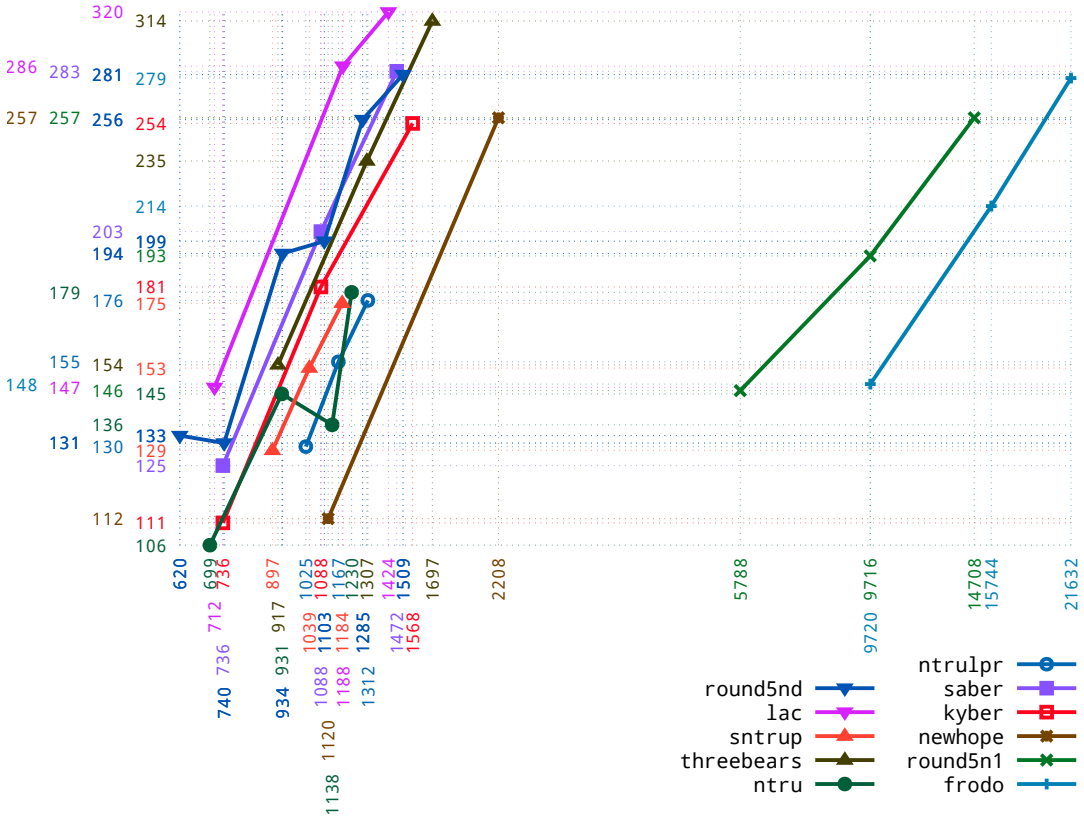

Fig. 1.1. Estimated security (vertical axis, log scale) vs. ciphertext size (horizontal axis, bytes, log scale). See Section 2 for problems with the data points, and Section 3 for problems with the graphing.

The outliers on the right side of the graph are `frodo` and `round5n1`. These outliers deliberately avoid the polynomial structures used in the other proposals. Polynomial structures seem to be essential for top performance, but perhaps the same structures allow devastating attacks.

There is a great deal more that can be said about the much smaller differences on the left side of the graph. The graph seems to support the intuition that there are measurable increases in ciphertext size from

- perfect correctness rather than decryption failures (with various levels of error correction);
- Noisy NTRU ("RLWE" etc.) rather than Rounded NTRU ("RLWR" etc.);
- Product NTRU ("LPR" etc.) rather than Quotient NTRU; and
- large Galois groups rather than small Galois groups.

On the other hand, there are arguments that some of these increases (1) are too small to matter to users and (2) decrease security risks.

The central topic of debate is the following. Should we push for an improved tradeoff between size and security against *known* attacks, if this improvement means increasing the risks from *unknown* attacks?

This paper focuses on part of the underlying information, the part that the graph is meant to show: the tradeoff between size and security against known attacks. The risks from unknown attacks are outside the scope of this paper.

The main objective of this paper is to highlight ways that graphs of this type can *fail* to show this tradeoff. Part of the problem (Section 2) is with the underlying data being graphed. Another part of the problem (Section 3) is with how the data points are presented as a graph. This includes a very common graphing mistake that often reverses comparisons between proposals. This mistake is encouraged by typical graphing tools and takes effort to correct.

2 The underlying data

This section highlights some important questions that readers should ask about the (cost, security) data points provided for lattice proposals.

2.1. Does "security" accurately report security against known attacks? In general, the answer to this type of question is "no". Computing the cost of known attacks against lattice systems (never mind the difficulty of predicting the impact of unknown attacks) is a bleeding-edge research problem. Existing estimates of this cost combine various calculations that are loosely inspired by known attacks but that include quite a few oversimplifications and potential oversimplifications. The resulting errors can lead to a variety of problems:

- Users thinking that they have more security against known attacks than they actually do.
- Users thinking that they have less security than they actually do, and then compensating by moving to bigger systems, damaging deployment.
- Errors affecting one proposal more than another, perhaps even reversing comparisons between proposals. For example, typical estimates omit known "hybrid attacks"; this omission has more effect on some proposals than on others. As another example, typical estimates overstate the extent to which known attacks can exploit "rotations" against `sntrup`.

The sample graphs in this paper use the estimate from the literature for which data points are most readily available: a pre-quantum estimate often called "Core-SVP".[3] All of the data points in these graphs are taken from the round-2 submission documents.[4] See Table 2.2.

I do not endorse the "Core-SVP" estimate. On the contrary, the state of the art (1) has moved beyond this estimate and (2) still has problems. See [5,

[3] The same "Core-SVP" name is also used for an estimate of post-quantum security. Post-quantum "Core-SVP" is almost as popular as pre-quantum "Core-SVP", and generally produces numbers about 10% smaller. The objections to post-quantum "Core-SVP" are a strict superset of the objections to pre-quantum "Core-SVP": for example, post-quantum "Core-SVP" relies on low-cost "QRAM".

[4] It is of course possible that the submissions did not correctly calculate "Core-SVP", or that they were actually trying to calculate something else, or that I copied some numbers incorrectly.

system	parameter set	size	sec	source for size; source for sec
frodo	640	9720	148	[2, p24; p38, “classical”]
frodo	976	15744	214	[2, p24; p38, “classical”]
frodo	1344	21632	279	[2, p24; p38, “classical”]
kyber	512	736	111	[3, p18, “ct”; p21, Table 3, “classical”]
kyber	768	1088	181	[3, p18, “ct”; p21, Table 3, “classical”]
kyber	1024	1568	254	[3, p18, “ct”; p21, Table 3, “classical”]
lac	128	712	147	[16, p13; p14, “classic”]
lac	192	1188	286	[16, p13; p14, “classic”]
lac	256	1424	320	[16, p13; p14, “classic”]
newhope	512	1120	112	[1, p20; p32, “known classical”]
newhope	1024	2208	257	[1, p20; p32, “known classical”]
ntru	hps2048509	699	106	[6, p29, “ct”; p35, Table 5, “non-local”]
ntru	hps2048677	931	145	[6, p29, “ct”; p35, Table 5, “non-local”]
ntru	hps4096821	1230	179	[6, p29, “ct”; p35, Table 5, “non-local”]
ntru	hrss701	1138	136	[6, p29, “ct”; p35, Table 5, “non-local”]
ntrulpr	653	1025	130	[5, p32; p65, “pre-quantum ignoring hybrid”]
ntrulpr	761	1167	155	[5, p32; p65, “pre-quantum ignoring hybrid”]
ntrulpr	857	1312	176	[5, p32; p65, “pre-quantum ignoring hybrid”]
round5n1	1	5788	146	[4, p57; p57, “classical primal/dual”]
round5n1	3	9716	193	[4, p57; p57, “classical primal/dual”]
round5n1	5	14708	257	[4, p57; p57, “classical primal/dual”]
round5nd	1.0d	740	131	[4, p55; p55, “classical primal/dual”]
round5nd	1.5d	620	133	[4, p56; p56, “classical primal/dual”]
round5nd	3.0d	1103	199	[4, p55; p55, “classical primal/dual”]
round5nd	3.5d	934	194	[4, p56; p56, “classical primal/dual”]
round5nd	5.0d	1509	281	[4, p55; p55, “classical primal/dual”]
round5nd	5.5d	1285	256	[4, p56; p56, “classical primal/dual”]
saber	light	736	125	[7, p9; p9, “classical”]
saber	main	1088	203	[7, p9; p9, “classical”]
saber	fire	1472	283	[7, p9; p9, “classical”]
sntrup	653	897	129	[5, p32; p65, “pre-quantum ignoring hybrid”]
sntrup	761	1039	153	[5, p32; p65, “pre-quantum ignoring hybrid”]
sntrup	857	1184	175	[5, p32; p65, “pre-quantum ignoring hybrid”]
threebears	baby	917	154	[9, p45, “capsule”; p39, “classical”]
threebears	mama	1307	235	[9, p45, “capsule”; p39, “classical”]
threebears	papa	1697	314	[9, p45, “capsule”; p39, “classical”]

Table 2.2. Data points for the sample graphs. (For Round5, the cited source is for a PKE that includes a 16-byte authenticator, so the numbers here are 16 bytes smaller.) See text for problems with the data points.

Section 6] for a survey of problems with “Core-SVP” and for some work towards improved estimates.

2.3. Does the “cost” number accurately report cost in applications? The horizontal axis in these sample graphs is ciphertext size. This is much easier

to measure than security, but is it the right thing to measure? One can argue that it is better to use other cost metrics.

I have chosen to focus on size metrics for these graphs because my assessment is that communication volume is by far the most serious performance problem with lattice systems. Here is some evidence supporting this assessment:

- Google reported [14] that sending an extra 1100 bytes in a TLS handshake usually added more than 2 milliseconds of latency, and often (5% of the time) added 20 milliseconds of latency. Lattice computations are much faster than this on typical CPUs. Presumably this situation will persist: computation is getting cheaper more quickly than communication is.
- Google also reported that sending 10000 extra bytes in a TLS handshake made an unacceptably large number of sites fail, including "common sites such as `godaddy.com`, `linkedin.com`, and `python.org`". The big picture is that many deployment strategies for lattice-based cryptography rely on opportunistically sending data through existing frameworks to sites that *might* have been upgraded. This causes serious problems when the data hits preexisting size limits, making the non-upgraded sites fail.
- There is also a new wave of TLS replacements built upon UDP, such as QUIC. These replacements generally rely on fitting cryptographic objects into single UDP packets; see, e.g., [11, Section 7]. Being forced to split data across multiple packets would lose simplicity and performance, and would make denial of service easier. The time spent for lattice computations is far less important.

This leaves the question of exactly which size metric to choose. This choice can affect comparisons between systems. For example, `sntrup` has smaller ciphertext sizes than `ntrulpr`, but `ntrulpr` has smaller public-key sizes than `sntrup`.

Google's experiment [13] with NewHope in TLS relied on single-use keys. If the experiment had been continued and expanded then, in the long run, each TLS session would have involved communicating one NewHope public key and one NewHope ciphertext. This example suggests that the right cost metric is ciphertext size *plus* public-key size.[5]

However, public keys can be used many times. This reuse allows the cost of communicating public keys to be reduced via standard networking techniques, such as multicasts and caching, as emphasized by McGrew in [17]. This reuse requires IND-CCA2 security, but IND-CCA2 security has only minor costs.

One consequence of this reusability is that *long-term* public keys can be used for authentication. For example, a client begins a session by encapsulating a secret key to the server's long-term public key, and checking that the server's response is authenticated using this secret key. This is generally more efficient than using public-key signatures. Note that communicating a separate ciphertext from each client to the server cannot take advantage of multicasts and caching.

[5] An example of a graph using this size metric is [6, Figure 11]. On the other hand, this graph does not compare different proposals; it compares different parameter sizes for one proposal. Changing size metric would have little effect on the graph.

Another consequence is that short-term public keys—for example, keys erased within 5 minutes for "forward secrecy"—can still be used multiple times. Clients can share the server's latest short-term public key through fast local networks, and then the server can simply send a 32-byte hash to authenticate the key. Each client still sends a ciphertext to the server's short-term public key, and another ciphertext to the server's long-term public key for authentication as above.

Of course, public keys still have to be communicated sometimes, but it would not be surprising if optimized lattice-based protocols can reduce the cost of communicating public keys by at least an order of magnitude compared to the cost of communicating ciphertexts. The public-key size for lattice proposals is on the same scale as the ciphertext size,[6] and thus becomes unimportant if key caching is even moderately effective. Measuring the sum of ciphertext size and public-key size for lattice proposals is then much farther away from reality than simply measuring ciphertext size.

3 Visualizing and comparing the data

This section highlights some important questions that readers should ask about how the data points provided for lattice proposals are converted into graphs.

The examples in this section imagine that the "Core-SVP" estimate on the vertical axis is accurately measuring security. This is not a defensible assumption (see Section 2), but the broader principles illustrated by these examples will also apply if "Core-SVP" is replaced by something more accurate.

3.1. What happens between the data points? The question that began this paper is how lattice proposals compare in security *within a limit on cost*. What Figure 1.1 communicates to the reader is many wrong answers to this question.

As a concrete example, let's compare `threebears` to `sntrup` for an application that needs to fit a ciphertext into a 1280-byte UDP-over-IPv6 packet.[7] There are 40 bytes of IPv6 overhead and 8 bytes of UDP overhead, so the ciphertext must fit into 1232 bytes. This limits `threebears` to security level 154 (with a 917-byte ciphertext), while `sntrup` can reach security level 175 (with an 1184-byte ciphertext).

Because `threebears` can reach security level 235 with a 1307-byte ciphertext, Figure 1.1 draws a line from $(917, 154)$ to $(1307, 235)$. This line includes, for example, the point $(1232, 219.42\ldots)$. But `threebears` does *not* reach security level 219 with a 1232-byte ciphertext; it reaches only a much lower security level, namely 154. Except for the endpoints, this entire line is fake.

More broadly, the `threebears` lines in Figure 1.1 are clearly above the `sntrup` lines. This communicates the following notion to the reader: `threebears` provides

[6] The situation is different for, e.g., a code-based system with a 128-byte ciphertext and a 261120-byte key. Round5 also specifies a "smallCT" parameter set for "specific use-cases", with a 972-byte ciphertext and a 163536-byte public key.

[7] IPv6 does not guarantee successful delivery of larger packets. See [5, Section 7.2] for further information and references. Packet-size limits are also considered in, e.g., [4, Section 2.10].

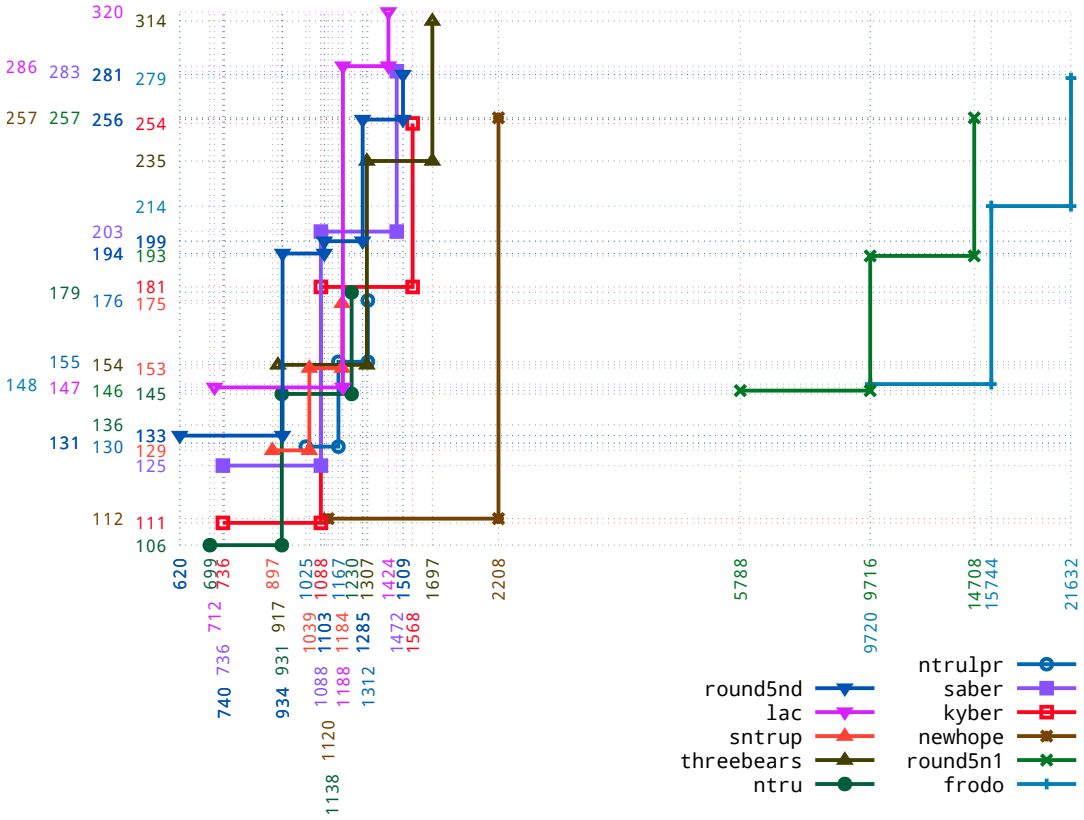

Fig. 3.2. Estimated security (vertical axis, log scale) vs. ciphertext size (horizontal axis, bytes, log scale). This corrects the fake lines from Figure 1.1. See Section 2 for problems with the data points.

better security than `sntrup` at every ciphertext size. But this notion is not true. The reality is more complicated, with `sntrup` providing better security for some ciphertext sizes and `threebears` providing better security for other ciphertext sizes.

Specifically, it's true that the $(917, 154)$ for `threebears` is better than `sntrup`'s $(1039, 153)$, which has a ciphertext size $1.13\times$ larger and security $1.01\times$ smaller. However, if the application's size limits allow an 1184-byte ciphertext, then `sntrup` jumps past `threebears` to security level 175. The `threebears` proposal cannot compete with this security level unless the application's ciphertext limit is as large as 1307 bytes.[8]

Even more broadly, `sntrup` provides

- better security than `kyber` for ciphertext limits between 897 bytes and 1087 bytes;
- better security than `lac` for ciphertext limits between 1039 bytes and 1187 bytes;

[8] Perhaps an even larger limit would return the lead to `sntrup` if the parameter ranges for both proposals were extended.

- better security than `ntru` for ciphertext limits between 897 bytes and 930 bytes, and for ciphertext limits between 1039 bytes and 1229 bytes;
- better security than `saber` for ciphertext limits between 897 bytes and 1087 bytes; and
- better security than `threebears` for ciphertext limits between 897 bytes and 916 bytes, and for ciphertext limits between 1184 bytes and 1306 bytes.

These are interesting sizes for applications that want to fit ciphertexts along with other data into Internet packets. The $(1120, 112)$ for `newhope` is not competitive, and `frodo` is not an option. The only proposal that consistently provides better tradeoffs than `sntrup` is `round5nd`.

For comparison, the fake lines in Figure 1.1 incorrectly tell the reader that `kyber`, `lac`, `saber`, and `threebears` consistently provide higher security than `sntrup`. Similar comments apply to comparisons of proposal P to proposal Q for various other pairs (P, Q). Correcting all of the fake lines in Figure 1.1 produces Figure 3.2, which accurately shows the complicated leapfrogging of security levels as the ciphertext limit increases.

It is important for graphs to not only omit the fake lines but also include the correct lines. Drawing merely the dots (as in Figure 3.3), without the fake lines but also without the correct lines, leaves the human eye free to imagine the fake lines, and leads people to incorrect conclusions. One of the sources of dot-only graphs claimed that "use cases with sharp bandwidth constraints" are "likely to be favorable" to "ThreeBears and Saber and especially LAC", since these are "inherently more bandwidth-efficient than much of the field". In fact, the small bandwidth differences shown on the left side of Figure 1.1 are often outweighed by the correction from Figure 1.1 to Figure 3.2, as shown by the many crossings in Figure 3.2. Figure 3.3 does not make this clear.

The correct lines have the same relevance to applications that ask for minimum ciphertext size subject to a required level of (estimated) security. One then reads the graph from left to right at the specified vertical position, rather than from top to bottom at a specified horizontal position. For example, if the security level is required to be at least 128, then `sntrup` provides 897-byte ciphertexts, whereas `threebears`—claimed to be more bandwidth-efficient—cannot do better than 917 bytes. As another example, if the security level is required to be at least 160, then `sntrup` provides 1184-byte ciphertexts, whereas `threebears` cannot do better than 1307 bytes. Of course, there are other levels where `threebears` does better than `sntrup`.

I chose to list the proposals in these sample graphs in increasing order of ciphertext size, subject to the (estimated) security level being at least 128. (There is a tie between `saber` and `kyber` for positions 7/11 and 8/11; security level at this ciphertext size is used as a tiebreaker.) The order is sensitive to changes in the security cutoff: for example, `sntrup` drops from position 3/11 to position 5/11 if 128 is increased to 144, but then moves back up to position 4/11 if 144 is increased to 160. The order is also sensitive to small tweaks in the security estimates.

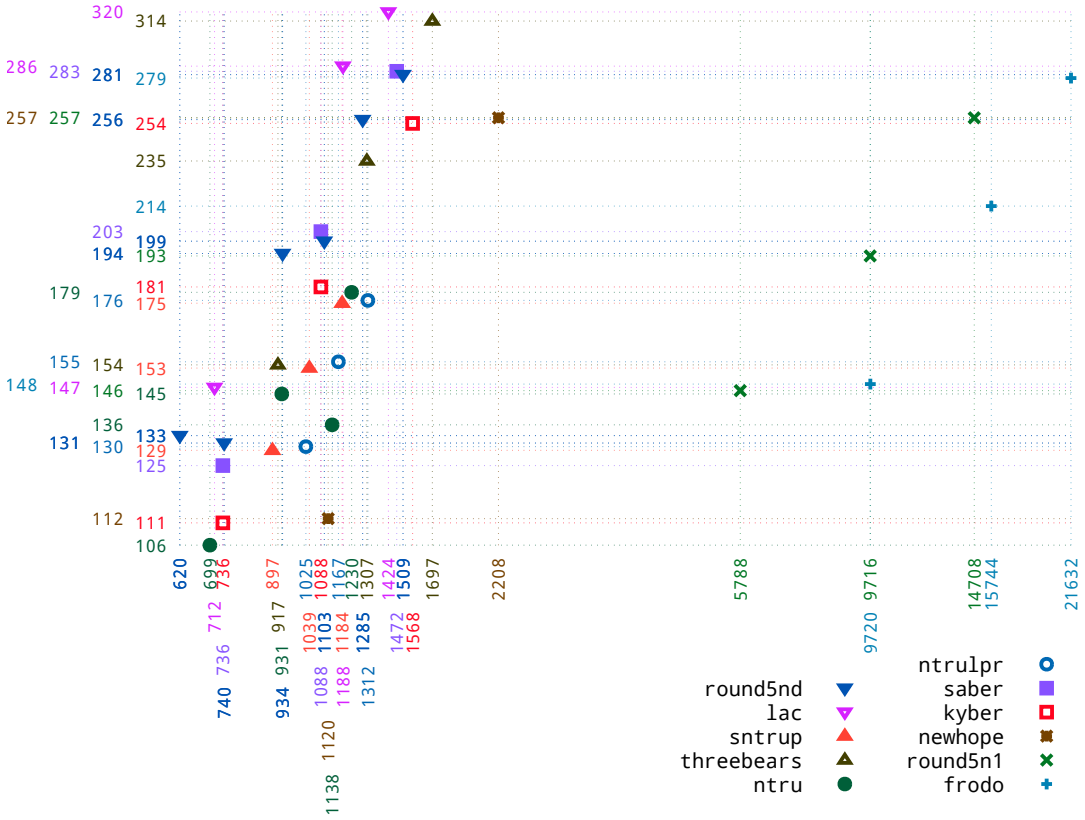

Fig. 3.3. Estimated security (vertical axis, log scale) vs. ciphertext size (horizontal axis, bytes, log scale). This does not include the correct lines from Figure 3.2 (except as hinted by the grid lines), so the human eye tends to connect the dots, filling in the fake lines from Figure 1.1. Consider, e.g., the `frodo` dots on the right. See also Section 2 for problems with the data points.

One could also choose, e.g., a ciphertext limit of 1200 bytes (1232 bytes as above, minus 32 bytes for other useful data such as an ECC key), and put proposals in decreasing order of security level subject to this limit (followed by proposals that cannot reach this limit). Again the order is sensitive to changes in the cutoff: e.g., `sntrup` jumps from position 5/11 up to position 3/11 if the 1200-byte limit is decreased by 128 bytes to 1072 bytes.

Readers not familiar with these post-quantum proposals might think that I'm simply hyping the gaps between the parameter sets that the proposals chose to provide—the proposals will respond by adding intermediate parameter sets, and adding these new parameter sets to Figure 3.2 will produce something much more like Figure 1.1. However, for many of the proposals, the parameter space is quite sparse, and incorporating intermediate options would spoil major features claimed by the submissions. For example, the Kyber submission

- uses short vectors over the large ring $(\mathbf{Z}/3329)[x]/(x^{256}+1)$;
- includes fast NTT software for this particular ring; and
- emphasizes that it reuses this software "*for all parameter sets*".

See [3, Section 6.1, "Advantages", first "unique advantage"; italics in original text]. The three points on the `kyber` curve use vectors of lengths 2, 3, and 4. The parameter space requires the vector length to be an integer.[9] Saber similarly uses vectors of polynomials modulo $x^{256}+1$, and emphasizes that changing security levels is "without any need to change the underlying arithmetic". LAC and NewHope similarly use $x^{512}+1$ and $x^{1024}+1$.

The situation is different for `ntru`, `ntrulpr`, `round5nd`, and `sntrup`. These proposals follow the NTRU tradition of allowing many closely spaced degrees of polynomial moduli, rather than limiting attention to power-of-2 cyclotomics. These proposals can easily add intermediate parameter sets, further improving their position in Figure 3.2.

3.4. How are the axes scaled? A size cutoff in an application is, presumably, much less likely to be within (e.g.) the 1.10× gap from 14288 bytes to 15744 bytes than the 1.31× gap from 897 bytes to 1184 bytes. However, the first gap occupies a horizontal distance 5× larger than the second gap if size is shown on a linear scale. In other words, a linear scale draws the reader's attention to a less important gap, while downplaying a more important gap. Readers often use their perception of such gaps in diagrams to draw unquantified conclusions about advantages being "small" or "large".

The sample graphs instead use log scale horizontally, so a constant horizontal distance is a constant factor in size.[10] Similarly, the sample graphs use log scale vertically, so a constant vertical distance is a constant factor in (estimated) security level.

There is still something artificial in how the scales are chosen in Figures 1.1 and 3.2: the graph is designed to fit all of the data points into a typical screen size, somewhat wider than tall. The presence of `frodo` and `round5n1` on the right side of the picture thus squeezes the size gaps between various systems on the left side of the picture.

Figure 3.5 instead chooses scales so that the vertical distance for a 2× change in security level matches the horizontal distance for a 2× change in size. It also zooms in on the left side of the graph; this zooming makes the differences more visible. It also increases the size of each dot.

3.6. How are colors chosen? I drew all of these sample graphs with `gnuplot`. If I had used the default choices of colors in `gnuplot` then I would have ended up with Figure 3.7 instead of Figure 3.5. In Figure 3.7, the `saber` color is a light yellow very close to the background white, while the `threebears` color has

[9] One can slightly improve the points by changing the "d" parameters, but this would spoil various claims in the submission regarding failure probabilities.

[10] One can object that if the application cost metric is actually (e.g.) the size of a ciphertext plus 1 megabyte of data then differences in small ciphertext sizes are unimportant, while log scale makes them look important. A reasonable answer is to draw another graph for this application, where the horizontal axis is ciphertext size plus 1 megabyte.

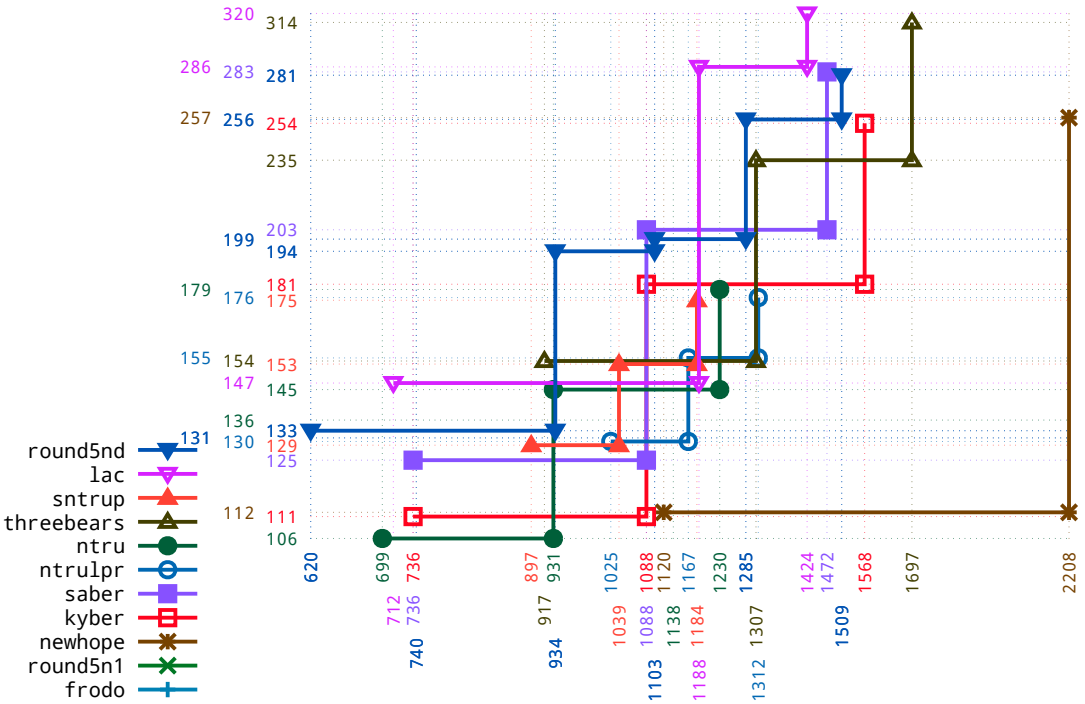

Fig. 3.5. Estimated security (vertical axis, log scale) vs. ciphertext size (horizontal axis, bytes, log scale). This adds equal scaling to Figure 3.2. See Section 2 for problems with the data points.

much more contrast. Such variations can and should trigger complaints about the colors emphasizing some proposals more than others.[11]

For the other figures, I selected colors within a restricted range of lightness, between lightness 55% and lightness 70% on the standard "Lab" color scale. Within this constraint, I chose each color to be as highly "saturated" (colorful) as possible. To choose the "hue" (red, blue, etc.), I rotated the direction in (a, b) space by $(1 + \sqrt{5})\pi/2$ for each new color, so that nearby colors in the list would have noticeably different hues. I copied Lab-to-RGB conversion code from [18].

It would be better to move a constant distance in (a, b) space, rather than rotating by a constant angle; this is different since the radius of (a, b) space (for any nontrivial lightness) varies with the direction. Another improvement would be to have nearby points in the graph push colors apart, as in [8]. It might also be helpful to allow some variations in saturation.

There are other ways to help the eye follow the lines if colors cannot be separated enough. For example, Figure 3.8 adds small notches to the lines, with a

[11] Of course, dynamically emphasizing some proposals is warranted if a user has selected those proposals for comparison. Furthermore, graphs should be optimized differently for users with protanopia, deuteranopia, etc.

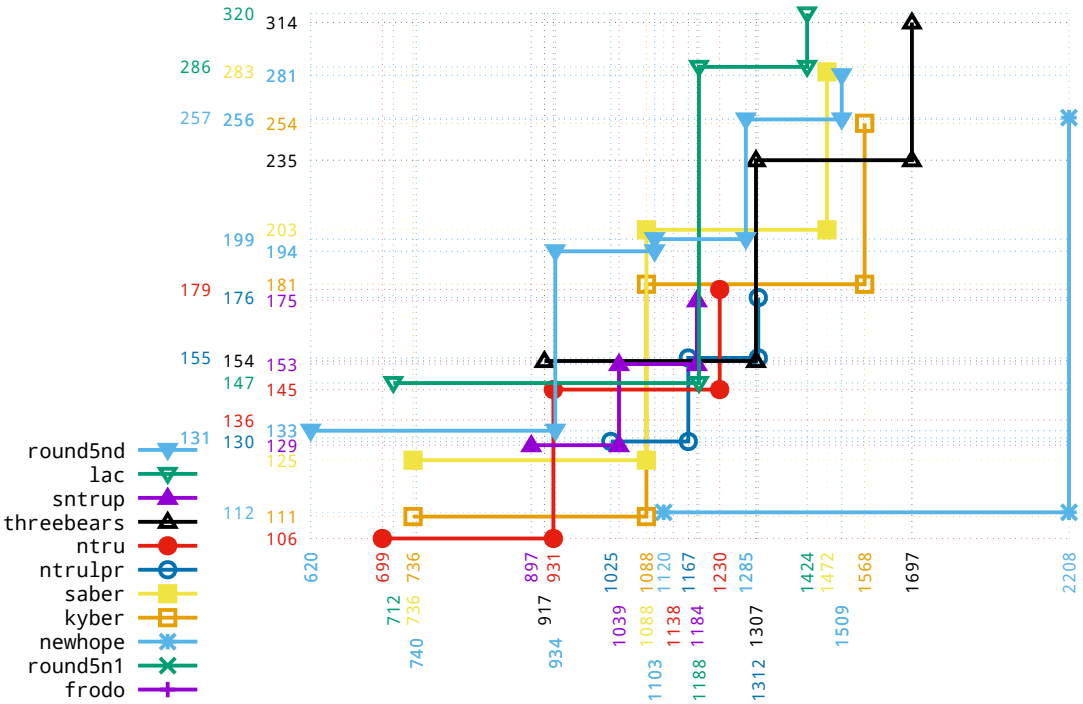

Fig. 3.7. Estimated security (vertical axis, log scale) vs. ciphertext size (horizontal axis, bytes, log scale). This is based on Figure 3.5 but with an arguably unfair variation in color lightness. See also Section 2 for problems with the data points.

different pattern of notches (at the same density) for each system. Figure 3.9 goes beyond this by using stripes (lightness 90) rather than blank notches, although further work will be required to show stripes on the labels. For comparison, Figure 3.10 shows the effect of cycling through `gnuplot`'s built-in "dash types": some lines have very high density and leap out from the picture, while others have very low density and are easy to overlook.

3.11. Are these issues new? There are many previous publications analyzing the impact of variations in graphing techniques. See, e.g., Tufte's classic books, such as "The visual display of quantitative information" [19]. My impression is that there is wide awareness of the value of scaling axes jointly in log scale, separating hues used for labeling, and avoiding excessive variations in lightness.

However, it seems standard to draw "Pareto fronts" ("Pareto frontiers") with fake diagonal lines. These fake lines

- communicate incorrect information to the reader;
- have the bizarre consequence that adding an improved Pareto-optimal point can produce a visible *retreat* in the graphed front, whereas one would expect an improved input to produce an improved front; and

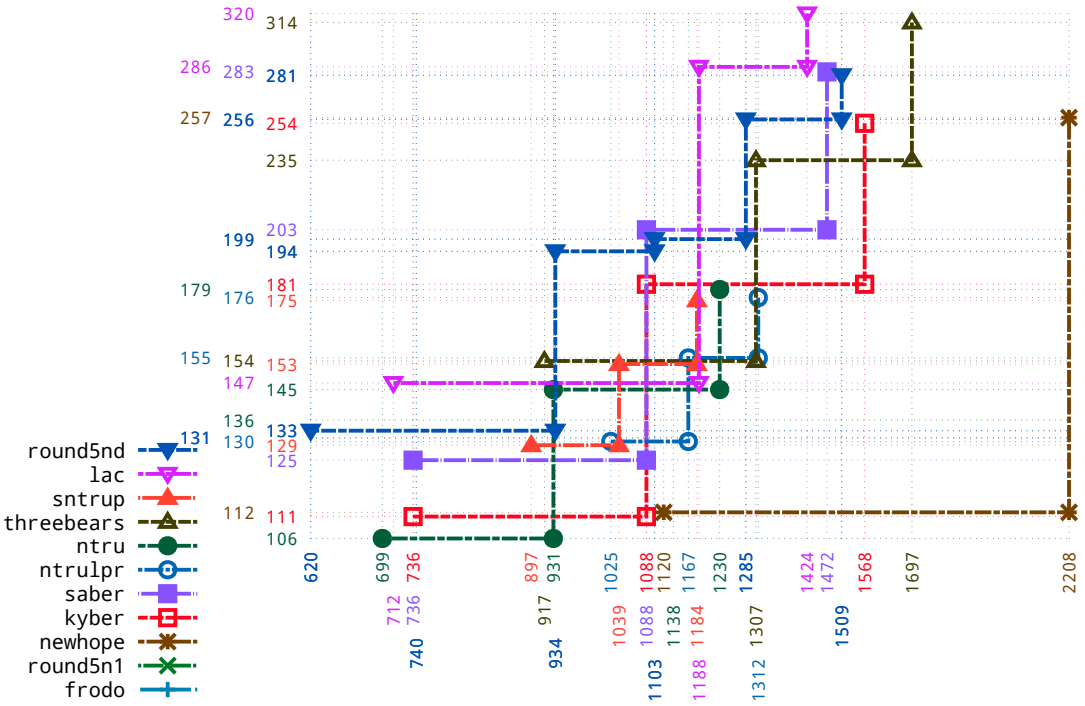

Fig. 3.8. Estimated security (vertical axis, log scale) vs. ciphertext size (horizontal axis, bytes, log scale). This adds notches to Figure 3.5. See Section 2 for problems with the data points.

- can reverse comparisons between alternatives, as illustrated by the examples highlighted in this paper.

The literature on scientific decision-making processes ("operations research") contains mathematical definitions of Pareto-front graphs that include the same fake lines, and generalizations of these definitions to more dimensions; see, e.g., [10]. The fact that the fake lines communicate incorrect information was pointed out in 2013 by Lu and Anderson-Cook [15], who advocated instead drawing horizontal and vertical lines. I have not found earlier literature pointing out this issue. I also have not found literature pointing out that the fake lines can damage comparison of alternatives ("concept selection"), and I have not found literature pointing out that omitting all the lines tends to produce the same damage since humans naturally visualize the fake lines.

References

[1] Erdem Alkim, Roberto Avanzi, Joppe Bos, Leo Ducas, Antonio de la Piedra, Thomas Poppelmann, Peter Schwabe, Douglas Stebila, Martin R. Albrecht,

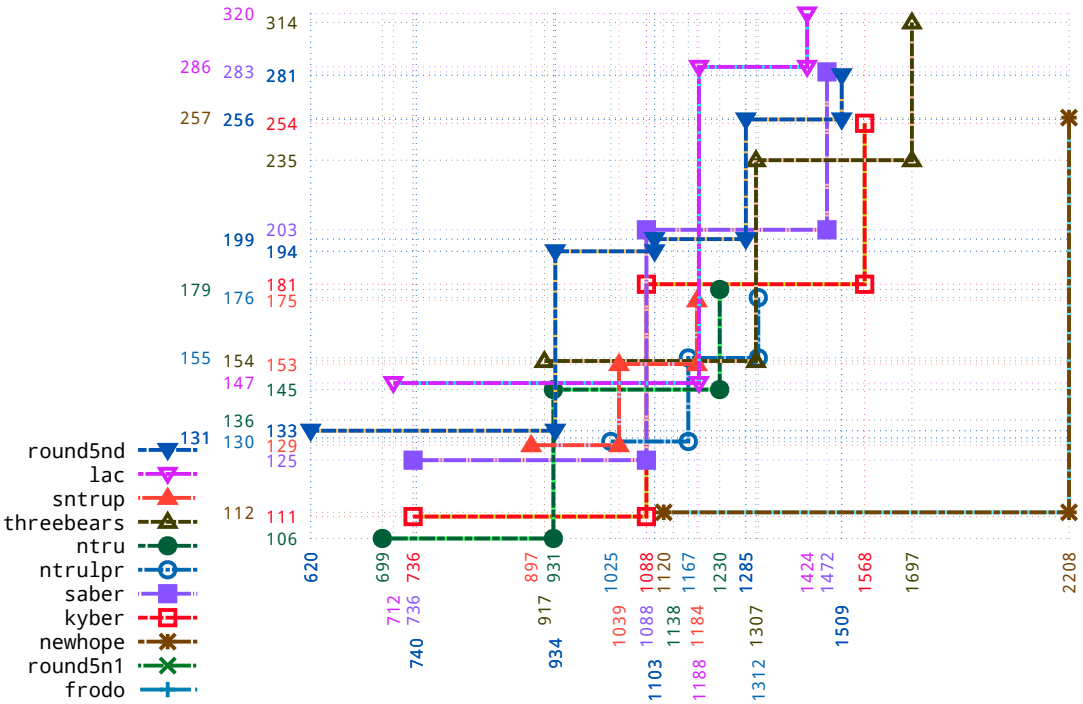

Fig. 3.9. Estimated security (vertical axis, log scale) vs. ciphertext size (horizontal axis, bytes, log scale). This adds stripes to Figure 3.8. See Section 2 for problems with the data points.

Emmanuela Orsini, Valery Osheter, Kenneth G. Paterson, Guy Peer, Nigel P. Smart, *NewHope: algorithm specifications and supporting documentation* (2019). URL: https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. Citations in this document: §2.1, §2.1.

[2] Erdem Alkim, Joppe Bos, Leo Ducas, Patrick Longa, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Christopher Peikert, Ananth Raghunathan, Douglas Stebila, *FrodoKEM: Learning With Errors key encapsulation* (2019). URL: https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. Citations in this document: §2.1, §2.1, §2.1.

[3] Roberto Avanzi, Joppe Bos, Leo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, Damien Stehlé, *CRYSTALS-Kyber: algorithm specifications and supporting documentation* (2019). URL: https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. Citations in this document: §2.1, §2.1, §2.1, §3.1.

[4] Hayo Baan, Sauvik Bhattacharya, Scott Fluhrer, Oscar Garcia-Morchon, Thijs Laarhoven, Rachel Player, Ronald Rietman, Markku-Juhani O. Saarinen, Ludo Tolhuizen, Jose-Luis Torre-Arce, Zhenfei Zhang, *Round5: KEM and PKE based on (Ring) Learning With Rounding* (2019). URL: https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. Citations in this document: §2.1, §2.1, §2.1, §2.1, §2.1, §2.1, §2.1, §2.1, §2.1, §7.

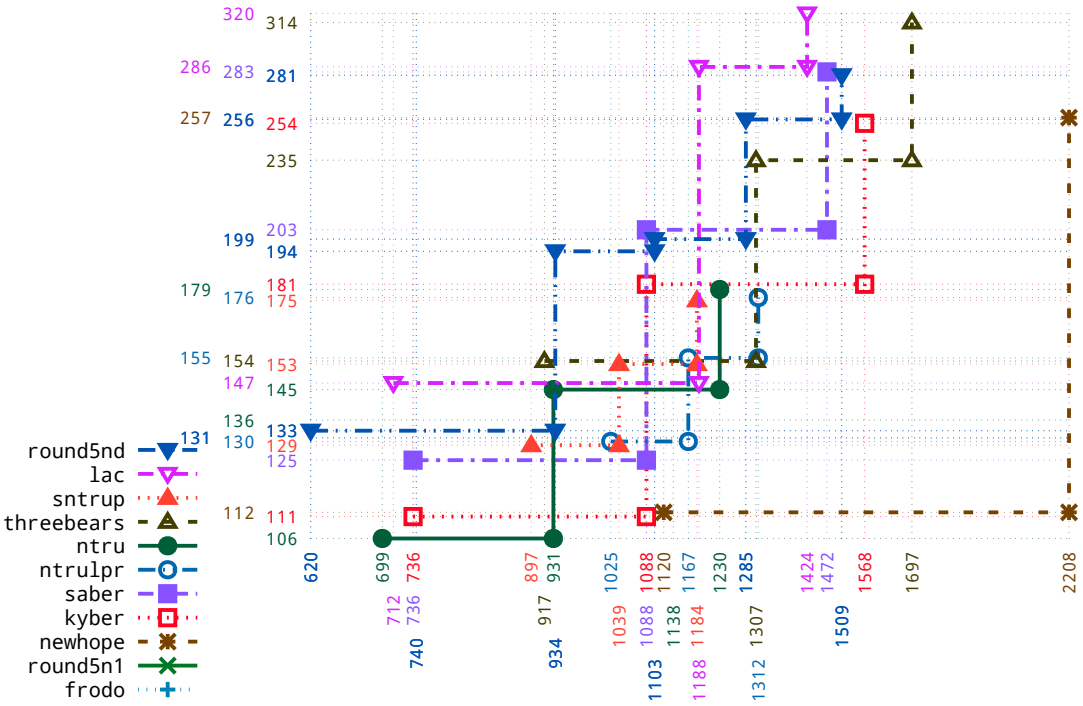

Fig. 3.10. Estimated security (vertical axis, log scale) vs. ciphertext size (horizontal axis, bytes, log scale). This is based on Figure 3.5 but with arguably unfair choices of "dash types". See also Section 2 for problems with the data points.

[5] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, Christine van Vredendaal, *NTRU Prime: round 2* (2019). URL: https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. Citations in this document: §2.1, §2.1, §2.1, §2.1, §2.1, §2.1, §2.1, §7.

[6] Cong Chen, Oussama Danba, Jeffrey Hoffstein, Andreas Hulsing, Joost Rijneveld, John M. Schanck, Peter Schwabe, William Whyte, Zhenfei Zhang, *NTRU: algorithm specifications and supporting documentation* (2019). URL: https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. Citations in this document: §2.1, §2.1, §2.1, §2.1, §5.

[7] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Frederik Vercauteren, *SABER: Mod-LWR based KEM (round 2 submission)* (2019). URL: https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. Citations in this document: §2.1, §2.1, §2.1.

[8] Michael B. Dillencourt, David Eppstein, Michael T. Goodrich, *Choosing colors for geometric graphs via color space embeddings*, in [**12**] (2007), 294–305. URL: https://www.ics.uci.edu/~goodrich/pubs/color.pdf. Citations in this document: §3.6.

[9] Mike Hamburg, *Post-quantum cryptography proposal: ThreeBears* (2019). URL: https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. Citations in this document: §2.1, §2.1, §2.1.

[10] Markus Hartikainen, Kaisa Miettinen, Margaret M. Wiecek, *Constructing a Pareto front approximation for decision making*, Mathematical Methods of Operations Research **73** (2011), 209–234. URL: https://jyx.jyu.fi/bitstream/handle/123456789/51010/hartikainenconstructingapareto.pdf. Citations in this document: §3.11.

[11] Jana Iyengar, Martin Thomson (editors), *QUIC: a UDP-based multiplexed and secure transport*, version 20 (2019). URL: https://datatracker.ietf.org/doc/draft-ietf-quic-transport/. Citations in this document: §2.3.

[12] Michael Kaufmann, Dorothea Wagner (editors), *Graph drawing, 14th international symposium, GD 2006, Karlsruhe, Germany, September 18–20, 2006, revised papers*, Lecture Notes in Computer Science, 4372, Springer, 2007. ISBN 978-3-540-70903-9. See [8].

[13] Adam Langley, *CECPQ1 results* (2016). URL: https://www.imperialviolet.org/2016/11/28/cecpq1.html. Citations in this document: §2.3.

[14] Adam Langley, *Post-quantum confidentiality for TLS* (2018). URL: https://www.imperialviolet.org/2018/04/11/pqconftls.html. Citations in this document: §2.3.

[15] Lu Lu, Christine M. Anderson-Cook, *Adapting the hypervolume quality indicator to quantify trade-offs and search efficiency for multiple criteria decision making using Pareto fronts*, Quality and Reliability Engineering International **29** (2013). Citations in this document: §3.11.

[16] Xianhui Lu, Yamin Liu, Dingding Jia, Haiyang Xue, Jingnan He, Zhenfei Zhang, Zhe Liu, Hao Yang, Bao Li, Kunpeng Wang, *LAC: Lattice-based Cryptosystems* (2019). URL: https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. Citations in this document: §2.1, §2.1, §2.1.

[17] David McGrew, *Living with postquantum cryptography* (2015). URL: https://csrc.nist.gov/csrc/media/events/workshop-on-cybersecurity-in-a-post-quantum-world/documents/presentations/session4-mcgrew-david.pdf. Citations in this document: §2.3.

[18] Brandon Mulcahy, *CIELAB color picker* (2015). URL: https://github.com/jangler/labrat. Citations in this document: §3.6.

[19] Edward R. Tufte, *The visual display of quantitative information*, 2nd edition, Graphics Press, 2001. ISBN 978-0961392147. Citations in this document: §3.11.