# Envelope Quadratizations: A method that can quadratize with a negative number of auxiliary variables

Nike Dattani[1] and Andreas Soteriou[2]

[1]*Harvard-Smithsonian Center for Astrophysics, USA*
[2]*Surrey University, Department of Electrical and Electronic Engineering, UK*

We present a quadratization method that we have found to be superior to all previously known quadratization methods for every case tested so far, in terms of requiring fewer auxiliary variables, less connectivity between the variables, and/or smaller coefficients. The quadratizations presented in this work, are also obtainable with orders of magnitude less CPU time than current state-of-the-art methods. The idea is that many *very* simple quadratic functions might *almost* reproduce a degree-$k$ function perfectly, but each of these *almost* perfect quadratic functions can compensate for each others imperfections, hence leading to a perfect quadratization when all such quadratic functions are considered collectively. It is the *envelope* of multiple quadratic functions which reproduces the original degree-$k$ function, so we call our method *envelope quadratization*. We are able to quadratize positive degree-$k$ monomials with quadratic functions that have *much smaller* coefficients than in [Boros *et al.* (2018) "Compact quadratizations of symmetric pseudo-Boolean functions: sub-linear bounds on the number of auxiliary variables"], while also having *far* less connectivity between the variables of the quadratics, and requiring fewer auxiliary variables: We will also show an example in which an envelope of quadratic functions, each with *fewer* total variables than the original degree-$k$ function, exactly reproduces the degree-$k$ function, meaning that we effectively have a 'negative' number of auxiliary variables. We will also show an application of our method to the types of degree-$k$ functions arising in a computer vision algorithm for image deblurring.

## I. INTRODUCTION

Any real-valued function of binary variables can be turned into a quadratic one that maintains all the desired properties of the original degree-$k$ function [? ]. We call this a 'quadratization' of the degree-$k$ function. The first quadratization technique of which we are aware, is Rosenberg's substitution method from 1975 [? ], which for quadratizing a single positive monomial of degree-$k$, would need $k-2$ auxiliary variables (meaning that the quadratic function obtained after the quadratization, would involve $k-2$ more auxiliary variables than the original degree-$k$ monomial). This was reduced by roughly one half in 2011 by Ishikawa, who introduced a quadratization for a positive degree-$k$ monomial, which needed only $\frac{k-1}{2}$ auxiliary variables (rounded down if $k$ were even). In 2018 several new quadratizations were introduced for positive degree-$k$ monomials: one required only $k/4$ auxiliary variables, one required only $\log_2 k$ (both of these were presented in [? ]), and while it might be surprising that an improvement can be made over $\log_2 k$, a subsequent 2018 paper finally showed a quadratization which required only $\log_2 (k/2)$ [? ] (all three of these numbers are rounded *up* to the nearest integer). While it was astonishing that any positive monomial can be quadratized with only $\log_2 (k/2)$ auxiliary variables, the resulting quadratic function would have very large coefficients and all possible quadratic terms over all original and auxiliary variables; and since large coefficients and a large number of quadratic terms, are two of the factors which most severely increase the cost when optimizing a quadratic discrete function, one may hesitate to use such quadratizations in practice.

Quadratization is also possible without the addition of *any* auxiliary variables [? ? ? ? ? ], and while these techniques should always be attempted, they in many cases might only reproduce the ground state of the degree-$k$ function rather than the whole spectrum [? ], and in many cases they might also be very expensive. One of these zero-auxiliary-variable methods that does reproduce the full spectrum, the split-reduc method [? ], can be considered a special case of our *envelope quadratization* method: While split-reduc can be very powerful when simultaneously quadratizing multiple terms that have common variables, its application can be far more costly than the methods described in the previous paragraph when quadratizing functions such as the degree-$k$ positive monomial.

In this work we introduce the concept of envelope quadratizations, which seem to be capable of 'perfectly' quadratizing functions more efficiently than any of the above methods (by 'perfectly' we mean, reproducing the whole spectrum of possible input/output pairs for the degree-$k$ function, since there exist methods such as [? ? ? ] that might be more efficient when the goal is only to reproduce the ground state). Even when the goal is only to preserve the ground state, envelope quadratizations are often the most efficient way to quadratize a function; and in other cases they should likely always be considered in combination with other methods in order to achieve the most efficient quadratization possible. We have not yet found a single case in which the envelope of more than one quadratic function is not *strictly* more efficient than *all* of the previously known methods for perfect quadratization.

## II. A QUICK EXAMPLE

Let us review the idea behind most known quadratizations. Consider the equality:

$$b_1 b_2 b_3 = \min_{b_a} \left( b_1 b_a + b_2 b_3 - 2b_2 b_a - 2b_3 b_a + 3b_a \right), \quad (1)$$

where all $b_i$ are either 0 or 1 and the subscript $a$ in $b_a$ indicates that this is an 'auxiliary' variable because it does not appear in the original cubic function on the left side. For clarity we explicitly show why Eq. 1 is true by constructing the truth tables for the left and right sides, starting with the left side:

| $b_1$ | $b_2$ | $b_3$ | $b_1 b_2 b_3$ |
|---|---|---|---|
| 0 | 0 | 0 | **0** |
| 0 | 0 | 1 | **0** |
| 0 | 1 | 0 | **0** |
| 0 | 1 | 1 | **0** |
| 1 | 0 | 0 | **0** |
| 1 | 0 | 1 | **0** |
| 1 | 1 | 0 | **0** |
| 1 | 1 | 1 | **1** |

For the right side we have:

| $b_1$ | $b_2$ | $b_3$ | $b_a$ | $b_1 b_a + b_2 b_3 - 2b_2 b_a - 2b_3 b_a + 3b_a$ | $\min_{b_a}$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | **0** | 0 | |
| 0 | 0 | 0 | **1** | 3 | **0** |
| 0 | 0 | 1 | **0** | 0 | |
| 0 | 0 | 1 | **1** | 1 | **0** |
| 0 | 1 | 0 | **0** | 0 | |
| 0 | 1 | 0 | **1** | 1 | **0** |
| 0 | 1 | 1 | **0** | 1 | |
| 0 | 1 | 1 | **1** | 0 | **0** |
| 1 | 0 | 0 | **0** | 0 | |
| 1 | 0 | 0 | **1** | 4 | **0** |
| 1 | 0 | 1 | **0** | 0 | |
| 1 | 0 | 1 | **1** | 2 | **0** |
| 1 | 1 | 0 | **0** | 0 | |
| 1 | 1 | 0 | **1** | 2 | **0** |
| 1 | 1 | 1 | **0** | 1 | |
| 1 | 1 | 1 | **1** | 1 | **1** |

It is impossible to construct a single quadratic function without any auxiliary variables, that reproduces the spectrum as seen in the tables above. However, minimizing over the quadratic function $b_1 b_2$ and the linear function $b_3$, perfectly yields the full spectrum of $b_1 b_2 b_3$ without introducing any auxiliary variables:

| $b_1$ | $b_2$ | $b_3$ | $b_1 b_2$ | $b_3$ | $\min(b_1 b_2, b_3)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **0** |
| 0 | 0 | 1 | | 1 | **0** |
| 0 | 1 | 0 | 0 | 0 | **0** |
| 0 | 1 | 1 | | 1 | **0** |
| 1 | 0 | 0 | 0 | 0 | **0** |
| 1 | 0 | 1 | | 1 | **0** |
| 1 | 1 | 0 | 1 | 0 | **0** |
| 1 | 1 | 1 | | 1 | **1** |

**We call the set** $\{b_1 b_2, b_3\}$ **an** *envelope quadratization* **for** $b_1 b_2 b_3$. **At each point** $(b_1, b_2, b_3)$**, the minimum (tangent) over all functions is equal to the degree-$k$ function** $b_1 b_2 b_3$.

## III. GENERALIZATIONS OF THE QUICK EXAMPLE

The simple example from the previous section, of a quadratic envelope reproducing $b_1 b_2 b_3$, can be generalized:

$$b_1 b_2 b_3 \ldots b_k = \min \left( b_1 b_2 \ldots b_{k_1}, b_{k_1+1} b_{k_1+2} \ldots b_{k_2}, \quad (2) \right.$$
$$\left. b_{k_2+1} b_{k_2+2} \ldots b_{k_3}, \ldots, b_{k_p+1} b_{k_p+2} \ldots b_k \right),$$

where $k_1, k_2, \ldots, k_p$ can be chosen arbitrarily between 1 and $k$, and if $k_1 = 2$ and all other $k_i = k_{i-1} + 2$ then it is an envelope quadratization. One interesting envelope quadratization which is an example of Eq. 2 is:

$$b_1 b_2 b_3 \ldots b_k = \min \left( b_1, b_2, b_3, \ldots, b_k \right), \quad (3)$$

which is actually an envelope *linearization* (and therefore also an envelope quadratization since all linear functions are examples of quadratic functions with the coefficient of the quadratic term equal to zero).

## IV. USAGE

### A. Quadratizing part of a function using envelope quadratizations

If we wished to minimize the function of binary variables:

$$3b_1 b_2 b_3 - 2b_1 b_2 - 2b_1 b_3 + 4b_2 b_4 - 3b_4 - 3b_2 b_5 \quad (4)$$

using the wealth of techniques we have available for minimizing *quadratic* functions of binary variables (such as the classical algorithm 'QPBO' and extensions of it [?], or quantum annealing using thousands of

qubits [? ] connected by graphs as complicated as Pegasus [? ? ]), we can quadratize the cubic term with the envelope quadratization $b_1b_2b_3 = \min(b_1b_2, b_3)$, and we get two objective functions to consider:

$$3b_1b_2 - 2b_1b_2 - 2b_1b_3 + 4b_2b_4 - 3b_4 - 2b_2b_5 \quad (5)$$

$$3b_3 - 2b_1b_2 - 2b_1b_3 + 4b_2b_4 - 3b_4 - 2b_2b_5. \quad (6)$$

Running the optimization procedure (for example QPBO or quantum annealing) on both functions would result in the following minima:

$$\text{Eq.5: -5 at } b_1, b_2, b_3, b_4, b_5 = 10110 \text{ \& } 10111 \quad (7)$$

$$\text{Eq.6: -4 at } b_1, b_2, b_3, b_4, b_5 = 11001. \quad (8)$$

### B. Quadratizing multiple parts of a function using envelope quadratizations

To show that the method of envelope quadratizations can be applied to multiple parts of a degree-$k$ objective function, consider the example:

$$2b_1b_2b_3b_4 - 3b_2b_4 + 3b_1b_4b_6 - 2b_3b_6 - 3b_1b_5 + b_5 + b_6 - b_3. \quad (9)$$

Here we quadratize $b_1b_2b_3b_4$ with $\{b_1b_2, b_3b_4\}$ and $b_1b_4b_6$ with $\{b_1b_4, b_6\}$. This gives us four possible objective functions to consider:

$$2b_1b_2 - 3b_2b_4 + 3b_1b_4 - 2b_3b_6 - 3b_1b_5 + b_5 + b_6 - b_3 \quad (10)$$

$$2b_1b_2 - 3b_2b_4 + 3b_6 - 2b_3b_6 - 3b_1b_5 + b_5 + b_6 - b_3 \quad (11)$$

$$2b_3b_4 - 3b_2b_4 + 3b_1b_4 - 2b_3b_6 - 3b_1b_5 + b_5 + b_6 - b_3 \quad (12)$$

$$2b_3b_4 - 3b_2b_4 + 3b_6 - 2b_3b_6 - 3b_1b_5 + b_5 + b_6 - b_3 \quad (13)$$

Running the optimization procedure on all four functions would result in the following minima:

$$\text{Eq.10: -5 at } b_1, b_2, b_3, b_4, b_5, b_6 = 011101 \quad (14)$$

$$\text{Eq.11: -4 at } b_1, b_2, b_3, b_4, b_5, b_6 = 011100 \text{ \& } 111110 \quad (15)$$

$$\text{Eq.12: -4 at } b_1, b_2, b_3, b_4, b_5, b_6 = 101011 \text{ \& } 111011 \quad (16)$$

$$\text{Eq.13: -5 at } b_1, b_2, b_3, b_4, b_5, b_6 = 111011. \quad (17)$$

The functions which have the lowest minima have values of $-5$ at $011101$ and $111011$ respectively, so these are the global minima of the original degree-$k$ function (again, this can be verified by simply optimizing the original quartic function, but for more complicated

functions, the algorithms to optimize a super-quadratic function can be much more costly than optimizing several quadratic functions, or one quadratic function with auxiliary variables, or even several quadratic functions that each contain auxiliary variables).

We note that it is not only the global minima (or ground states), that are reproduced by our envelope quadratization. The full spectrum is actually reproduced, meaning that for each of the $2^6 = 64$ combinations for: $(b_1, b_2, b_3, b_4, b_5, b_6)$, we have $f = \min_{f_1, f_2, f_3, f_4}(f_1, f_2, f_3, f_4)$, where $f$ is the function in Eq. 9 and $f_1, f_2, f_3, f_4$ are the functions in Eqs. 10-13 respectively. The 64-row table showing that $f = \min_{f_1, f_2, f_3, f_4}(f_1, f_2, f_3, f_4)$ for all 64 possible configurations $(b_1, b_2, b_3, b_4, b_5, b_6)$ is given in our Appendix.

### V. WORST-CASE COST ANALYSIS

Unless there is some property of the function making it easier, optimizing a function of $n$ binary variables by brute force requires $2^n$ queries, so if $m$ binary variables are added in the quadratization process, the cost becomes $2^n 2^m$ (meaning the cost is increased by a factor of $2^m$). While it is true that better optimization algorithms exist than doing brute force queries (hence the reason for quadratizing the original function at the expense of a possible $2^m$ increase in cost!), all general-purpose methods for optimizing such functions that are at least quadratic, still cost $\mathcal{O}(2^x)$ for $x$ binary variables (this includes quantum annealing, where the exponential scaling up to ~1000 variables can be seen in the figures of [? ] and [? ]), so we claim that a factor of $2^m$ increase in cost is not unfair for analyzing costs of quadratizations that add $m$ auxiliary variables (especially in the absence of any better measure for the added cost to the optimization due to adding auxiliary variables) .

Having established our notion of how much adding $m$ auxiliary variables increases the cost of optimizing a quadratic function of binary variables, we will also mention that having to do $r$ optimizations instead of 1, can be considered to increase the cost of the overall process by $r$ times the cost of doing one optimization.

So if a quadratization splits the original optimization problem into $r$ cases (or 'runs' of the optimization process) that each need to be optimized in order to find the global optimum, and each of these runs involves at most $m$ auxiliary variables, then the following is an upper bound on the added cost $C$ due to quadratizing:

$$C \leq r2^m. \quad (18)$$

In practice, some runs may require fewer auxiliary variables than others (or none at all), so the added cost can be lower than $r2^m$.

## VI. CLAIMS OF SUPERIORITY OVER ALL PREVIOUSLY KNOWN METHODS

### A. Positive degree-$k$ monomial

Envelope quadratizations are able to *strictly* outperform the previous state-of-the-art quadratization method for quadratizing a single positive degree-$k$ monomial. In 2018, Boros *et al.* published a new method that requires only $\log_2(k/2)$ auxiliary variables (rounded up) [? ?], and in the handbook of quadratizations [? ] this method was called **PTR-BCR-4**.

For the positive degree-$k$ monomial we compare the added cost of envelope quadratizations to the added cost of PTR-BCR-4. PTR-BCR-4 adds $m = \lceil \log_2(k/2) \rceil$ auxiliary variables, which means that the added cost $C$ of optimization is in the interval $[k/2, k-1]$ with $C = k/2$ if $k$ is a power of 2 and $C = k-1$ if $k$ is 1 less than the next power of 2, and in between otherwise (see footnote[1]).

Envelope quadratizations are for all $k$, capable of providing quadratizations for the positive degree-$k$ monomial with an added cost factor of $C = \lceil k/2 \rceil$ (using Eq. 2 we can factor the monomial into $k/2$ quadratic monomials if $k$ is even, and one extra linear monomial if $k$ is odd). If $k$ is even, $C = k/2$ which PTR-BCR-4 matches *only if $k$ is a power of 2*. If $k$ is odd, then $C = \lceil k/2 \rceil = \frac{k+1}{2}$ and PTR-BCR-4 *only if $k$ is one less than a power of 2*. In these two unique cases, in which the cost of the envelope quadratization and PTR-BCR-4 are the same, we note that with the described envelope quadratization the quadratic functions obtained are just monomials with no change to the coefficient $c$ of the original degree-$k$ monomial being quadratized; whereas with PTR the quadratic function obtained has all possible quadratic terms for $k + \lceil \log_2 k/2 \rceil$ variables, and the coefficients range (in the very best possible case) from $-\frac{ck}{4}$ to $\frac{ck^2}{8}$ (see the PTR-BCR-4 page of [? ]). The enormous number of quadratic terms and the enormous range of coefficients in the quadratic function generated by PTR-BCR-4 make envelope quadratizations a superior choice even in these cases where the added cost factor estimated by Eq. 18 is the same.

---

[1] If $x$ is an integer then $\lceil x \rceil = x$, otherwise we have $x < \lceil x \rceil < x+1$. So if we have $m = \lceil \log_2 k/2 \rceil$, then when $\lceil \log_2 k \rceil$ is an integer (i.e. when $k$ is a power of 2) we have $2^m = 2^{\lceil \log_2 k/2 \rceil} = k/2$. When $k$ is *not* a power of 2, we have $2^{\log_2 k/2} < 2^{\lceil \log_2 k/2 \rceil} < 2^{(\log_2 k/2)+1}$ which means $k/2 < 2^m < k$, but we actually can write *tighter* bounds because $k$ and $2^m$ must be integers; for example the last integer smaller than $k$ is $k-1$, so we can write a tighter upper bound of $2^m \leq k-1$. At last we state that if $k$ is a power of 2 we have $2^m = k/2$ and otherwise we have $2^m \leq k-1$, **so overall, quadratizing PTR-BCR-4 adds a cost of a factor of $2^m$ which is somewhere in the range of $[k/2, k-1]$ depending on how close $k$ is to being an exact power of 2.**

For all other values of $k$ (anything not a power of 2 or one less than a power of 2, which constitutes the vast majority of the possible values of $k$), the added cost factor $C$ is strictly smaller than for PTR-BCR-4 by up to a factor of $\frac{2(k-1)}{(k+1)}$ (which is realized when $k$ is the odd number that is one more than a power of 2). As $k$ gets large, $(k-1)$ and $(k+1)$ become more and more similar, and the envelope quadratizations become roughly half as costly as PTR-BCR-4. We can think of envelope quadratizations as adding a cost factor that is equivalent to effectively giving, instead of $r > 1$ quadratic functions with $m = 0$ auxiliary variables, only $\tilde{r} = 1$ effective quadratic function with:

$$\lceil \log_2 k/4 \rceil < \tilde{m} \leq \lfloor \log_2 k/2 \rfloor \tag{19}$$

auxiliary variables. For example, when $k = 7$, which is 1 less than a power of 2, we have $(r, m) = (4, 1)$ and therefore $C = 1$, but $(\tilde{r}, \tilde{m}) = (1, \lfloor \log_2 7/2 \rfloor)$ also leads to $C = 4$; and when $k = 9$ which is 1 more than a power of 2, we have $(r, m) = (5, 1)$ leading to an added cost factor of $C = 5$ which is only slightly larger than $(r, m) = (1, \lceil \log_2 9/4 \rceil)$ for which $C$ would be 4. The strict inequality $\lceil \log_2 k/4 \rceil < \tilde{m}$ can get closer and closer to being an equality as $k$ gets large (for $k = 33$ we have $C = 17$ and $2^{\lceil \log_2 k/4 \rceil} = 16$.

### B. General functions of binary variables with real-valued coefficients

We have shown that when quadratizing a positive degree-$k$ monomial, envelope quadratizations are now the state-of-the-art in terms of the cost added to the optimization process. What about when quadratizing not only a single monomial but multiple terms in a function? PTR-BCR-4 is no longer necessarily the best quadratization. For example, quadratizing $b_1 b_2 b_3 b_4 b_5 + b_2 b_3 b_4 b_5 b_6$ would require 4 auxiliary variables (2 for each term), but Rosenberg's substitution would only need two auxiliary variables: $b_{a_1} = b_2 b_3$ and $b_{a_2} = b_4 b_5$. In this case we can make a 'trivial' argument that envelope quadratizations are at least as good as the previous state-of-the-art method, because the previous state-of-the-art method is just an example of an envelope quadratization in which $r = 1$. However, we have also found that for every case we have examined so far [put book on arXiv and cite it], we are also able to obtain an envelope quadratization with $r \geq 1$ that is strictly better than the previous state-of-the-art quadratization. There are some cases in which the added cost factor of Eq. 18 can be matched by previously known methods, but in those cases envelope quadratizations still have fewer quadratic terms and smaller coefficients and fewer non-submodular terms (for example, see Section VII A below).

## VII.   MORE EXAMPLES AND POTENTIAL APPLICATIONS

### A.   Positive degree-8 monomial

It has been known for some time that no positive monomial with degree greater than 4 can be quadratized by a single function with fewer than 2 auxiliary variables (for example, see the last line of Pg. 182 in [? ]). We therefore found it quite fascinating that we could find an envelope quadratization for the positive degree-8 monomial which involves at most 1 auxiliary variable in the two pieces:

$$b_1b_2b_3b_4b_5b_6b_7b_8 : \tag{20}$$
$$\longrightarrow \ 3b_a + b_1b_2 + b_1b_3 + b_1b_4 + b_2b_3 + b_2b_4 \tag{21}$$
$$+ \ b_3b_4 - 2b_a(b_1 + b_2 + b_3 + b_4) \tag{22}$$
$$\longrightarrow \ 3b_a + b_5b_6 + b_5b_7 + b_5b_8 + b_6b_7 + b_6b_8 \tag{23}$$
$$+ \ b_7b_8 - 2b_a(b_5 + b_6 + b_7 + b_8). \tag{24}$$

Here the added cost of a factor of $2 \times 2^1 = 4$ is still the same as it would be with PTR-BCR-4 (which would need only 2 auxiliary variables and therefore an added cost of a factor of $1 \times 2^2 = 4$). However the quadratic function arising from PTR-BCR-4 contains all possible quadratic terms for the 8 original variables and the 2 auxiliary variables, which would result in a *gadget graph* [? ] of $K_{10}$, whereas the above envelope quadratization would for both quadratic functions involve the much simpler gadget graph of $K_5$. Furthermore, the quadratic terms resulting from the PTR-BCR-4 quadratization would have coefficients ranging from -2 to 8, and in the above envelope quadratization, they range from -2 to 1, which means the coefficient range is more than 3 times smaller in the envelope quadratization.

### B.   Application to a computer vision algorithm

The image deblurring algorithm of [? ? ] involves degree-$k$ terms with $k = d^2$ with $d \geq 2$ being the length in pixels of the square-shaped mask). In all known applications so far, $d = 2$ and $k = 4$ (though larger masks can be used, it would just be more difficult since the optimization problem would involve terms of degree at least 9). With $d = 2$, this optimization problem involves degree-4 terms for every set of pixels that can be covered by the $2 \times 2$ pixel mask, so we can have neighboring degree-4 terms such as:

$$b_1b_2b_3b_4 + b_3b_4b_5b_6 = \min(b_2b_3 + b_3b_6, b_1b_4 + b_4b_5, \tag{25}$$
$$b_1b_2 + b_5b_6 - b_3 - b_4 + 2) \tag{26}$$

Likewise we have an envelope quadratization for trinomials (Eq. 27 equals the minimum over the three functions given in Eqs. 28-30):

$$b_1b_2b_3b_4 + b_2b_3b_4b_5 + b_3b_4b_5b_6 : \tag{27}$$
$$\longrightarrow 3b_3b_4 + b_3b_5 + b_4b_5 - b_3 - b_4 - b_5 + 1 \tag{28}$$
$$\longrightarrow b_1b_4 + b_3b_5 + b_4b_5 \tag{29}$$
$$\longrightarrow b_1b_2 + b_2b_6 + b_3b_5 + b_5b_6 + b_2 - b_3 - b_4 - b_5 + 2. \tag{30}$$

### C.   Envelope quadratization with a reduction in added cost factor of more than 7 compared to the previous state-of-the-art

We discovered the concept of envelope quadratizations on 3 September 2019, so the number of functions for which we have searched for envelope quadratizations so far has been limited by the amount of time available since then. However the best reduction in added cost factor (in the spirit of Eq. 18) we have found *so far* with envelope quadratizations compared to the previous state-of-the-art, was for the function:

$$b_1b_2b_3b_4b_5b_6b_7b_8 + b_2b_3b_4b_5b_6b_7b_8b_9 + b_3b_4b_5b_6b_7b_8b_9b_{10}. \tag{31}$$

This can be quadratized with $m = 6$ auxiliary variables by quadratizing each of the three terms by PTR-BCR-4. It can also be quadratized by a combination of pairwise covers [? ] and split-reduc [? ] with the same added cost factor as we would obtain with $m = 6$, which is $C = 64$. We have not been able to find a way to quadratize this function with $C < 64$ using techniques known to us before writing this paper. Below we show that with envelope quadratizations we can reduce the added cost factor down to $C = 9$ which is a factor of 7.1 times smaller than 64. We start by factoring Eq. 31 into the product of a quadratic function and a degree-6 function:

$$(b_1b_2 + b_2b_9 + b_9b_{10})b_3b_4b_5b_6b_7b_8. \tag{32}$$

Using Eq. 2 we can split the degree-6 factor into 3 cases, which results in the following 3 cases overall:

$$\longrightarrow (b_1b_2 + b_2b_9 + b_9b_{10})b_3b_4 \tag{33}$$
$$\longrightarrow (b_1b_2 + b_2b_9 + b_9b_{10})b_5b_6 \tag{34}$$
$$\longrightarrow (b_1b_2 + b_2b_9 + b_9b_{10})b_7b_8. \tag{35}$$

Each of these cases is a function which can be split by the envelope quadratization of Eq. 27, so we have $(r, m) = (9, 0)$ and $C = 9 \times 2^0 = 9$.

## VIII. DISCUSSION

### A. Speed of finding multi-run envelope quadratizations verus single-run quadratizations

Perhaps the only published algorithm for finding quadratizations of arbitrary functions, is the method which involves enumerating all vertices and rays of a convex polyhedron, as mentioned on Pg. 152 in section 3.8 of [**?** ]; and Pg. 153 in the same section estimates that it would take at least 3 full months of CPU time to do this for a quadratization into a quadratic function with 7 total variables (for example 6 original variables and 1 auxiliary variable). This is in contrast to the ~1 hour of CPU time it takes to do this for 6 total variables (for example 5 original variables and 1 auxiliary variable). Therefore, finding quadratizations with 1-auxiliary variable for a 7-variable function could take several years, and finding quadratizations with 1-auxiliary variable for an 8-variable function could take several centuries. However Section VII A shows a 1-auxiliary-variable envelope quadratization of an 8-variable function, which was found with only 3.52 seconds of CPU time. Compared to the envelope quadratization found with 3.52 seconds of CPU time, the centuries of estimated CPU time for vertex enumeration would aim to find a quadratization twice as efficient (by the standard of Eq. 18, because $r$ would be 1 instead of 2 and $m$ would be 1 in both cases), but we already know that the centuries of estimated CPU time would not find any $(r, m) = (1, 1)$ quadratization because it is already known that positive monomials of degree greater than 4 cannot be quadratized with $(r, m) = (1, 1)$: But the message we wish to convey is that if an $(r, m) = (1, 1)$ quadratization were to exist for some 8-variable function, it could take many centuries to find it using the only known other algorithm for finding quadratizations of arbitrary functions, and we were able to find in 3.52s an envelope quadratization with an equivalent added cost factor, and based on all other envelope quadratizations found so far, it is very likely that the coefficients and number of quadratic terms in the envelope quadratization would be significantly lower.

Since with envelope quadratizations we are no longer demanding that $r = 1$, we now have far more quadratizations available, and one could search through this space randomly or by some clever Monte Carlo algorithm, but we have found reinforcement learning to be the most successful so far (and it was reinforcement learning that was used in the above-mentioned calculation which took 3.52s: we will describe our reinforcement learning algorithm in a subsequent paper).

### B. Envelope quadratizations with effectively a 'negative' number of auxiliary variables

Two ways: quadratize the whole thing, or quadratize pieces such that other terms don't have the same variable in positive linear term?

### C. It is not yet the end of our journey

We originally wanted to find 'heuristic quadratizations', which would be quadratizations that do not necessarily need to reproduce a full spectrum, or even a ground state (optimum) of the original function, but would come very close (for example by finding the next lowest-energy state above the global minimum). This is because often we do not even need the *exact* ground state of a function, and all we seek is a good approximation to it, for example the true ground state is not being sought in the above-mentioned image de-blurring algorithm: it is an algorithm in which the closer we get to the optimum solution, the better the de-blurring will be, but eventually the difference between one de-blurring and a de-blurring closer to the optimal solution will hardly be noticeable, and eventually we may even reach the limitations of the de-blurring algorithm itself, meaning that reaching a lower-energy solution will no longer even guarantee a better de-blurring. When we are not seeking the absolute ground state of a function, then it would be overkill to demand that the quadratization exactly preserves the ground state, and this extra and unnecessary demand ought to be limiting us in terms of the efficiency of quadratizations available for us to find (for example we must be able to find quadratizations that do not require as many auxiliary variables, do not require such large coefficients, do not have such dense connectivity in the graph describing the quadratic terms, and do not have so many non-submodular terms, as our currently known quadratizations which do more than we need).

Envelope quadratizations not only reproduce the ground state perfectly (which is more than what we usually want), they actually reproduce the entire spectrum which contains $2^n$ states and their corresponding energies (which is *far* more than what we seek in almost all real-world applications). Therefore we have accidentally found something better than what we were looking for, but ultimately we should be able to reduce costs even more when we (successfully!) relax the requirement that all $2^n$ states of the original function are reproduced in the very end (we used in parentheses the adjective 'successfully' because envelope quadratizations were in fact *born* out of our pursuit to relax the requirement that all $2^n$ states are reproduced, and while we were very happy to discover envelope quadratizations, they turned out to satisfy the requirement that

we were originally hoping to relax).

We hope that by relaxing the requirement that a quadratization of an $n$-variable function must reproduce all $2^n$ states after minimizing over all runs, we may *further* improve the cost factor that our new state-of-the-art quadratizations add to the solving of degree-$k$ binary optimization problems when turning them into quadratic optimization problems..

uses multiple quadratic gadgets which are imperfect in that they do not on their own reproduce the *full* spectrum of the original degree-$k$ function, but they are complimentary in that we can reproduce the desired full spectrum when all imperfect gadgets complementing each other are combined and we minimize over them). Overall we prefer the term *envelope quadratizations* the most since it is the envelope of many quadratic functions which is our final quadratization that exactly reproduces the original degree-$k$ function.

## IX. ACKNOWLEDGMENTS

## Appendix A: Proof that the full spectrum is reproduced in the envelope quadratization of Section IV A

| $b_1$ | $b_2$ | $b_3$ | $b_4$ | $b_5$ | $f_1$ Eq. 5 | $f_2$ Eq. 6 | $\min f_1 f_2$ | $f$ Eq. 4 |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | -3 | -3 | -3 | -3 |
| 0 | 0 | 0 | 1 | 1 | -3 | -3 | -3 | -3 |
| 0 | 0 | 1 | 0 | 0 | 0 | 3 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 3 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | -3 | 0 | -3 | -3 |
| 0 | 0 | 1 | 1 | 1 | -3 | 0 | -3 | -3 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | -2 | -2 | -2 | -2 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | -1 | -1 | -1 | -1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 3 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | -2 | 1 | -2 | -2 |
| 0 | 1 | 1 | 1 | 0 | 1 | 4 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | -1 | 2 | -1 | -1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | -3 | -3 | -3 | -3 |
| 1 | 0 | 0 | 1 | 1 | -3 | -3 | -3 | -3 |
| 1 | 0 | 1 | 0 | 0 | -2 | 1 | -2 | -2 |
| 1 | 0 | 1 | 0 | 1 | -2 | 1 | -2 | -2 |
| 1 | 0 | 1 | 1 | 0 | -5 | -2 | -5 | -5 |
| 1 | 0 | 1 | 1 | 1 | -5 | -2 | -5 | -5 |
| 1 | 1 | 0 | 0 | 0 | 1 | -2 | -2 | -2 |
| 1 | 1 | 0 | 0 | 1 | -1 | -4 | -4 | -4 |
| 1 | 1 | 0 | 1 | 0 | 2 | -1 | -1 | -1 |
| 1 | 1 | 0 | 1 | 1 | 0 | -3 | -3 | -3 |
| 1 | 1 | 1 | 0 | 0 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 0 | 1 | -3 | -3 | -3 | -3 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | -2 | -2 | -2 | -2 |

**Appendix B: Proof that the full spectrum is reproduced in the envelope quadratization of Section IV B**

| $b_1\ b_2\ b_3\ b_4\ b_5\ b_6$ | $f_1$ Eq. 10 | $f_2$ Eq. 11 | $f_3$ Eq. 12 | $f_4$ Eq. 13 | min $f_1 f_2 f_3 f_4$ | $f$ Eq. 9 |
|---|---|---|---|---|---|---|
| 0 0 0 0 0 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 0 0 0 0 1 | 1 | 4 | 1 | 4 | 1 | 1 |
| 0 0 0 0 1 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 0 0 0 1 1 | 2 | 5 | 2 | 5 | 2 | 2 |
| 0 0 0 1 0 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 0 0 1 0 1 | 1 | 4 | 1 | 4 | 1 | 1 |
| 0 0 0 1 1 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 0 0 1 1 1 | 2 | 5 | 2 | 5 | 2 | 2 |
| 0 0 1 0 0 0 | -1 | -1 | -1 | -1 | -1 | -1 |
| 0 0 1 0 0 1 | -2 | 1 | -2 | 1 | -2 | -2 |
| 0 0 1 0 1 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 0 1 0 1 1 | -1 | 2 | -1 | 2 | -1 | -1 |
| 0 0 1 1 0 0 | -1 | -1 | 1 | 1 | -1 | -1 |
| 0 0 1 1 0 1 | -2 | 1 | 0 | 3 | -2 | -2 |
| 0 0 1 1 1 0 | 0 | 0 | 2 | 2 | 0 | 0 |
| 0 0 1 1 1 1 | -1 | 2 | 1 | 4 | -1 | -1 |
| 0 1 0 0 0 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 1 0 0 0 1 | 1 | 4 | 1 | 4 | 1 | 1 |
| 0 1 0 0 1 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 1 0 0 1 1 | 2 | 5 | 2 | 5 | 2 | 2 |
| 0 1 0 1 0 0 | -3 | -3 | -3 | -3 | -3 | -3 |
| 0 1 0 1 0 1 | -2 | 1 | -2 | 1 | -2 | -2 |
| 0 1 0 1 1 0 | -2 | -2 | -2 | -2 | -2 | -2 |
| 0 1 0 1 1 1 | -1 | 2 | -1 | 2 | -1 | -1 |
| 0 1 1 0 0 0 | -1 | -1 | -1 | -1 | -1 | -1 |
| 0 1 1 0 0 1 | -2 | 1 | -2 | 1 | -2 | -2 |
| 0 1 1 0 1 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 1 1 0 1 1 | -1 | 2 | -1 | 2 | -1 | -1 |
| 0 1 1 1 0 0 | -4 | -4 | -2 | -2 | -4 | -4 |
| 0 1 1 1 0 1 | -5 | -2 | -3 | 0 | -5 | -5 |
| 0 1 1 1 1 0 | -3 | -3 | -1 | -1 | -3 | -3 |
| 0 1 1 1 1 1 | -4 | -1 | -2 | 1 | -4 | -4 |
| 1 0 0 0 0 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 0 0 0 0 1 | 1 | 4 | 1 | 4 | 1 | 1 |
| 1 0 0 0 1 0 | -2 | -2 | -2 | -2 | -2 | -2 |
| 1 0 0 0 1 1 | -1 | 2 | -1 | 2 | -1 | -1 |
| 1 0 0 1 0 0 | 3 | 0 | 3 | 0 | 0 | 0 |
| 1 0 0 1 0 1 | 4 | 4 | 4 | 4 | 4 | 4 |
| 1 0 0 1 1 0 | 1 | -2 | 1 | -2 | -2 | -2 |
| 1 0 0 1 1 1 | 2 | 2 | 2 | 2 | 2 | 2 |
| 1 0 1 0 0 0 | -1 | -1 | -1 | -1 | -1 | -1 |
| 1 0 1 0 0 1 | -2 | 1 | -2 | 1 | -2 | -2 |
| 1 0 1 0 1 0 | -3 | -3 | -3 | -3 | -3 | -3 |
| 1 0 1 0 1 1 | -4 | -1 | -4 | -1 | -4 | -4 |
| 1 0 1 1 0 0 | 2 | -1 | 4 | 1 | -1 | -1 |
| 1 0 1 1 0 1 | 1 | 1 | 3 | 3 | 1 | 1 |
| 1 0 1 1 1 0 | 0 | -3 | 2 | -1 | -3 | -3 |
| 1 0 1 1 1 1 | -1 | -1 | 1 | 1 | -1 | -1 |
| 1 1 0 0 0 0 | 2 | 2 | 0 | 0 | 0 | 0 |
| 1 1 0 0 0 1 | 3 | 6 | 1 | 4 | 1 | 1 |
| 1 1 0 0 1 0 | 0 | 0 | -2 | -2 | -2 | -2 |
| 1 1 0 0 1 1 | 1 | 4 | -1 | 2 | -1 | -1 |
| 1 1 0 1 0 0 | 2 | -1 | 0 | -3 | -3 | -3 |
| 1 1 0 1 0 1 | 3 | 3 | 1 | 1 | 1 | 1 |
| 1 1 0 1 1 0 | 0 | -3 | -2 | -5 | -5 | -5 |
| 1 1 0 1 1 1 | 1 | 1 | -1 | -1 | -1 | -1 |
| 1 1 1 0 0 0 | 1 | 1 | -1 | -1 | -1 | -1 |
| 1 1 1 0 0 1 | 0 | 3 | -2 | 1 | -2 | -2 |
| 1 1 1 0 1 0 | -1 | -1 | -3 | -3 | -3 | -3 |
| 1 1 1 0 1 1 | -2 | 1 | -4 | -1 | -4 | -4 |
| 1 1 1 1 0 0 | 1 | -2 | 1 | -2 | -2 | -2 |
| 1 1 1 1 0 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 1 1 1 1 0 | -1 | -4 | -1 | -4 | -4 | -4 |
| 1 1 1 1 1 1 | -2 | -2 | -2 | -2 | -2 | -2 |