Low Autocorrelation Binary Sequences

Tom Packebusch\textsuperscript{1} and Stephan Mertens\textsuperscript{1,2}
\textsuperscript{1}Inst. f. Theo. Physik, Otto-von-Guericke Universität, PF 4120, 39016 Magdeburg, Germany
\textsuperscript{2}Santa Fe Institute, 1399 Hyde Park Rd, Santa Fe, NM 87501, USA
E-mail: mertens@ovgu.de

Abstract. Binary sequences with minimal autocorrelations have applications in communication engineering, mathematics and computer science. In statistical physics they appear as groundstates of the Bernasconi model. Finding these sequences is a notoriously hard problem, that so far can be solved only by exhaustive search. We review recent algorithms and present a new algorithm that finds optimal sequences of length $N$ in time $O(N^{1.73})$. We computed all optimal sequences for $N \leq 66$ and all optimal skewsymmetric sequences for $N \leq 119$.

1. Introduction

Consider a sequence $S = (s_1, \ldots, s_N)$ with $s_i = \pm 1$. The autocorrelations of $S$ are defined as

$$C_k(S) = \sum_{i=1}^{N-k} s_i s_{i+k}$$

for $k = 0, 1, \ldots, N-1$, and the “energy” of $S$ is defined as the sum of the squares of all off-peak correlations,

$$E(S) = \sum_{k=1}^{N-1} C_k^2(S).$$

The low-autocorrelation binary sequence (LABS) problem is to find a sequence $S$ of given length $N$ that minimizes $E(S)$ or, equivalently, maximizes the merit factor

$$F(S) = \frac{N^2}{2E(S)}.$$

The LABS problem arises in practical applications in communications engineering, where low autocorrelation sequences are used for example as modulation pulses in radar and sonar ranging \cite{1,3}. A particularly exciting application is the interplanetary radar measurement of spacetime curvature \cite{4}.

In mathematics, the LABS problem appears in terms of the Littlewood problem \cite{5,6}, the problem of constructing polynomials with coefficients $\pm 1$ that are “flat” on the unit circle in the complex plane.

In statistical physics, $E(S)/N$ can be interpreted as the energy of $N$ interacting Ising spins $s_i = \pm 1$. This is the Bernasconi model \cite{7}. It has long-range 4-spin interactions and is completely deterministic, i.e. there is no explicit or quenched disorder like in spin-glasses. Nevertheless the ground states are highly disordered – quasi by definition. This self-induced disorder resembles very much the situation in real glasses. In fact, the Bernasconi-model exhibits features of a glass transition like a jump in the specific heat and slow dynamics and
Low Autocorrelation Binary Sequences

A clever variation of the replica method allows an analytical treatment of the Bernasconi model in the high-temperature regime \([9,10]\). For the low-temperature regime, analytical results are rare – especially the ground states are not known. Due to this connection to physics we refer to the \(s_i\) as spins throughout the paper.

These examples illustrate the importance of the LABS problem in various fields. For more applications and the history of the problem we refer to existing surveys \([11,12]\). In this contribution we focus on algorithms to solve the LABS problem. But before we discuss algorithms, we will give a brief survey on what is known about solutions.

2. What is known

The correlation \(C_k\) is the sum of \(N - k\) terms \(±1\), hence the value of \(|C_k|\) is bounded from below by

\[
|C_k| \geq b_k = (N - k) \mod 2 .
\]

A binary sequence with \(|C_k| = b_k\) is called a Barker sequence \([13]\). The merit factor of a Barker sequence is

\[
F^\text{Barker}_N = \begin{cases} 
N & \text{for } N \text{ even}, \\
\frac{N^2}{N-1} & \text{for } N \text{ odd}.
\end{cases}
\]

If it exists, a Barker sequence is a solution of the LABS problem. Barker sequences exist for \(N = 2, 3, 4, 5, 7, 11, 13\), but probably for no other values of \(N\). In fact it can be proven that there are no Barker sequences for odd values of \(N > 13\) \([14,15]\). For even values of \(N\), the existence of Barker sequences can be excluded for \(4 < N \leq 2 \cdot 10^3\) \([16]\).

Let \(F_N\) denote the maximum merit factor for sequences of length \(N\). It is an open problem to prove (or disprove) that \(F_N\) is bounded. For Barker sequences, \(F_N \approx N\), and the same is true more generally for sequences such that \(|C_k| \leq C^*\) for some constant \(C^*\) that does not depend on \(N\) or \(k\). The common belief is that no such sequences exist and that \(F_N\) is bounded by some constant.

A non-rigorous argument for \(F_N\) being bounded was given by Golay \([17]\). Assuming that the correlations \(C_k\) are independent, he argued that asymptotically \(F_N \lesssim 12.3248\), or more precisely, that

\[
F_N \lesssim \frac{12.3248}{\left(\frac{8\pi N}{N}\right)^3} .
\]

There are some rigorous results for lower bounds on \(F_N\). The mean value of \(1/F\), taken over all binary sequences of length \(N\), is \((N - 1)/N\) \([18]\). Hence we expect \(F_N \geq 1\). In fact one can explicitly construct sequences for all values of \(N\) that have merit factors larger than 1. The current record is set by so called appended rotated Legendre sequences with an asymptotic merit factor of 6.342061 \ldots \([19,20]\).

Beyond that, our knowledge about solutions of the LABS problem is based on computer searches. Figure 1 shows the best merit factors known for \(N < 300\). For small values of \(N\), we can exhaustively search through all sequences to find the sequences with the maximum merit factor \(F_N\). An evaluation of \(E(S)\) from scratch takes time \(\Theta(N^2)\), but one can loop through all sequences such that any two successive sequences differ by exactly one spin, an arrangement known as Gray code \([22]\). The corresponding update of \(E(S)\) takes only linear time, and the total time complexity of exhaustive enumeration is then given by \(\Theta(N2^N)\). In this paper we
will discuss a class of exact enumeration algorithms with time complexity $\Theta(N b^N)$ with $b < 2$ that we used to solve the LABS problem up to $N \leq 66$.

For larger values of $N$ exhaustive enumeration is not feasible and one has to resort to either partial enumerations or heuristic searches. In both cases one obtains sequences with large but not necessarily maximal merit factors.

Partial enumerations are exhaustive enumerations of a well defined subset of sequences. A particular promising subset is given by skewsymmetric sequences of odd length $N = 2n - 1$. These sequences satisfy

$$s_{n+\ell} = (-1)^\ell s_{n-\ell} \quad (\ell = 1, \ldots, n-1),$$

which implies that $C_k = 0$ for all odd $k$. The restriction to skewsymmetric sequences reduces the size of the search space from $2^N$ to $2^{N/2}$. Sequences with maximum merit factor are often, but not always skewsymmetric: from the 31 LABS problems for odd $N \leq 65$, 21 have skewsymmetric solutions (Section 4). For the other values of $N$, skewsymmetric sequences provide lower bounds for $F_N$. We used our enumeration algorithm to compute the optimal skewsymmetric sequences for all $N \leq 119$.

Enumerative algorithms (complete or partial) are limited to small values of $N$ by the exponential size of the search space. Heuristic algorithms use some plausible rules to locate good sequences more quickly. Examples are simulated annealing, evolutionary algorithms,
tabu search—the list of heuristic algorithms that have been applied to the LABS problem is much longer, see [23]. The state of the art are the solvers described in [21], which have found many of the merit factors shown in Figure 1. The figure shows a significant drop of the merit factors for \( N > 200 \). This is generally attributed to the fact that even sophisticated search heuristics fail for LABS problems of larger size. This hardness has earned the LABS problem a place in CSPLIB, a library of test problems for constraint solvers [24, problem 005].

3. Algorithm

According to the current state of knowledge, the only way to get exact solutions for the LABS problem is exhaustive search. With a search space that grows like \( 2^N \), this approach is limited to rather small values of \( N \), however. The exponential complexity calls for a method to restrict the search to smaller subspaces without missing the exact solutions. This is where branch&bound comes in, a powerful and versatile method from combinatorial optimization [25]. All exact solutions of the LABS problem for \( N > 32 \) have been obtained with variations of a branch&bound algorithm proposed in [26] that reduces the size of the search space from \( 2^N \) to \( b^N \) with \( b < 2 \). In this section we review these algorithms and we present a new variant which has \( b = 1.72 \), the best value to date.

The idea of branch&bound is to solve a discrete optimization problem by breaking up its feasible set into successively smaller subsets (branch), calculating bounds on the objective function value over each subset, and using them to discard certain subsets from further consideration (bound) [25]. The procedure ends when each subset has either produced a feasible solution, or has been shown to contain no better solution than the one already in hand. The best solution found during this procedure is a global optimum.

The goal is of course to discard many subsets as early as possible during the branching process, i.e. to discard most of the feasible solutions before actually evaluating them. The success of this approach depends on the branching rule and very much on the quality of the bound, but it can be quite substantial.

For the LABS problem we specify a set of feasible solutions be fixing the \( m \) leftmost and the \( m \) rightmost spins of the sequence. The \( N - 2m \) centre spins are not specified, i.e. the set contains \( 2^{N-2m} \) feasible solutions. Given a feasible set specified by the \( 2m \) outer elements, four smaller sets are created by fixing the elements \( s_{m+1} \) and \( s_{N-m} \) to \( \pm 1 \) and \( m \) is increased by 1. This is applied recursively until all elements have been fixed. This is the branching rule introduced by the original branch&bound algorithm [26], and it is shared by all later versions. It has the nice property that the long range correlations are fixed early in the recursion process. Specifically, if the \( m \) left- and rightmost spins are fixed, all \( C_k \) for \( k \geq N - m \) are fixed. In addition, this branching rule supports the computation of lower bounds very well, as we will see below.

The branching process can be visualized as a tree in which nodes represent subsets. Each node has four children corresponding to the four possible ways to set the two spins in the \((m+1)\)th shell. The branch&bound algorithm traverses this tree and tries to exclude as many branches as possible by computing a bound on the energy that can be achieved in a branch. The number of nodes actually visited is a measure of quality for the bound.

3.1. Bounds

Bounds are usually obtained by replacing the original problem over a given subset with an easier (relaxed) problem such that the solution value of the latter bounds that of the former.
A good relaxation is one that is easy and fast to solve and yields strong lower bounds. Most often these are conflicting goals.

An obvious relaxation of the LABS problem is given by the problem to minimize all values $C_k^2$ independently. Hence we replace the original problem

$$E_{\min} = \min_{\text{free}} \left( \sum_{k=1}^{N-1} C_k^2 \right)$$

by the relaxed version

$$E_{\min}^* = \sum_{k=1}^{N-1} \min_{\text{free}} (C_k^2) = \min_{\text{free}} \left( \left( \sum_{k=1}^{N-1} |C_k| \right)^2 \right) \leq E_{\min},$$

where “free” refers to the $N - 2m$ center elements of $s$ that have not yet been assigned. All previous branch&bound approaches to LABS considered $E_{\min}^*$ to be too expensive to compute and replaced it by a weaker, but easily computable bound $E_b \leq E_{\min}^*$ obtained from bounding $\min_{\text{free}} |C_k|$ from below.

### 3.1.1. The original bound

In the original algorithm \cite{26} the bound $E_b$ is computed by assigning (arbitrary) values to all free spins, thereby fixing the values for all correlations to $C_k^*$. Since flipping a free spin can decrement $|C_k|$ at most by 2, a lower bound for $|C_k|$ is given by

$$\min_{\text{free}} |C_k| \geq \max(b_k, |C_k^*| - \hat{f}_k),$$

where

$$\hat{f}_k = \begin{cases} 
0 & \text{if } k \geq N - m, \\
2(N - m - k) & \text{if } N/2 \leq k < N - m \text{ or} \\
N - 2m & \text{otherwise}
\end{cases}$$

denotes the number of free spins that appear in $C_k$ and $b_k$ is given by \cite{4}. The running time of this algorithm scales like $O(1.85^N)$. A parallelized version of the algorithm was used to solve the LABS problem up to $N = 60$ \cite{27}.

### 3.1.2. The Prestwich bound

The quality of the bound \cite{10} depends on the values of $C_k^2$ and hence on the arbitrary values assigned to the free spins. In principle, these values should be chosen to maximize $C_k^*$, but this requires the solution of another optimization problem for each bound. This can be avoided by considering free products instead of free spins: a product $s_i s_{i+k}$ is free if $s_i$ or $s_{i+k}$ is a free spin. Products $s_i s_{i+k}$ in which both spins are fixed are called fixed. Let $c_k(s)$ denote the sum of all fixed products that contribute to $C_k$. Note that $c_k = C_k$ for $k \geq N - m$. Then

$$\min_{\text{free}} |C_k| \geq \max(b_k, |c_k(s)| - f_k),$$

where

$$f_k = (N - k) - 2\max(m, 0) - \max(k + 2m, 0)$$

denotes the number of free products in $C_k$, and $b_k$ is given by \cite{4}. The reasoning behind \cite{12} is that the sum $c_k$ of fixed products may be offset by the sum of free products, which is no greater than $f_k$. If $|c_k(s)| > f_k$ then $|c_k(s)| - f_k$ is a lower bound for $|C_k|$. If $|c_k(s)| \leq f_k$, this bound is useless and we have to resort to the trivial lower bound $|C_k(s)| \geq b_k$. The bound \cite{12}
was used by Prestwich to prune parts of the search space in a local search algorithm for the LABS problem [28].

For his recent branch&bound algorithm for LABS, Prestwich [29] improved that bound by taking into account some of the interactions between fixed and free spins. Suppose that \( s_i \) is a free spin while \( s_{i-k} \) and \( s_{i+k} \) are fixed. If \( s_{i-k} \neq s_{i+k} \), the contributions

\[
s_{i-k}s_i + s_is_{i+k} = s_i(s_{i-k} + s_{i+k})
\]

(14)
of \( s_i \) to \( C_k \) are zero, no matter what the value of \( s_i \) is. For each such cancellation, the number \( f_k \) in (12) can be decreased by two. For \( s_{i-k} = s_{i+k} \), the contribution of the term (14) is ±2, a situation referred to as reinforcement by Prestwich. Now, if all free contributions to \( C_k \) are either cancellations or reinforcements, then \( f_k \) must be even. If the sum of the fixed contributions \( c_k \) is also even and \( c_k \mod 4 \neq f_k \mod 4 \), we can set \( b_k = 2 \) in (12). With this bound, Prestwich reports a running time that scales like \( O(1.80^N) \). Since Prestwich didn’t parallelize his algorithm, this estimate was based on enumerations only up to \( N \leq 44 \).

3.1.3. The Wiggenbrock bound. A different bound was used by Wiggenbrock in his branch&bound algorithm [30]. Flipping a spin changes the sum \( C_k + C_{N-k} \) by ±4 because every spin occurs twice in that sum. Taking the all +1 configuration as a reference, we get

\[
(N - C_{N-k}) \equiv C_k \pmod{4}.
\]

(15)

For \( k \geq N - m \), the \( C_k \) are completely fixed. For other values of \( k \), the correlations can be bounded by

\[
|C_k| \geq \begin{cases} 
|N - C_{N-k} \mod 4| & \text{if } k \leq m, \\
 b_k & \text{if } m < k < N - m,
\end{cases}
\]

(16)

where we assumed the residue system \( \{-1, 0, 1, 2\} \) for the mod 4 operation.

The Wiggenbrock bound seems to be weak since it bounds \( |C_k| \) by small numbers 0, 1, 2 only. Yet it is surprisingly efficient: Wiggenbrock reported a running time of \( O(1.79^N) \), slightly better than the scaling of Prestwich’s bound. Using a parallelized implementation and running it on 18 GPUs, Wiggenbrock solved the LABS problem for \( N \leq 64 \) [30].

3.1.4. The combined bound. High up in the search tree, where \( m \) is small, the contributions of the free products overcompensate the fixed contributions and the Prestwich bound (12) reduces to \( b_k \). The Wiggenbrock bound (16) provides a better bound in exactly these situations. The fact that it yields such a good running time indicates that even this weak bound is efficient because it applies high up in the search tree: a branch, that can be pruned at this level, is usually very large. The Prestwich bound with the free products applies for larger values of \( m \), on the other hand. An obvious idea is to combine these complimentary bounds and use

\[
|C_k| \geq \begin{cases} 
 \max \left( |N - C_{N-k} \mod 4|, |c_k(s)| - f_k \right) & \text{if } k \leq m, \\
 \max(b_k, |c_k(s)| - f_k) & \text{if } m < k < N - m,
\end{cases}
\]

(17)
as a bound.

When we measure the number of recursive calls (i.e. the number of nodes visited in the search tree) and the CPU time per call (Figure 2), we find that the running time of the branch&bound algorithm with the combined bound (17) scales like \( \Theta(N 1.729^N) \).
Figure 2. Branch&bound algorithm with the combined bound (17) (circles) and with the tight bound (21) (triangles). The number of recursive calls (top) scales like $\Theta(b^N)$. A numerical fit to the existing data yields $b = 1.729$ (solid line) for the combined bound and $b = 1.727$ (dashed line) for the tight bound, but this small difference is caused by a non-exponential reduction of the number of calls, see Figure 3. The CPU time per call (bottom) is linear in $N$ for both bounds.
Let at least one spin is free. Let the free contribution be $u_k$. We call every sum in parentheses a chain.

We will show below that there exist easy to compute integers $U_k^{\text{min}}$ and $U_k^{\text{max}}$ such that the free contribution $u_k$ can take on all values in $C_k = c_k + u_k$, \begin{equation} \label{eq:ck}
C_k = c_k + u_k, \end{equation}
where $c_k$ is the sum of all fixed terms $s_is_{i+k}$ (as above) and $u_k$ sums up all terms in which at least one spin is free. Let \begin{equation} \label{eq:gw}
g_k = \begin{cases} 4 & \text{if } k \leq m, \\ 2 & \text{otherwise}. \end{cases}
\end{equation}
We will show below that there exist easy to compute integers $U_k^{\text{min}}$ and $U_k^{\text{max}}$ such that the free contribution $u_k$ can take on all values in \begin{equation} \label{eq:uk}
\{U_k^{\text{min}}, U_k^{\text{min}} + g_k, U_k^{\text{min}} + 2g_k, \ldots, U_k^{\text{max}} - g_k, U_k^{\text{max}}\}
\end{equation}
All we need to know are the values of $c_k$, $U_k^{\text{min}}$ and $U_k^{\text{max}}$ to compute
\begin{equation} \label{eq:ckm}
\min |C_k| = \begin{cases} c_k + U_k^{\text{min}} & \text{if } -c_k \leq U_k^{\text{min}}, \\ c_k + U_k^{\text{max}} & \text{if } -c_k \geq U_k^{\text{max}}, \\ \left|(-c_k - U_k^{\text{min}}) \mod g_k\right| & \text{otherwise}, \end{cases}
\end{equation}
and then $E_k^{\text{min}} = \sum (\min |C_k|)^2$.

To prove \eqref{eq:gw} and \eqref{eq:uk}, we rearrange the sum \eqref{eq:ck} for $C_k$ a little bit. For $C_3$ and $N = 12$, for example, we can write \begin{align*}
C_3 &= s_1s_4 + s_1s_4 + s_1s_4 + s_1s_4 + s_1s_4 + s_1s_4 + s_1s_4 + s_1s_4 + s_1s_4 \\
&= (s_1s_4 + s_4s_7 + s_7s_{10}) + (s_2s_5 + s_5s_8 + s_8s_{11}) + (s_3s_6 + s_6s_9 + s_9s_{12})).
\end{align*}
We call every sum in parentheses a chain. For general values of $k$ and $N$ we write
\begin{equation} \label{eq:ck}
C_k = \sum_{j=1}^{k} \sum_{q=1}^{N-k} s_{j+(q-1)k}s_{j+qk}. \end{equation}
The chains are the sums over $q$. For $k < N - m$, each chain contains a subchain of free terms
\begin{equation} \label{eq:chain}
s_a s_{a+k} + s_{a+k}s_{a+2k} + \cdots + s_{b-k}s_b
\end{equation}
where only the spins $s_a$ and $s_b$ may be fixed. We refer to these subchains as free chains. The sum of all free chains equals $u_k$.

Let us first prove the “granularity” \eqref{eq:gw}. If both spins $s_a$ and $s_b$ are fixed, then every free spin appears exactly in two terms, and flipping any free spin changes the sum \eqref{eq:chain} by $\pm 4$. Hence the granularity $g_k$ is $4$ and only if all contributing free chains have both $s_a$ and $s_b$ fixed, and $2$ otherwise.

Now $s_a$ can be free and the leftmost member of a free chain if and only if $a > m$ and if it has no left partner, i.e. if $a - k \leq 0$. Together, both conditions imply $k > m$. Hence by argumentum e contrario, $k \leq m$ implies that $s_a$ is fixed and, by similar reasoning, also that that $s_b$ is fixed. This proves that $g_k = 4$ for $k \leq m$.

If $k > m$, we only need to find a single free chain that starts with a free spin. Consider the spin $s_{m+1}$: It is free and it has no left neighbor. Hence it is the leftmost spin of a free chain that contributes to $u_k$. Therefore $g_k = 2$ for $k > m$. Note that for $k > m$ there can be free chains...
with both $s_a$ and $s_b$ fixed. All we have proven is that for $k > m$ this can’t happen for all free chains.

Now we will prove (26). Let $n$ denote the number of terms $s_j s_{j+k}$ in a free chain (23), and let $u$ denote its value. If $s_a$ or $s_b$ (or both) are free, then $n$ can take on all values between $-n$ and $n$ with granularity $2$:

$$u \in [-n, -n+2, \ldots, n-2, n] \quad (s_a \text{ or } s_b \text{ free}).$$

(24)

If both spins $s_a$ and $s_b$ are fixed, the granularity is 4 and the range of values varies with $s_a$, $s_b$ and the parity of $n$ according to

$$u \in \begin{cases} [-n, \ldots, n] & \text{if } s_a = s_b \text{ and } n \text{ even,} \\ [-n-2, \ldots, n] & \text{if } s_a = s_b \text{ and } n \text{ odd,} \\ [-n-2, \ldots, n-2] & \text{if } s_a \neq s_b \text{ and } n \text{ even,} \\ [-n, \ldots, n-2] & \text{if } s_a \neq s_b \text{ and } n \text{ odd.} \end{cases}$$

(25)

This can be proven by induction over $n$. For $n$ odd, the base case is $n = 3$, i.e.

$$u = s_a s_{a+k} + s_a k s_{b-k} + s_b k s_b .$$

The value of $u$ is maximized by setting the free spins $s_{a+k} = s_a$ and $s_{b-k} = s_b$. If $s_a = s_b$, the center term is 1 and $u_{max} = 3$. For $s_a \neq s_b$, the center term is $-1$ and $u_{max} = 1$. The value of $u$ is minimized by setting $s_{a+k} = -s_a$ and $s_{b-k} = -s_b$. If $s_a = s_b$, the center term is 1 and $u_{min} = -1$. If $s_a \neq s_b$, the center term is $-1$ and $u_{min} = -3$. Now let us assume that (25) holds for some odd $n \geq 3$ and consider a free chain

$$u = s_a s_{a+k} + s_{a+k} s_{b-k} + \cdots + s_{b-2k} s_{b-k} + s_b k s_b$$

with $n + 2$ terms. To maximize $u$, we set $s_{a+k} = s_a$ and $s_{b-k} = s_b$, and the remaining free chain has $n$ terms. Applying (25), we get $u_{max} = n + 2$ if $s_a = s_b$ and $u_{max} = n$ if $s_a \neq s_b$. The induction step for $u_{min}$ is obvious.

Since the proof for even $n$ is very similar, it is omitted here. We only mention that the base case ($n = 2$) corresponds to the “cancellation” and “reinforcement” used by Prestwich to improve the bound (12).

Now (24) and (25) tell us how to compute $u_{min}$ and $u_{max}$ for each individual free chain. The corresponding values $U_{k}^{min}$ and $U_{k}^{max}$ are obtained by summing over all free chains that contribute to $u_k$.

Every branch of the search tree that can be pruned according to the combined bound (17) (or any other relaxation of (9)) is also pruned by the tight bound (21), but the tight bound allows us to prune additional branches. Hence the number of recursive calls with the tight bound can not be larger than the number of calls with any other bound based on (9). What we observe is that for $N \leq 66$ the number of calls for the tight bound is in fact strictly smaller than that for the combined bound. A numerical fit to the existing data yields a scaling of $\Theta(1.727^N)$ for the tight bound, compared to $\Theta(1.729^N)$ for the combined bound, see Figure 2. This difference is too small to tell whether the tight bound actually provides an exponential speedup or not. In fact, if one looks at the ratio of the number of calls for the combined bound divided by the number of calls for the tight bound, one observes that the speedup factor grows linearly with $N$, not exponentially (Figure 3). Since the time per call scales linearly for both bounds (Figure 2 bottom), a reduction of the number of calls that grows with $N$ implies that the tight bound will asymptotically outperform the combined bound.

For the values of $N$ considered in this paper, however, the absolute computational costs per call matter. And here the simpler combined bound (17) is faster, see Figure 2 (bottom). If we extrapolate the number of calls and the time per call to $N = 66$, we get a running
time of roughly 12600 CPU days for the combined bound but 14300 CPU days for the tight bound. This is why we used the weaker combined bound for all the new solutions (exact and skewsymmetric) reported in this paper. Note that the time per call depends considerably on the implementation. It might well be possible to implement the tight bound such that it outperforms the combined bound already for the values of $N$ considered here. In any case, the measured running times illustrate that we need to parallelize the computation if we don’t want to wait 35 years for the $N = 66$ LABS solution.

3.2. Symmetry and Parallelization

The correlations $C_k$ are unchanged when the sequence is complemented or reversed. When alternate elements of the sequence are complemented, the even-indexed correlations are not affected, the odd-indexed correlations only change sign. Hence, with the exception of a small number of symmetric sequences, the $2^N$ sequences will come in classes of eight which are equivalent. The total number of nonequivalent sequences is slightly larger than $2^{N-3}$.

The $m$ left- and $m$ rightmost elements of the sequence can be used to parameterize the symmetry classes. The total number $c(m)$ of symmetry classes that can be distinguished by $m$ left- and $m$ right-border elements reads

$$c(m) = 2^{2m-3} + 2^{m-2} + (N \mod 2).$$

(26)

We derive this formula in Appendix A, where we also describe how to compute the values of
the $2m$ boundary spins that represent each symmetry class.

The symmetry classes can be enumerated independently, which allows us to parallelize the computation. For our largest system ($N = 66$) we used $c(m = 10) = 131328$ symmetry classes that we searched in parallel on various computers with number of computing cores ranging from 8 to 5700. In principle, the branch&bound algorithm requires some communication between the parallel tasks since every task should know the lowest energy found so far by other tasks to compare it to the bound. We avoid this communication completely by using a static value for this reference energy: the lowest energy found by heuristic searches. In all cases we considered, this value turned out to be the true minimum energy.

4. Results and Conclusions

We have used the branch&bound algorithm with the combined bound to compute all sequences with maximum merit factor for $N \leq 66$, see Tables 1 and 2. The previous record was $N \leq 64$, obtained with the Wiggenbrock bound [16] and using 18 GPUs [30]. For the performance measurements for $40 \leq N \leq 64$ shown in Figure 2 we have used a Linux cluster with a collection of Intel® Xeon® CPUs: $10 \times$ E5-2630 (at 2.30 Ghz), $10 \times$ E5-2630 v2 (at 2.60 Ghz) and $2 \times$ E5-1620 (at 3.60 Ghz) with a total of 248 (virtual) cores. On this machine, the computation for $N = 64$ took about a week (wallclock time). As one can see in Figure 2, the solution of $N = 63$ and $N = 64$ involves a surprisingly low number of calls and took therefore less time than actually expected.

Note that with our algorithm systems of size $N \leq 43$ can be solved in less than an hour on a laptop.

For $N = 65$ and $N = 66$ we used a variety of computing machinery that makes an accurate determination of “single CPU time” impossible. For $N = 65$ and 66, the equivalent wallclock time on our benchmark cluster is roughly 32 and 55 days.

Tables 1 and 2 show all sequences (except those related by symmetries) with maximum merit factors up to $N = 66$ in run-length encoding, i.e. the digits specify the length of runs of equal spins. We use $a = 10$, $b = 11$ etc. for runs of spins that are longer than 9.

We have used our branch&bound algorithm also to find all skewsymmetric sequences with maximum merit factor up to $N = 119$. The previous record was $N \leq 89$ [29]. Table 3 shows the skewsymmetric sequences with maximum merit factor as far as they are not listed in Tables 1 and 2. Skewsymmetric merit factors marked with $\star$ are known to be not maximal.

We know this either from exhaustive enumerations (for $N \leq 65$) or from heuristic searches that have yielded non skewsymmetric sequences with larger merit factors.

Figure 4 shows the ratio of the maximum merit factors of skewsymmetric and general sequences for $N \leq 119$. In 20 out of 58 cases the skewsymmetric subset does not contain a maximum merit factor sequence. Note that the values of $F_N$ for $N > 66$ are from heuristic searches, but we believe that these values are the true maximum merit factors. But strictly speaking, the gray symbols in Figure 4 are only upper bounds for the ratio $F^\text{skew}_N / F_N$.

The available data seems to indicate that roughly two thirds of all odd values of $N$ have skewsymmetric maximum merit factor sequences. Figure 4 also suggests that

$$\liminf_{N \to \infty} \frac{F^\text{skew}_N}{F_N} = 1.$$  (27)

We think that the branch&bound approach based on the relaxation (9) can be used to solve the LABS problem for $N > 66$ by devoting more compute cores and more CPU time. Improving the implementation to reduce the constant factor in the $\Theta(N^{b_N})$ scaling can also
**Low Autocorrelation Binary Sequences**

| \( N \) | \( E \) | \( F_N \) | sequences | skew | \( N \) | \( E \) | \( F_N \) | sequences | skew |
|------|------|-------|----------|------|------|------|-------|----------|------|
| 3    | 1    | 4.500 | 21       | ×    | 19   | 29   | 6.224 | 411142212 |      |
| 4    | 2    | 4.000 | 112      |      | 20   | 26   | 7.692 | 5113112321 |      |
| 5    | 2    | 6.250 | 311      | ×    | 21   | 26   | 8.481 | 27221111121 | ×    |
| 6    | 7    | 2.571 | 141      |      | 22   | 39   | 6.205 | 512211111233 |      |
|      |      |       | 312      |      |      | 113   | 23    | 47   | 5.628 | 212121111632 |      |
| 7    | 3    | 8.167 | 1123     | ×    |      |       |       |       |       | 8321111221 |      |
| 8    | 8    | 4.000 | 32111    |      | 24   | 36   | 8.000 | 223611111212 |      |
| 9    | 12   | 3.375 | 311121   |      | 25   | 36   | 8.681 | 337111112212 |      |
|      |      |       | 42111    | ×    | 26   | 45   | 7.511 | 2121211111632 |      |
|      |      |       | 32211    | ×    |      |       |       |       |       | 532111111211 |      |
|      |      |       | 31122    |      |      |       |       |       |       | 532611111211 |      |
| 10   | 13   | 3.846 | 42211    |      | 27   | 37   | 9.851 | 3431311212 | ×    |
|      |      |       | 52111    |      | 28   | 50   | 7.840 | 3431311212 |      |
|      |      |       | 311122   |      | 29   | 62   | 6.782 | 2121211313431 | ×    |
|      |      |       | 41122    |      |      |       |       |       |       | 3237111112121 | ×    |
|      |      |       | 31121    |      | 30   | 59   | 7.627 | 55121111113231 |      |
| 11   | 5    | 12.100| 112133   | ×    |      |       |       |       |       | 4612111113231 |      |
| 12   | 10   | 7.200 | 4221111  |      | 31   | 67   | 7.172 | 73322121111122221 |      |
|      |      |       | 4111221  |      | 32   | 64   | 8.000 | 71112111133221221 |      |
| 13   | 6    | 14.083| 5221111  | ×    | 33   | 64   | 8.508 | 74211211111122221 |      |
| 14   | 19   | 5.158 | 41112221 | ×    | 34   | 65   | 8.892 | 84211211111122221 |      |
|      |      |       | 6221111  |      | 35   | 73   | 8.390 | 71221211111111332 |      |
|      |      |       | 5222111  |      | 36   | 82   | 7.902 | 363231113121111121 |      |
|      |      |       | 33111212 |      | 37   | 86   | 7.959 | 8442112111111122221 |      |
|      |      |       | 41112222 |      | 38   | 87   | 8.299 | 84421121111111122221 |      |
|      |      |       | 42211112 |      | 39   | 99   | 7.682 | 82121121234321111111 | ×    |
|      |      |       | 5221112  |      |      |       |       |       |       | 23241171111141122 | ×    |
|      |      |       | 5311121  |      | 40   | 108  | 7.407 | 44412112311213131 |      |
| 15   | 15   | 7.500 | 522211111| ×    | 41   | 108  | 7.782 | 3431111112222812121 | ×    |
|      |      |       | 33131211 | ×    | 42   | 101  | 8.733 | 31313134313411121122 |      |
| 16   | 24   | 5.333 | 225111121|      | 43   | 109  | 8.482 | 11324321111172112223 | ×    |
|      |      |       | 6322111  |      | 44   | 122  | 7.934 | 52531131112221111121 |      |
|      |      |       | 313311211|      | 45   | 118  | 8.581 | 82121121232343211111 |      |
|      |      |       | 2131441 |      | 46   | 131  | 8.076 | 8234312312112211111111 |      |
| 17   | 32   | 4.516 | 252111121| ×    |      |       |       |       |       | 82121121234321111111 | ×    |
|      |      |       | 4412131  |      |      |       |       |       |       | 73235111123122112121 | ×    |
|      |      |       | 4221211112| 47  | 135  | 8.181 | 923431231211221111111 |      |
|      |      |       | 3611221  |      |      |       |       |       |       | 42942222121111112211 | ×    |
|      |      |       | 2122411112|    |      |       |       |       |       | 41112114131312421242 | ×    |
|      |      |       | 211213132 |    |      |       |       |       |       | 383422132211211111112 | ×    |
| 18   | 25   | 6.480 | 441112222 |      |      |       |       |       |       | 2363316111312111111121 | ×    |
|      |      |       | 51121322 |      |      |       |       |       |       | 21a121121324211111111231 | ×    |

Table 1. All optimal low autocorrelation binary sequences for \( N \leq 47 \) modulo symmetries.
Table 2. All optimal low autocorrelation binary sequences for $48 \leq N \leq 66$ modulo symmetries.

| $N$ | $E$  | $F_N$ | sequences | skew |
|-----|------|-------|-----------|------|
| 48  | 140  | 8.229 | 3111111832143212221121121 |
| 49  | 136  | 8.827 | 2151311224112241141 | × |
| 50  | 153  | 8.170 | 2151311224112241141 | × |
| 51  | 153  | 8.500 | 23432111413131162121121 | × |
| 52  | 166  | 8.145 | 511612121111131223123332 |
| 53  | 170  | 8.262 | 4511311332513122211211121 |
| 54  | 175  | 8.331 | 3562251412121122211111121 |
| 55  | 171  | 8.845 | 92121232114321233211111111 |
| 56  | 192  | 8.167 | 7612231123241111321121221 |
| 57  | 188  | 8.641 | 3323263111112712111212221 |
| 58  | 197  | 8.538 | 72121211412121233211111111 |
| 59  | 205  | 8.490 | 77214224211223112112111111 |
| 60  | 218  | 8.257 | 76111214111113124211322211 |
| 61  | 226  | 8.323 | 314162231111131113112562121 |
| 62  | 235  | 8.179 | 3232321111711154112151122212 |
| 63  | 207  | 9.587 | 212121271111151112143111422321 |
| 64  | 208  | 9.846 | 21212127111115111214311142232 |
| 65  | 240  | 8.802 | 32322411341211151111721212212 |
| 66  | 257  | 8.475 | 2112111211222b222111111112224542 |

help. Solving systems significantly larger than $N = 66$, however, requires a stronger bound than (9), i.e., a bound that takes into account the fact that the $C_k$ are not independent. Or a completely new approach other than branch&bound.

Appendix A. Symmetry

To find the exact number of symmetry classes in the LABS problem we need some group theory. The operators $R$ (reverse), $C$ (complement) and $A$ (alternate complement) act on the sequences, leaving the energy invariant. Together with the identity operator $I$ these operators generate a group $G$ of order 8. The structure of $G$ depends on $N$ being odd or even.

For $N$ even, $G = G_e$ is non-abelian and isomorphic to the dihedral group $D_4$, the symmetry group of a square plate, generated by a 90 degree rotation and a flip. The elements of $G_e$ are $\{I, R, C, A, RA, AR, RC, AC\}$. The group elements act on a sequence $s$. Let

$$G(s) := \{g(s) : g \in G\}$$

(A.1)

denote the orbit of $s$, i.e. the set of all spin sequences that can be generated from $s$ by application of the group elements. The length of the orbit is $|G(s)|$. The orbits partition the set of all spin sequences in the symmetry classes we want to count. If all orbits were of
Table 3. All optimal skewsymmetric low autocorrelation binary sequences for $N \leq 119$ as far as they are not listed in Table 1 or 2. Merit factors marked with ⋆ are known to be not maximal, either from exhaustive enumeration (for $N \leq 65$) or from heuristic searches (for $N \geq 67$).
Figure 4. Ratio of maximum merit factors: skewsymmetric $F_{N}^{\text{skew}}$ versus general $F_N$. Black symbols are exact, gray symbols are based on lower bounds for $F_N$, which are believed to be exact.

length 8, we would have $2^{N-3}$ symmetry classes. Unfortunately there are orbits of smaller length, to wit

$$G(++++) = \{++++, −−−−, −+−+, +−+−\}.$$  

The true number $c$ of orbits is given by Burnside’s Lemma,

$$c = \frac{1}{|G|} \sum_{g \in G} |\text{Fix}(g)|,$$  \hspace{1cm} (A.2)

where $\text{Fix}(g) = \{s : g(s) = s\}$ denotes the set of all sequences $s$ that are fixed points of $g$. The group elements $A, C, AR, RA$ and $AC$ can’t fix a sequence, but $R$ and $RC$ can. Sequences with $s_j = s_{N+1−j}$ are fixed by $R$, sequences with $s_j = −s_{N+1−j}$ are fixed by $RC$, and there are $2^{N/2}$ sequences of each type. Hence

$$c_{N\text{even}} = \frac{1}{8} (|\text{Fix}(I)| + |\text{Fix}(R)| + |\text{Fix}(RC)|) = 2^{N−3} + 2^{N/2−2}. \hspace{1cm} (A.3)$$

For $N$ odd, the group $G = G_o$ is again of order 8, but this time it is abelian. Group elements are $G_o = \{I, R, C, A, RC, RA, CA, RCA\}$, and $g^2 = I$ for all $g \in G_o$. $G_o$ is isomorphic to the reflection-symmetry group of the cube. If $(N−1)/2$ is odd, only $R$, $RA$ and $I$ have fixed points. Sequences that are fixed by $R$ have $s_j = s_{N+1−j}$ with arbitrary center spin $s_{(N+1)/2}$. There are $2 \cdot 2^{(N−1)/2}$ such sequences. The same number of sequences are fixed by $RA$. Hence

$$c_{N\text{odd}} = \frac{1}{8} (|\text{Fix}(I)| + |\text{Fix}(R)| + |\text{Fix}(RA)|) = 2^{N−3} + 2^{(N−1)/2−1}. \hspace{1cm} (A.4)$$
If $(N - 1)/2$ is even, only $I$, $R$ and $RCA$ have fixed points, and their numbers are the same as in (A.4). Combining (A.3) and (A.4) provides us with

\[ c_N = 2^{N-3} + 2^{(N-1)/2-2+(N \mod 2)}. \]

(A.5)

This is the total number of symmetry classes if we consider all elements of the sequence. If we only consider the $m$ leftmost and $m$ rightmost elements, the arguments are similar. For $N$ even, the symmetry group $G_e = \{I, R, C, A, RA, AR, RC, AC\}$ acts only on the $2m$ elements, and only $I$, $R$ and $RC$ have fixed points. Hence

\[ c(m) = 2^{2m-3} + 2^{m-2} \quad N \text{ even}. \]

(A.6)

For $N$ odd, the symmetry group is again $G_o = \{I, R, C, A, RC, RA, CA, RCA\}$, but this time $I$, $R$, $RC$, $RA$ and $RCA$ have fixed points:

\[ c(m) = 2^{2m-3} + 2^{m-1} \quad N \text{ odd}. \]

(A.7)

Combining (A.6) and (A.7) provides us with (26).

The $c(m)$ symmetry classes can be uniquely parameterized by the values of the $2m$ boundary spins. Consider the list of all $2^{2m}$ possible configurations of the boundary spins. For each such configuration compute $G(s) = G_e(s)$ (for $N$ even) or $G(s) = G_o(s)$ (for $N$ odd). If $s$ is not the lexicographically smallest element in $G(s)$, remove it from the list. The remaining elements are a unique representation of the symmetry classes.

References

[1] Marcel Jules Edouard Golay. A class of finite binary sequences with alternate autocorrelation values equal to zero. IEEE Transactions on Information Theory, IT-18:449–450, 1972.
[2] G.F.M. Beenker, T.A.C.M. Claasen, and P.W.C. Hermens. Binary Sequences With a Maximally Flat Amplitude Spectrum. Philips Journal of Research, 40:289–304, 1985.
[3] I. A. Pasha, P. S. Moharir, and N. Sudarshan Rao. Bi-alphabetic pulse compression radar signal design. Sadhana, 25:481–488, 2000.
[4] I. Shapiro, G.H. Pettengill, M.E. Ash, M.L. Stone, W.B. Smith, R.P. Ingalls, and R.A. Brockelman. Fourth test of general relativity. Physical Review Letters, 20:1265–1269, 1968.
[5] John Edensor Littlewood. Some problems in real and complex analysis. D C Heath & Co, Lexington, MA, 1968.
[6] Peter Borwein. Computational Excursions in Analysis and Number Theory. Springer-Verlag, New York, 2002.
[7] Jakob Bernasconi. Low autocorrelation binary sequences: statistical mechanics and configuration space analysis. Journal de Physique, 48(4):559–567, 1987.
[8] W. Krauth and M. Mézard. Aging without disorder on long time scales. Z. Physik B, 97:127–131, 1995.
[9] J.P. Bouchaud and M. Mézard. Self induced quenched disorder: a model for the glass transition. Journal de Physique I, 4:1109–1114, 1994.
[10] Enzo Marinari, Giorgio Parisi, and Felix Ritort. Replica field theory for determistic models: I. binary sequences with low autocorrelation. Journal of Physics A: Mathematical and General, 27:7615–7645, 1994.
[11] Jonathan Jedwab. A survey of the merit factor problem for binary sequences. In Tor Helleseth et al., editor, Sequences and Their Applications — Proceedings of SETA 2004, volume 3486 of Lecture Notes in Computer Science, pages 30–55. Springer-Verlag, 2005.
[12] Tom Hoholdt. The merit factor problem for binary sequences. In Marc P.C. Fossorier, Hideki Imai, Shu Lin, and Alain Poli, editors, Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, volume 3857 of Lecture Notes in Computer Science, pages 51–59. Springer-Verlag, Berlin Heidelberg, 2006.
[13] R.H. Barker. Group synchronizing of binary digital systems. In Willis Jackson, editor, Communication Theory, pages 273–287. Butterworths Publications Ltd., London, 1953.
[14] R. Turyn and J. Storer. On binary sequences. Proceedings of the American Mathematical Society, 12:394–399, 1961.
[15] Kai-Uwe Schmidt and Jürgen Willms. Barker sequences of odd length. Designs, Codes and Cryptography, pages 1–6, 2015.
[16] Ka Hin Leung and Bernhardt Schmidt. New restrictions on possible orders of circulant Hadamard matrices. Designs, Codes and Cryptography, 64(1):143–151, 2012.
[17] Marcel Jules Edouard Golay. The merit factor of long low autocorrelation binary sequences. IEEE Transactions on Information Theory, IT-28:543, 1982.
[18] Donald J. Newmann and J.S. Byrnes. The $l^4$ norm of a polynomial with coefficients ±1. *American Mathematical Monthly*, 97(1):42–45, 1990.

[19] Jonathan Jedwab, Daniel J. Katz, and Kai-Uwe Schmidt. Advances in the merit factor problem for binary sequences. *Journal of Combinatorial Theory (A)*, 120:882–906, 2013.

[20] Jonathan Jedwab, Daniel J. Katz, and Kai-Uwe Schmidt. Littlewood polynomials with small $l^4$ norm. *Advances in Mathematics*, 241:127–136, 2013.

[21] Borko Bošković, Franc Brglez, and Janez Brest. Low-autocorrelation binary sequences: on the performance of memetic-tabu and self-avoiding walk solvers. arXiv:1406.5301.

[22] Carla Savage. A survey of combinatorial Gray codes. *SIAM Reviews*, 39(4):605–629, 1997.

[23] C. de Groot, D. Würtz, and K. H. Hoffmann. Low autocorrelation binary sequences: exact enumeration and optimization by evolutionary strategies. *Optimization*, 23:369–384, 1992.

[24] CSPLib: A problem library for constraints. [www.csplib.org](http://www.csplib.org).

[25] Cristopher Moore and Stephan Mertens. *The Nature of Computation*. Oxford University Press, 2011. [www.nature-of-computation.org](http://www.nature-of-computation.org).

[26] Stephan Mertens. Exhaustive search for low-autocorrelation binary sequences. *J. Phys. A*, 29:L473–L481, 1996.

[27] Heiko Bauke and Stephan Mertens. Ground states of the Bernasconi model with open boundary conditions. [http://www.ovgu.de/mertens/research/labs/open.dat](http://www.ovgu.de/mertens/research/labs/open.dat) 2004.

[28] Steven David Prestwich. Exploiting relaxation in local search for LABS. *Annals of Operations Research*, 156(1):129–141, 2007.

[29] Steven David Prestwich. Improved branch-and-bound for low autocorrelation binary sequences. [http://arxiv.org/abs/1305.6187](http://arxiv.org/abs/1305.6187) July 2013.

[30] Jens Wiggenbrock. Parallele Optimierungsstrategien des LABS-Problems in einem GPU-Grid. Bachelor’s thesis, Fachhochschule Südwestfalen, 2010.