# An Elementary Proof of a $3n - o(n)$ Lower Bound on the Circuit Complexity of Affine Dispersers[⋆]

Evgeny Demenkov[1] and Alexander S. Kulikov[2]

[1] St. Petersburg State University
[2] St. Petersburg Department of Steklov Institute of Mathematics

**Abstract.** A Boolean function $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$ is called an affine disperser of dimension $d$, if $f$ is not constant on any affine subspace of $\mathbb{F}_2^n$ of dimension at least $d$. Recently Ben-Sasson and Kopparty gave an explicit construction of an affine disperser for sublinear $d$. The main motivation for studying such functions comes from extracting randomness from structured sources of imperfect randomness. In this paper, we show another application: we give a very simple proof of a $3n - o(n)$ lower bound on the circuit complexity (over the full binary basis) of affine dispersers for sublinear dimension. The same lower bound $3n - o(n)$ (but for a completely different function) was given by Blum in 1984 and is still the best known.

The main technique is to substitute variables by linear functions. This way the function is restricted to an affine subspace of $\mathbb{F}_2^n$. An affine disperser for sublinear dimension then guarantees that one can make $n - o(n)$ such substitutions before the function degenerates. It remains to show that each such substitution eliminates at least 3 gates from a circuit.

## 1 Introduction

Proving lower bounds on the circuit complexity of explicitly defined Boolean functions is one of the most famous and difficult problems in theoretical computer science. Already in 1949 Shannon [1] showed by a counting argument that almost all Boolean functions have circuits of size $\Omega(2^n/n)$ only. Still, we have no example of an explicit function requiring super-linear circuit size. Moreover, only a few proofs of linear lower bounds are known. We review some of them is Sect. 3. The best lower bound $3n - o(n)$ for the basis $B_2$ was proved by Blum in 1984 [2], the current record lower bound $5n - o(n)$ for the basis $U_2 = B_2 \setminus \{\oplus, \equiv\}$ was given in 2002 by Iwama, Lachish, Morizumi, and Raz [3].

⋆ Research is partially supported by Federal Target Program "Scientific and scientific-pedagogical personnel of the innovative Russia" 2009–2013, Russian Foundation for Basic Research, RAS Program for Fundamental Research, Grant of the President of Russian Federation (NSh-5282.2010.1), and PDMI Computer Science Club scholarship.

All bounds mentioned above are proved by the gate elimination method. The main idea of this method is the following. One considers a Boolean function on $n$ variables from a certain class of functions and shows that for any circuit computing this function setting some variables to constants yields a sub-function of the same type and eliminates several gates. Usually, a gate is eliminated just because one of its inputs becomes a constant. By induction, one concludes that the original circuit must have many gates. Though this method is essentially the only known method for proving non-trivial lower bounds for general circuit complexity, as many authors note it is unlikely that it will allow to prove non-linear bounds.

In this paper, we prove a $3n - O(d)$ lower bound on the circuit complexity of a Boolean function $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$ that is not constant on any affine subspace of $\mathbb{F}_2^n$ of dimension at least $d$. Such functions are called affine dispersers for dimension $d$. The proof of a lower bound is much simpler than the proof of the currently strongest lower bound $3n - o(n)$ given by Blum in 1984 [2]. However, it is not easy to construct explicitly an affine disperser for small $d$. Only recently Ben-Sasson and Kopparty [4] presented a construction for sublinear $d = o(n)$ (namely, $d = \Theta(n^{4/5})$).

The main idea of the proof is as follows. Consider an affine disperser $f$ for dimension $d$. We know that $f$ is not constant on any affine subspace of $\mathbb{F}_2^n$ of dimension at least $d$. Hence for any $I_1, \ldots, I_{n-d} \subseteq \{1, \ldots, n\}$ and $c_1, \ldots, c_{n-d} \in \mathbb{F}_2$, $f$ is not constant on affine subspace

$$\{x \in \mathbb{F}_2^n \mid \bigoplus_{i \in I_k} x_i = c_k, \text{ for all } 1 \le k \le n-d\}$$

of dimension at least $d$. We consequently find substitutions of the form $x_{i_k} = \bigoplus_{i \in I_k} x_i \oplus c_k$ so that at least 3 gates are eliminated under each of them from the current circuit. This way we eliminate at least $3n - O(d)$ gates.

To find a substitution under which at least 3 gates are eliminated we just take the topologically first non-linear gate $R$ (i.e., a gate that does not compute a function of the form $\bigoplus_{i \in I} x_i \oplus c$, for $I \subseteq \{1, 2, \ldots, n\}$, $c \in \mathbb{F}_2$) of a circuit. Since it is the first such gate, both its inputs $P$ and $Q$ are linear functions. By an appropriate substitution, we make $P$ constant which also makes $R$ constant. This kills $P$, $R$ and all the successors of $R$, i.e., at least 3 gates in total. In the example given in Fig. 1, one can make a substitution $x_1 = x_2 \oplus 1$. Then $P$ evaluates to 0, $R$ also evaluates to 0 and $T$ is also eliminated. The formal proof is given in Section 4.

Similar ideas (substituting variables by linear functions) were used by Boyar and Peralta [5] for proving lower bounds on the multiplicative complexity of Boolean functions. The multiplicative complexity of a Boolean function is defined as the smallest number of $\wedge$ gates in a circuit over $\{\wedge, \oplus, 1\}$ computing this function. As with the circuit complexity, it is known [6] that the multiplicative complexity of almost all Boolean functions is about $2^{n/2}$, while the best known lower bound for an explicit function is $n - 1$. As an easy consequence we obtain a lower bound $n - d - 1$ on the multiplicative complexity of an affine disperser for dimension $d$.

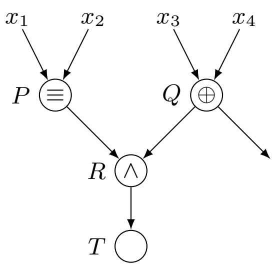


**Fig. 1.** Substitution $x_1 = x_2 \oplus 1$ eliminates at least 3 gates from this circuit

## 2 General Setting

By $B_n$ we denote the set of all Boolean functions $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$. A circuit over a basis $\Omega \subseteq B_2$ is a directed acyclic graph with nodes of in-degree 0 or 2. Nodes of in-degree 0 are marked by variables from $\{x_1, \ldots, x_n\}$ and are called inputs. Nodes of in-degree 2 are marked by functions from $\Omega$ and are called gates. There is also a special output gate where the result is computed. The size of a circuit is its number of gates. By $C_\Omega(f)$ we denote the minimum size of a circuit over $\Omega$ computing $f$. The two commonly studied bases are $B_2$ and $U_2 = B_2 \setminus \{\oplus, \equiv\}$.

We call a function $f \in B_2$ degenerate if it does not depend essentially on some of its variables, i.e., there is a variable $x_i$ such that the sub-functions $f|_{x_i=0}$ and $f|_{x_i=1}$ are equal. It is easy to see that a gate computing a degenerate function from $B_2$ can be easily eliminated from a circuit without increasing its size (when eliminating this gate one may need to change the functions computed at its successors). The set $B_2$ contains the following sixteen functions $f(x, y)$:

- six degenerate functions: 0, 1, $x$, $x \oplus 1$, $y$, $y \oplus 1$;
- eight functions of the form $((x \oplus a)(y \oplus b)) \oplus c$, where $a, b, c \in \mathbb{F}_2$ (throughout all the paper we write $xy$ instead of $x \wedge y$); we call them $\wedge$-type functions;
- two functions of the form $x \oplus y \oplus a$, where $a \in \mathbb{F}_2$; these are called $\oplus$-type functions.

An example on simplifying a circuit is given in Fig. 2. We assign $x_2$ the value 1. Then $Q$ computes the constant 1, so $P$ and hence also $R$ compute $x_1 \oplus 1$. These 3 gates can be eliminated from the circuit. After that $S$ computes $(x_1 \oplus 1) \oplus x_4$, i.e., $x_1 \equiv x_4$, while $T$ computes $(x_1 \oplus 1)S$. The negation sign on the wire from $x_1$ to $T$ is intended to reflect the fact that the binary function computed at $T$ is not just $xy$ as in the picture, but $(x \oplus 1)y$.

Below we state several simple but important facts illustrated in this example.

- The substitution $x_2 = 1$ trivializes the gate $Q$ (i.e., makes it constant), so not only $Q$ is eliminated, but also all its successors. At the same time, $P$ is not trivialized, but becomes a degenerate function. This illustrates the difference between $\oplus$- and $\wedge$-type gates and explains why currently best lower bounds for circuits over $U_2$ are stronger than those for circuits over $B_2$.

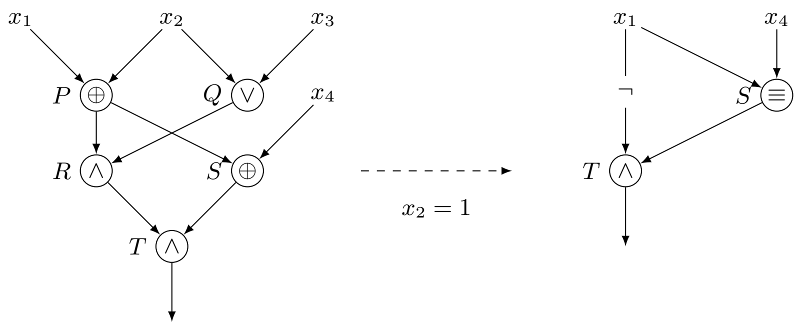


**Fig. 2.** Example of simplifying a circuit under a substitution $x_2 = 1$

- While simplifying a circuit under a substitution one may need to change the functions computed at gates.
- The resulting circuit depends on neither $x_2$ nor $x_3$, though only $x_2$ was substituted.

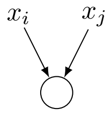
## 3 Known Lower Bounds

Below we review some of the known lower bounds on circuit complexity and in each case indicate a property of a Boolean function that is important for the proof. We concentrate on the circuit size, while there are many other models such as formulas, branching programs, monotone circuits, constant-depth circuits, where functions with other interesting properties are needed. Note that apart from the properties described below, each function for which one would like to prove a lower bound by the gate elimination method must also satisfy the following natural property: it must remain essentially the same after replacing a variable by a constant.

- Bounds on $C_{B_2}$
  - Schnorr [7] proved a $2n - c$ lower bound on $C_{B_2}$ for a function satisfying the following property: for any two different input variables $x_i$ and $x_j$, there are at least three different sub-functions among

$$f|_{x_i=0,x_j=0}, f|_{x_i=1,x_j=0}, f|_{x_i=0,x_j=1}, f|_{x_i=1,x_j=1} .$$

    This property is needed to argue that a top of a circuit cannot look like this:



    That is, at least one of $x_i$ and $x_j$ must feed also some other gate (as otherwise one would get at most two different subfunctions w.r.t. $x_i$ and $x_j$). One then assigns a constant to this variable and kills two gates. A $2n - c$ lower bound follows by induction.

There are many natural functions satisfying this property. E.g., $\mathrm{MOD}^n_{m,r}$, for $m \geq 3$, defined as follows:

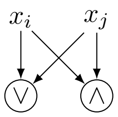
$$\mathrm{MOD}^n_{m,r}(x_1, \dots, x_n) = 1 \text{ iff } \sum_{i=1}^{n} x_i \equiv r \pmod{m}\,.$$

- Paul [8] proves a $2n - o(n)$ lower bound on $C_{B_2}$ for the storage access function: for $a \in \mathbb{F}_2^{\log n}$ and $x \in \mathbb{F}_2^n$, $f(a,x) = x_{\bar{a}}$, where $\bar{a}$ is the number from $\{0, \dots, n-1\}$ whose binary representation is $a$, and $x_{\bar{a}}$ is the corresponding bit of $a$. An important property of this function is that for any input variable $x_i$, when all the bits of $a$ are already assigned, the output of $f$ either equals $x_i$ or does not depend on $x_i$ at all. This allows to substitute $x_i$ not only by a constant, but by an arbitrary function.
- Stockmeyer [9] proved a $2.5n - c$ lower bound on $C_{B_2}$ for many symmetric functions (in particular, for all $\mathrm{MOD}^n_{m,r}$ functions). He essentially uses the fact that for symmetric functions substituting $x_i = h$, $x_j = h \oplus 1$ for a function $h$ is the same as just saying that $x_i \oplus x_j = 1$.
- A function for which Blum [2] proved a $3n - o(n)$ lower bound on $C_{B_2}$ (a similar function was also used by Paul [8]) is defined as follows. Let $a, b, c \in \mathbb{F}_2^{\log n}$, $x \in \mathbb{F}_2^n$, $p, q, r \in \mathbb{F}_2$. Then

$$f(a,b,c,p,q,r,x) = q((x_{\bar{a}} x_{\bar{b}}) \vee (p x_{\bar{b}}(x_{\bar{c}} \oplus r))) \vee (1 \oplus q)(x_{\bar{a}} \oplus x_{\bar{b}})\,.$$

  For any $x_i$ and $x_j$, one can get $x_i \oplus x_j$ as well as $x_i x_j$ from $f$ by assigning some of the remaining variables.
- Kojevnikov and Kulikov [10] proved a $7n/3 - c$ lower bound for functions with high multiplicative complexity. Any circuit computing such a function must have several $\wedge$-type gates. This allows to assign different weights to $\oplus$- and $\wedge$-type gates when counting the number of gates that are eliminated.

– Bounds on $C_{U_2}$
  - Schnorr [7] proved a $3n - c$ lower bound on $C_{U_2}$ for the parity function. A property that helps here is that an optimal circuit cannot contain a variable of out-degree exactly 1. Indeed, if such a variable $x_i$ existed, one could substitute all the other variables to trivialize the unique gate fed by $x_i$. This would make the function independent of $x_i$, a contradiction.
  - Zwick [11] proved a $4n - c$ lower bound for all $\mathrm{MOD}^n_{m,r}$ functions, $m \geq 3$. He noticed that any optimal circuit for such a function can contain only a constant number of out-degree 1 variables. This allows to remove such variables from the consideration by using a circuit complexity measure equal to the number of gates minus the number of out-degree 1 variables.
  - Iwama, Lachish, Morizumi, and Raz [3] used Zwick's circuit complexity measure to prove a lower bound $5n - o(n)$ on $C_{U_2}$ for strongly two-dependent functions, i.e., functions satisfying the following property: for any two variables all the four sub-functions resulting by fixing the values of these variables are different. This property guarantees that a top of a circuit cannot look like this:



This case is the main bottleneck in Zwick's proof. An explicit construction of a strongly two-dependent function was previously given by Savicky and Zak [12]. In fact, this function is even $k$-mixed, for $k = n - o(n)$: for any subset of $k$ variables, all the $2^k$ sub-functions w.r.t. these $k$ variables are different. Recently, Amano and Tarui [13] showed that this property is not enough for proving stronger than $5n$ lower bounds on $C_{U_2}$ by constructing a function of circuit complexity $5n + o(n)$ that is $k$-mixed, for $k = n - o(n)$.

## 4 A $3n - o(n)$ Lower Bound

In this section we consider only circuits over $B_2$. Let $\mu(\mathcal{C}) = s(\mathcal{C}) + N(\mathcal{C})$, where $s(\mathcal{C})$ is the size (number of gates) of $\mathcal{C}$ and $N(\mathcal{C})$ is the number of input variables of $\mathcal{C}$ with out-degree at least 1.

**Lemma 1.** *Let $P$ be a gate of a circuit $\mathcal{C}$ that is a $\oplus$-type gate that depends only on $\oplus$-type gates of out-degree 1 and variables. Then there is a variable $x_j$ and a (possibly empty) subset of variables $I \subseteq \{1, \dots, n\} \setminus \{j\}$ such that for any constant $c \in \mathbb{F}_2$, the substitution $x_j = \bigoplus_{i \in I} x_i \oplus c$ makes the gate $P$ constant and reduces $N(\mathcal{C})$ at least by 1.*

*Proof.* Clearly $P$ computes a function $\bigoplus_{i \in I} x_i \oplus x_j \oplus c_0$ for some $1 \leq j \leq n$, $I \subseteq \{1, \dots n\} \setminus \{j\}$, $c_0 \in \mathbb{F}_2$. We analyse the effect of reducing $\mathcal{C}$ under the substitution $x_j = \bigoplus_{i \in I} x_i \oplus c$, for $c \in \mathbb{F}_2$. Let $S$ be the set of gates that $P$ depends on. Clearly $S$ contains at least $|I| - 1$ gates (as $\bigoplus_{i \in I} x_i \oplus c_0$ cannot be computed by less than $|I| - 1$ gates). To simplify $\mathcal{C}$ under the substitution $x_j = \bigoplus_{i \in I} x_i \oplus c$, we eliminate the gate $P$ (as it now computes the constant $c \oplus c_0$) and all its successors (as they now compute degenerate functions).

To reduce $N(\mathcal{C})$ by 1, we need to replace $x_j$ by $\bigoplus_{i \in I} x_i \oplus c$. For this, we eliminate all the gates from $S$ (they were needed only for computing $P$) and add $|I| - 1$ gates computing $\bigoplus_{i \in I} x_i$. We then use them instead of $x_j$. Clearly, the resulting circuit outputs the same as the initial circuit for all $x \in \mathbb{F}_2^n$ such that $x_j = \bigoplus_{i \in I} x_i \oplus c$. $\square$

An example of such simplification is given in Fig. 3 ($I = \{2, 3, 4, 5\}$, $j = 1$).

**Theorem 1.** *Let $f \colon \mathbb{F}_2^n \to \mathbb{F}_2$ be an affine disperser for dimension $d$, $A$ be an affine subspace of $\mathbb{F}_2^n$ of dimension $D$, and $\mathcal{C}$ be a circuit with $n$ inputs such that $\forall x \in A, \mathcal{C}(x) = f(x)$. Then*

$$\mu(\mathcal{C}) \geq 4(D - d) .$$

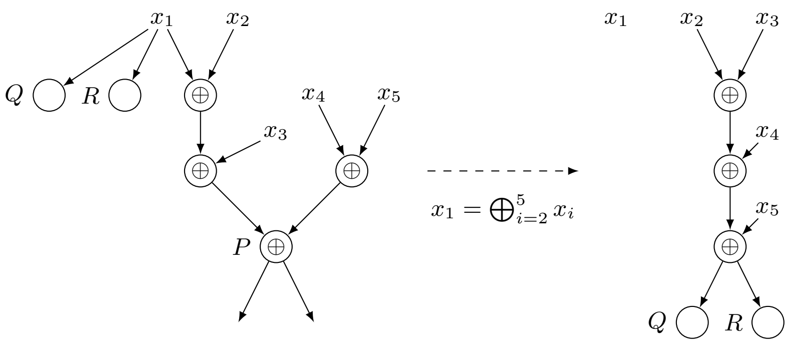


**Fig. 3.** Example of a linear substitution

*Proof.* We prove the inequality by induction on $D$. The base case $D \le d$ is trivial. Consider now the case $D \ge d + 1$. Take a circuit $\mathcal{C}$ computing $f$ on $A$ with the minimal possible $\mu(\mathcal{C})$. Assume wlog that $\mathcal{C}$ does not contain degenerate gates (all such gates can be eliminated without increasing $\mu(\mathcal{C})$). Note also that $\mathcal{C}$ cannot compute a linear function. Indeed, if $\mathcal{C}$ computed a function of the form $\bigoplus_{i\in I} x_i \oplus c$, then $f$ would be constant on an affine space $A' = \{x \in A \colon \bigoplus_{i\in I} x_i = c\}$ of dimension at least $D - 1 \ge d$. Thus, $\mathcal{C}$ contains at least one $\wedge$-type gate.

In the following we find a substitution of the form $\bigoplus_{i\in I} x_i \oplus c$ under which $\mathcal{C}$ is reduced to $\mathcal{C}'$ such that $\mu(\mathcal{C}) \ge \mu(\mathcal{C}') + 4$ and $\mathcal{C}(x) = \mathcal{C}'(x)$ for all $x \in A' = \{x \in A \colon \bigoplus_{i\in I} x_i = c\}$. Since $A'$ has dimension at least $D - 1$, we conclude by induction that $\mu(\mathcal{C}) \ge 4(D - 1 - d) + 4 = 4(D - d)$. Note that any gate that becomes constant under such substitution cannot be an output gate, as otherwise $\mathcal{C}$ would compute a linear function.

Consider a topological order on all the gates of $\mathcal{C}$ and let $P$ be the first gate in this order that is not a $\oplus$-type gate of out-degree 1. Since it depends only on $\oplus$-type gates and input variables, functions computed at both inputs of $P$ are of the form $\bigoplus_{i\in I_1} x_i \oplus c_1$ and $\bigoplus_{i\in I_2} x_i \oplus c_2$. Below we consider five cases, Fig. 4 shows all of them.

- **Case 1.** $P$ is a $\oplus$-type gate of out-degree at least 2. Then it clearly computes a function of the form $\bigoplus_{i\in I} x_i \oplus c$. By the lemma above, by making $P$ constant we reduce $\mu$ at least by 4.
- **Case 2.** $P$ is an $\wedge$-type gate.
  - **Case 2.1.** One of the inputs of $P$ is a gate $Q$. Then $Q$ is a $\oplus$-type gate. By making $Q$ the constant (as in the lemma) which trivializes $P$ we kill $P$, $Q$, and all the successors of $P$. Also, $N(\mathcal{C})$ is reduced at least by 1, hence $\mu$ is reduced at least by 4.
  - **Case 2.2.** Both inputs of $P$ are variables $x_i$ and $x_j$ and at least one of them (say, $x_i$) have out-degree at least 2. By assigning $x_i$ the constant which trivializes $P$ we kill all the successors of $x_i$ and all the successors of $P$. Clearly $N(\mathcal{C})$ is reduced at least by 1. By considering two sub-cases we show that $s(\mathcal{C})$ is reduced at least by 3.

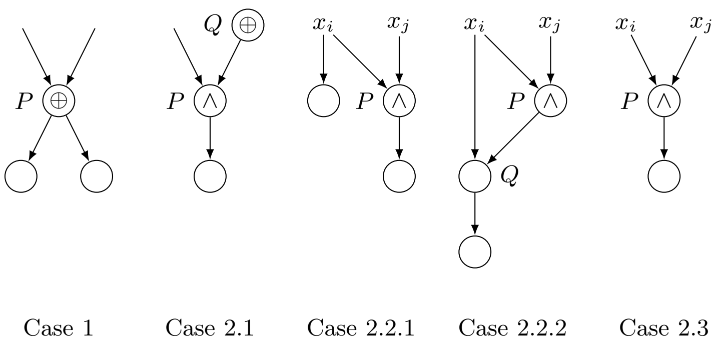


**Fig. 4.** All cases of the proof

* ∗ **Case 2.2.1.** $x_i$ has a successor that is not fed by $P$. Then this successor is eliminated as well as $P$ and all the successors of $P$.
* ∗ **Case 2.2.2.** The only successor of $P$ is the gate $Q$ and it is also fed by $x_i$. Then clearly it becomes constant (as both its inputs are constants) and so all its successors are also eliminated.

- **Case 2.3.** Both inputs of $P$ are out-degree 1 variables $x_i$ and $x_j$. By assigning $x_i$ the constant which trivializes $P$, we eliminate $P$ and all its successors and reduce $N(\mathcal{C})$ at least by 2. Hence $\mu$ is reduced at least by 4. □

**Corollary 1.** *Any circuit over $B_2$ computing an affine disperser for dimension $d$ has at least $3n - 4d$ gates.*

*Proof.* Indeed, by Theorem 1, for any circuit $\mathcal{C}$ computing an affine disperser for dimension $d$,

$$s(\mathcal{C}) = \mu(\mathcal{C}) - N(\mathcal{C}) \geq 4(n-d) - N(\mathcal{C}) \geq 3n - 4d\,. \qquad \square$$

Thus, an affine disperser for sublinear dimension requires circuits of size at least $3n - o(n)$. It is also easy to see that by the same method one can prove a lower bound $n - o(n)$ on the multiplicative complexity of affine dispersers for sublinear dimension. For this, we just make $n - o(n)$ linear substitutions each time killing the first $\wedge$-type gate.

## 5 Further Directions

1. It would be interesting to improve the presented lower bound by a more involved case analysis or to find another property of Boolean functions implying a stronger than $3n$ lower bound.
2. Another interesting direction is to prove a non-trivial upper bound for an affine disperser.

3. An (apparently) easier problem is to close one of the following gaps (see [9], [14], [11]):

$$2.5n - c \leq C_{B_2}(\mathrm{MOD}_3^n) \leq 3n + c,$$
$$4n - c \leq C_{U_2}(\mathrm{MOD}_4^n) \leq 5n + c\,.$$

   Also it is still not known whether $C(\mathrm{MOD}_p^n)$ is strictly greater than $C(\mathrm{MOD}_q^n)$ for primes $p > q$. Note however that any symmetric function can be computed by a circuit (over $B_2$) of size $4.5n + o(n)$ [14].
4. It would also be interesting to find a Boolean function of multiplicative complexity at least $cn$, for a constant $c > 1$.

**Acknowledgements.** We would like to thank Edward A. Hirsch and Arist Kojevnikov as well as the anonymous referees for helpful comments and Arnab Bhattacharyya for pointing us out the paper [4].

## References

1. Shannon, C.E.: The synthesis of two-terminal switching circuits. Bell System Technical Journal 28, 59–98 (1949)
2. Blum, N.: A Boolean function requiring $3n$ network size. Theoretical Computer Science 28, 337–345 (1984)
3. Iwama, K., Lachish, O., Morizumi, H., Raz, R.: An explicit lower bound of $5n-o(n)$ for boolean circuits (2005) (unpublished manuscript), `http://www.wisdom.weizmann.ac.il/~ranraz/publications/Podedl.ps`
4. Ben-Sasson, E., Kopparty, S.: Affine dispersers from subspace polynomials. In: Proceedings of the Annual Symposium on Theory of Computing (STOC), vol. 679, pp. 65–74. ACM Press, New York (2009)
5. Boyar, J., Peralta, R.: Tight bounds for the multiplicative complexity of symmetric functions. Theoretical Computer Science 396, 223–246 (2008)
6. Boyar, J., Peralta, R., Pochuev, D.: On The Multiplicative Complexity of Boolean Functions over the Basis $(\wedge, \oplus, 1)$. Theoretical Computer Science 235(1), 1–16 (2000)
7. Schnorr, C.P.: Zwei lineare untere Schranken für die Komplexität Boolescher Funktionen. Computing 13, 155–171 (1974)
8. Paul, W.J.: A $2.5n$-lower bound on the combinational complexity of Boolean functions. SIAM Journal of Computing 6(3), 427–433 (1977)
9. Stockmeyer, L.J.: On the combinational complexity of certain symmetric Boolean functions. Mathematical Systems Theory 10, 323–336 (1977)
10. Kojevnikov, A., Kulikov, A.S.: Circuit Complexity and Multiplicative Complexity of Boolean Functions. In: Ferreira, F., Löwe, B., Mayordomo, E., Mendes Gomes, L. (eds.) CiE 2010. LNCS, vol. 6158, pp. 239–245. Springer, Heidelberg (2010)
11. Zwick, U.: A $4n$ lower bound on the combinational complexity of certain symmetric boolean functions over the basis of unate dyadic Boolean functions. SIAM Journal on Computing 20, 499–505 (1991)

12. Savicky, P., Zak, S.: A large lower bound for 1-branching programs. Technical Report TR96-036, ECCC (1996)
13. Amano, K., Tarui, J.: A well-mixed function with circuit complexity $5n \pm o(n)$: Tightness of the lachish-raz-type bounds. In: Agrawal, M., Du, D.-Z., Duan, Z., Li, A. (eds.) TAMC 2008. LNCS, vol. 4978, pp. 342–350. Springer, Heidelberg (2008)
14. Demenkov, E., Kojevnikov, A., Kulikov, A.S., Yaroslavtsev, G.: New upper bounds on the Boolean circuit complexity of symmetric functions. Information Processing Letters 110(7), 264–267 (2010)