The Autocorrelation Transform and its Application to the Classification of Boolean Functions

J. Rice
Dept. of Math & Computer Science
University of Lethbridge
4401 University Dr. W. Lethbridge, AB, Canada
j.rice@uleth.ca

Abstract

Classification of Boolean functions is a known problem in digital logic. There are far too many possible Boolean functions to examine each one, and so we attempt to classify similar functions together into groups, or classes. This paper continues work that uses the autocorrelation transform to determine a classification technique. Some new considerations are raised and additional analysis of the autocorrelation classes is presented.

1. Introduction

When considering Boolean functions of n variables, there exists 2^{2^n} different ways of assigning input and output values, and thus there are 2^{2^n} possible Boolean functions. This number quickly becomes far too large to manage in any useful way. One way to handle this problem is to group or classify functions. The goal is to identify similar characteristics within these classes such that analysis or synthesis efforts applied to a single representative of each class can be leveraged over all other members of the class.

Well-known techniques in this area include NPN classification and spectral classification [3]. In this work we address the autocorrelation transform and its use in classifying Boolean functions. This paper continues work previously introduced in [10] and [8].

2. Background

2.1. The Autocorrelation (AC) Transform

In many cases a transform such as the Hadamard, Walsh, or Rademacher-Walsh is used to convert a Boolean function from the functional domain to the spectral domain [3]. In this work we suggest the use of the (AC) transform for this purpose. The AC transform has recently been revisited by a number of authors, including Rice *et al.* [8, 9, 11], Tomczuk [13], and Karpovosky *et al.* [5, 6, 7, 12]. Like the more well-known Walsh transform, this transform can also be used to re-express a function in the spectral domain. The AC transform is based on the general cross-correlation (convolution) function between two given functions f and g at a distance τ. When $f(X) = g(X)$, the resulting equation gives the cross-correlation of a function with itself, translated by τ, the coefficients of which are referred to as the AC coefficients of the function. The AC function is thus defined as [4]

$$B^{ff}(\tau) = \sum_{v=0}^{2^n-1} f(v) \cdot f(v \oplus \tau) \tag{1}$$

where the superscripts ff are generally omitted, except to distinguish between coefficients for different functions.

For multiple output functions a second step must be performed to combine the AC function for each of the individual functions into the total AC function. In this work we are considering only single-output functions, and so do not need to perform this step. We are assuming the use of $\{0, 1\}$ encoding of the function's output for these definitions, however Equation 1 may also be applied if $\{+1, -1\}$ encoding of the function outputs is used. In this case the resulting AC coefficients are referred to as $C^{ff}(\tau)$, or $C(\tau)$.

An example illustrating the use of Equation 1 may be useful. Let us examine the function $f(X) = x_3 + x_2x_1$; then the truth table for f is shown in Table 1. If we wish to compute the AC coefficient for $\tau = 001$ then we expand Equation 1 as follows:

$$\begin{aligned} B(001) &= \sum_{v=0}^{2^n-1} f(v) \cdot f(v \oplus 001) \\ &= [f(000) \cdot f(000 \oplus 001)] + \cdots \\ &\quad + [f(111) \cdot f(111 \oplus 001)] \end{aligned}$$

	Z	Y		$B(\tau)$	$C(\tau)$
$x_3x_2x_1$	$\{0,1\}$	$\{+1,-1\}$	τ	$\{0,1\}$	$\{+1,-1\}$
000	0	1	000	5	8
001	0	1	001	4	4
010	0	1	010	4	4
011	1	-1	011	4	4
100	1	-1	100	2	-4
101	1	-1	101	2	-4
110	1	-1	110	2	-4
111	1	-1	111	2	-4

Table 1. The output vectors and AC coefficient vectors in both $\{0,1\}$ and $\{+1,-1\}$ encoding the function $f(X) = x_3 + x_2x_1$.

$$\begin{aligned} &= [f(000)\cdot f(001)] + \cdots + [f(111)\cdot f(110)] \\ &= [0\cdot 0] + [0\cdot 0] + [0\cdot 1] + [1\cdot 0] \\ &\quad + [1\cdot 1] + [1\cdot 1] + [1\cdot 1] + [1\cdot 1] \\ &= 0+0+0+0+1+1+1+1 \\ &= 4 \end{aligned}$$

We note that in performing this computation we assume the use of the logical exclusive-or operator to combine v and τ and the use of the arithmetic multiplication and summation operators to combine the subsequent values. Table 1 lists the resulting AC coefficients for f for all values of τ.

2.2. Classification

There are many techniques for classification, but in general a classification of a set of functions F into classes $Q_1, Q_2, \ldots, Q_p$ based on transformations $T_1, T_2, \ldots, T_m$ is such that

$$F = Q_1 \cup Q_2 \cup \cdots \cup Q_p$$

and

$$Q_i \cap Q_j = \emptyset.$$

Two functions f_i and f_j, $i \neq j$ are in the same class Q_k if and only if f_i can be obtained from f_j by the application of some appropriate set of transformations from $T_1, \ldots, T_m$. No set of transformations applied to a function in Q_i can lead to a function in Q_j for any $i, j \in \{l, \ldots, p\}$ where $i \neq j$.

3. Meaning

It is useful to briefly consider the meaning of the AC transform and also of the coefficients which result when applying it to a Boolean logic function. Unlike any other type of measure for Boolean logic functions, each AC coefficient provides a measure of similarity between the function unchanged and the same function shifted, or translated by a given amount. This alternative view of the function allows us to identify various properties of the underlying function, such as

- whether or not the function is a trivial function (*i.e.* $f(X) = 1$ or $f(X) = 0$),
- sparse functions,
- independence of one or more variables, and
- exclusive-or decompositions, *e.g.* $f(X) = f^*(X) \oplus x_i$,

as well as testing for symmetries within the function. For instance the function trivial $f(x_3, x_2, x_1) = 1$ has an AC vector of $C = [8,8,8,8,8,8,8,8]$, where every value is 2^n, and the function $f(x_3, x_2, x_1) = x_1x_2x_3$ has an AC vector of $C = [8,4,4,4,4,4,4,4]$, where every value is $2^n - 4$ except for $C(0)$. These and other properties are detailed and proven in [11] and [8].

Many classification techniques discard the signs of the coefficients when classifying Boolean logic functions, but it is through analysis of the meaning of the coefficients that we have decided in this case to retain sign information. For example, $f(x_3, x_2, x_1) = 1$ has all coefficients equal to 8 (2^n), indicating a great deal of similarity at all distances (values of τ). A function $f(x_3, x_2, x_1) = x_1 \oplus x_2 \oplus x_3$ has a coefficient vector of $C = [8, -8, -8, 8, -8, 8, 8, -8]$. This shows that at some distances the function is internally very similar, but at others it is very different. Taken individually this may not be particularly significant, but when comparing one function with another there is a greal deal of difference, and discarding the sign information would result in the loss of this information.

4. Classes

Work in [8] found that the AC classes can be defined by four invariance operations:

1. Permutation of any input variables x_i and x_j, $i, j \in 1..n$, $i \neq j$. This results in the AC coefficients corresponding to these variables also being permuted.
2. Negation of any input variable x_i, $i \in 1..n$. This has no effect on the AC coefficients.
3. Negation of the output of the switching function. Output negation is an invariance operation for the AC classes only when $\{+1,-1\}$ encoding is used. In this case there is no effect on the AC coefficients. If $\{0,1\}$ encoding is used then

$$B^{f^*f^*}(\tau) \quad = \quad B^{ff}(\tau) - 2B^{\,f\,f}(0) + 2^n$$

X	$f(X)$	$g(X)$	τ	$C^{ff}(\tau)$	$C^{gg}(\tau)$
000	0	0	$000 \rightarrow c_0$	8	8
001	0	0	$001 \rightarrow c_1$	-4	-4
010	1	0	$010 \rightarrow c_2$	-4	4
011	0	1	$011 \rightarrow c_{12}$	4	-4
100	0	0	$100 \rightarrow c_3$	-4	-4
101	1	1	$101 \rightarrow c_{13}$	4	4
110	0	1	$110 \rightarrow c_{23}$	4	-4
111	1	0	$111 \rightarrow c_{123}$	-4	4

Table 2. An example showing how replacing x_1 with $x_1 \oplus x_2$ results in the swapping of coefficients c_2 and c_{12} and of coefficients c_{23} and c_{123}.

where $f^*(X) \triangleq \overline{f(X)}$.

4. Replacement of any input variable x_i with $x_i \oplus x_j$, $i, j \in 1..n$, $i \neq j$. This has the result again of swapping certain AC coefficients. Each individual AC coefficient can be referred to as $C(\tau)$, or alternatively as c_α where α consists of the variables corresponding to 1s in the binary expansion of τ. Table 2 illustrates how this notation corresponds to each value of τ.

   Given this notation, if we then label the variable being replaced as x_i and the replacement operation as $x_i \oplus x_j$ then the AC coefficients associated with x_i are not affected, while coefficients associated with x_j are exchanged with those associated with both x_i and x_j, *i.e.* $c_{x_j} \Leftrightarrow c_{x_i x_j}$ and $b_{x_j} \Leftrightarrow b_{x_i x_j}$

   For example, the function $f(X) = \overline{x}_3 x_2 \overline{x}_1 + x_3 x_1$ has the truth table and coefficient vector shown in Table 2. If we construct a function from $f(X)$ by replacing x_1 with $x_1 \oplus x_2$ we get $g(X) = \overline{x}_3 x_2 x_1 + x_3 \overline{x}_2 x_1 + x_3 x_2 \overline{x}_1$, whose output and coefficient vectors are also shown in Table 2. We can see that the coefficients c_2 and c_{12} have been exchanged, as well as coefficients c_{23} and c_{123}.

For the purpose of classification we use only the $\{+1, -1\}$ representation for the AC coefficients.

4.1. Connections between the Spectral and AC Classes

Those familiar with NPN and spectral classification techniques will realize that the first three invariance operations identified in the previous section are the three operations used in NPN (negation of inputs, permutation and negation of output) classification, and that all four invariance operations are also used in spectral classification techniques [3].

The spectral classes, however, are also defined by a fifth invariance operation. This fifth spectral invariance operation consists of replacing $f(X)$ with $f(X) \oplus x_i$. Like output negation, this operation has a very different result on the AC coefficients depending on which encoding has been chosen. This operation involves combining the function's output with one of the inputs using the XOR operator. The resulting effect on the AC coefficients can be stated as follows: if $g^*(x) = g(x) \oplus x_i$, $i \in \{1, ..., n\}$, then

$$\begin{array}{ll} C^{g^*g^*}(\tau_{\,i\alpha}) = -C^{gg}(\tau_{\,i\alpha}) & \text{and} \\ C^{g^*g^*}(\tau_{\,\bar{i}\alpha}) = C^{gg}(\tau_{\,\bar{i}\alpha}) & \forall\, \alpha \text{ such that } i \notin \alpha. \end{array}$$

Note that $\bar{i}\alpha$ refers to a binary expansion of τ such that there is a 0 in the i^{th} bit while the remaining $n-1$ bits have the value α.

There is clearly a tight coupling between the spectral and AC classes, as the invariance operations defining each classification are similar. Given this information, if two functions are both in the same AC class, does this imply that they are both in the same spectral class and vice versa?

Let us define two functions $f_1(X)$ and $f_2(X)$ such that $f_1(X)$ and $f_2(X)$ are in the same AC class. Then by definition we know that either

$$\begin{array}{ll} f_1(X) = f_2(X^*) & \text{or} \\ f_1(X) = \overline{f_2(X^*)} & \end{array}$$

where X^* represents the inputs X modified by one of the three autocorrelation invariance operations that affect the inputs. The last invariance operation affects the output, and is negation, hence the two options given above.

We know that $f_2(X)$ and $\overline{f_2(X)}$ are in the same spectral class, as output negation is one of the spectral invariance operations. We also know that all three of the remaining autocorrelation invariance operations are also spectral invariance operations, so then if $f_1(X) = f_2(X^*)$ or $f_1(X) = \overline{f_2(X^*)}$ then they must be in the same spectral class, by definition.

Now let us redefine our functions such that $f_1(X)$ and $f_2(X)$ are known to be in the same spectral class. Then $f_1(X) = f_2^{**}(X^*)$ where $**$ represents a type (iii) (output negation) or type (v) (replacement of the output with the exclusive-or combination of the output and an input), and $*$ represents a type (i) (permutation), type (ii) (input negation) or type (iv) (replacement of an input with the exclusive-or combination of that input and another input) spectral invariance operation. Then either the two functions are in the same autocorrelation class (if the type (v) invariance operation is not used) or they are in a different class. If they are in a different class, then we can narrow down which classes they may belong to, as certain AC classes are related to each other by the type (v) transformation, as discussed in Section 4.2. This further implies that if the type (v) operation is also applied and one examines only the *magnitude* of the resulting AC coefficients, then the resulting smaller set of classes is identical to the spectral classes. Indeed, this can be seen in Table 4.

4.2. Effect of fifth spectral invariance operation on the AC Classes

Each AC class defined by the four invariance operations may be related to one or more other classes through the application of the fifth spectral operation. Identification of these related classes is relatively straightforward, since the only difference in AC values between the classes is the sign of the values. For example, let us examine the functions $f(X) = x_3 + x_2x_1$ and $g(X) = x_3\overline{x}_1 + \overline{x}_3\overline{x}_2x_1$ where $g(X)$ is generated by replacing the output $f(X)$ with $f(X) \oplus x_1$. The AC vectors for each of these functions are as follows:

$$\begin{array}{llrrrrrrrrl} C^{ff} & = [8 & 4 & 4 & 4 & -4 & -4 & -4 & -4] & \text{and} \\ C^{gg} & = [8 & -4 & 4 & -4 & -4 & 4 & -4 & 4] & \end{array}$$

Of note in this example is the fact that the only difference between the two spectra occurs in the sign, and that these differences occur only where the value of τ contains a 1 in the x_1 position (again, assuming a variable ordering of x_3, x_2, x_1). In addition, Table 4 illustrates how certain classes are related through the application of this operation.

4.3. Selection of Canonical Representatives

The selection of canonical representatives for the AC classes is described in [8] and [10]. In general we select a representative function for each class such that the first order coefficients are the highest valued, followed by second order, and so on. Within each order the values decrease such that c_n has the highest value, followed by c_{n-1}, and continuing down to c_1.

As detailed in [10] this ordering was chosen to coincide with the canonical spectrums for the spectral classes, which were devised such that the canonical representative would have an optimum synthesis in terms of threshold logic [1]. We can draw some similar conclusions for the AC classes. The highest possible AC coefficient value is 2^n, and if that value appears for a particular first order coefficient this indicates that the function is independent of the corresponding variable [8]. One can extend this to say that the higher the value for a particular first order coefficient, the less dependent the function is on that particular variable. This could be useful in the context of reversible logic synthesis, where one technique requires identification of variables that appear more or less frequently in the cubelist describing the function [2].

Table 3 in the Appendix lists the canonical AC class representatives for $n \leq 4$. In this work we chose to retain sign information; however, if we chose to discount the sign information then the resulting reduced set of canonical AC spectra is shown in Table 4. It should be noted that this results in the same number of classes for $n \leq 4$ as does the spectral classification technique.

5. Uses

5.1. Logic Synthesis from Representative Implementations

As suggested for the spectral classes, the intent of the AC classes is that an implementation can be synthesized for a canonical representation of a class, and then additional functions from the same class can be synthesized by adding logic to the design for the canonical function. For example, the function $f(X) = x_1x_2 + x_1x_3 + x_2x_3$ is a very desirable function, as it is totally symmetric. If we define f^* to be some function in the same AC class as f, then the AC coefficients may be used to identify how logic may be added to convert $f(X)$ into $f^*(X)$. We point out that $f(X)$ is very likely to have an efficient implementation due to the symmetry properties it possesses, while $f^*(X)$ may or not may not possess the same properties. Figure 1 illustrates such a situation.

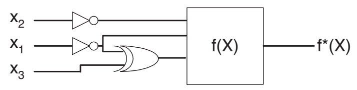

Figure 1. The additional logic required to convert f into f^*.

There is, of course, one problem; the input negation invariance operation has no effect on the AC coefficients, meaning that two functions may have the same AC coefficient but actually be separate functions, one with negated literals and the other with non-negated literals. Work developing an algorithm for this process must take this factor into account, either disregarding whether positive or negative literals are used in the functions, or by performing some sort of verification at each stage of the algorithm to determine if the correct function has been synthesized.

5.2. Identification of Useful Properties

In addition to the above application the AC classes may help identify or rule out a variety of useful properties such as degeneracy, symmetry, self-duality and sparseness.

Degeneracy Degeneracy refers to a function being independing of one or more of the variables listed. This is easy to identify using the AC coefficients, as any first order coefficient (a coefficient whose value of τ contains only a single 1) with a value of 2^n signifies that the function is independent of that variable. For $n \leq 4$ we have, referring to the numeric labeling used in Table 3, classes 1–6 each containing functions possessing some degree of degeneracy. In fact, as mentioned in Section 4.3 the canonical representative for each of these classes is guaranteed to exhibit the highest degree of degeneracy possible for the class.

Symmetry Although in practical terms the number of totally symmetric functions is small, there are a variety of ways in which a totally symmetric function can be defined. For instance, a function whose only true minterm is $x_4x_3x_2x_1$ is totally symmetric, and so is the function $f(X) = x_1+x_2+x_3+x_4$, which has 2^n-1 true minterms. We can enumerate the types of functions which are totally symmetric combinatorically for $n \leq 4$. Initial investigations seem to suggest that most classes can contain totally symmetric functions, with the exception of class 9.

Self-Duality A self-dual function is one for which $f(X) = \overline{f}(\overline{X})$ holds true. It can be shown that such a function must have 2^{n-1} true minterms. This significantly reduces the number of classes that can contain self-dual functions, and in fact in Table 3 only classes 2, 4, 13 and 15 can contain self-dual functions.

Sparseness A function whose output vector contains a relatively small number of 1s can be said to be sparse. If we define sparseness as containing 2^{n-2} or fewer 1s, then this can also be identified through the AC coefficients. [8] states that a function with all coefficients $C(\tau) = 2^n - 4$ except $C(0)$ either has exactly one true minterm or exactly one false minterm. This is the situation for class 8. This has also been extended to functions having exactly two true (false) minterms, and ongoing work is examining generalizations for 2^{n-2} true (false) minterms.

6. Conclusions & Future Work

In this paper we present some additional considerations relating to the use of the AC coefficients in formulating a classification technique for Boolean functions. The proposed classes are closely related to the well-known spectral classes, but we retain sign information so that valuable data on the internal structure of functions is not lost. Work on analysis of these classes is ongoing, and other useful properties such as unateness, monotonicity and decomposability are being considered.

References

[1] C. Edwards. The application of the Rademacher-Walsh transform to Boolean function classification and threshold logic synthesis. *IEEE Transactions on Computers*, C–24(1):48–62, Jan. 1975.

[2] K. Fazel, M. Thornton, and J. E. Rice. ESOP-based Toffoli Gate Cascade Generation. In *Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, pages 206–209, 2007. Aug. 22–24 2007, Victoria, BC, Canada, IEEE Press.

[3] S. L. Hurst, D. M. Miller, and J. C. Muzio. *Spectral Techniques in Digital Logic*. Academic Press, Inc., Orlando, Florida, 1985.

[4] M. Karpovsky. *Finite Orthogonal Series in the Design of Digital Devices*. John Wiley & Sons, 1976.

[5] M. G. Karpovsky and E. S. Moskalev. Utilization of autocorrelation functions for realization of systems of logical functions. *Automata. and Remote Control*, 31(2):243–250, Feb. 1970. Downloaded from http://mark.bu.edu/resume.htm.

[6] M. G. Karpovsky and P. Nagvajara. Functions with flat autocorrelation and their generalizations. In *Proceedings of the 3rd International Workshop on Spectral Techniques*, pages 56–66, 1988. Downloaded from http://mark.bu.edu/resume.htm.

[7] M. G. Karpovsky, R. Stancovic, and J. Aastola. Reduction of sizes of decision diagrams by autocorrelation functions. *IEEE Transactions on Computers*, pages 592–607, May 2003.

[8] J. E. Rice. *Autocorrelation Coefficients in the Representation and Classification of Switching Functions*. PhD thesis, University of Victoria, 2003.

[9] J. E. Rice. On the use of autocorrelation coefficients in the identification of three-level decompositions. In *Proceedings of the International Workshop on Logic Synthesis (IWLS)*, pages 187–191, 2003.

[10] J. E. Rice and J. C. Muzio. Use of the autocorrelation function in the classification of switching functions. In *Proceedings of the Euromicro Symposium on Digital System Design: Architectures, Methods and Tools (DSD)*, pages 244–251, 2002.

[11] J. E. Rice, J. C. Muzio, N. Anderson, and R. Jansen. Properties of autocorrelation coefficients for single-output switching functions. *in preparation*, 2009.

[12] R. S. Stankovic, M. G. Karpovsky, and J. T. Aastola. Reduction of the number of coefficients in arithmetic expressions by autocorrelation functions. In *Proceedings of the 2004 International Workshop on Spectral Methods and Multirate Signal Processing (SMMSP)*, 2004. Downloaded from http://mark.bu.edu/resume.htm.

[13] R. Tomczuk. *Autocorrelation and Decomposition Methods in Combinational Logic Design*. PhD thesis, University of Victoria, 1996.

A Canonical AC Class Representatives, $n \leq 4$

Table 3 lists the 18 canonical class representatives for the AC classes for $n \leq 4$. It should be noted that there are 8 spectral classes for $n \leq 4$.

class no.																	
1	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	
2	16	16	16	16	-16	16	16	-16	16	-16	-16	-16	-16	-16	16	-16	
3	16	16	16	0	0	16	0	0	0	0	0	0	0	0	0	0	
4	16	16	0	0	0	0	0	0	0	0	0	-16	0	0	0	-16	
5	16	16	8	8	8	8	8	8	8	8	8	8	8	8	8	8	
6	16	16	8	8	-8	8	8	-8	8	-8	-8	-8	-8	-8	8	-8	
7	16	12	12	12	-12	12	12	-12	12	-12	-12	-12	-12	-12	12	-12	
8	16	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	
9	16	12	12	4	4	12	4	4	4	4	4	4	4	4	4	4	
10	16	12	12	4	-4	12	4	-4	4	-4	-4	-4	-4	-4	4	-4	
11	16	12	4	4	-4	4	4	-4	4	-4	-4	-12	-4	-4	4	-12	
12	16	8	8	8	0	0	8	0	8	0	0	0	0	0	8	0	
13	16	8	8	8	-8	8	8	-8	8	-8	-8	-8	-8	-8	8	-16	
14	16	8	8	0	0	8	0	0	0	0	0	-8	-8	0	0	-8	
15	16	8	8	0	0	0	0	0	0	0	-8	-8	-8	0	0	-8	
16	16	4	4	4	4	4	4	-4	-4	4	-4	-4	-4	4	4	4	
17	16	4	4	4	4	4	-4	-4	-4	-4	4	-4	-4	-4	-4	-4	
18	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	0000	1000	0100	0010	0001	1100	1010	1001	0110	0101	0011	0111	1011	1101	1110	1111	τ

Table 3. The canonical representatives for the $n \leq 4$ AC classes in $\{+1, -1\}$ notation.

class no.																	
1,2	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	
3,4	16	16	16	0	0	16	0	0	0	0	0	0	0	0	0	0	
5,6,13	16	16	8	8	8	8	8	8	8	8	8	8	8	8	8	8	
7,8	16	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	
9,10,11	16	12	12	4	4	12	4	4	4	4	4	4	4	4	4	4	
12,14,15	16	8	8	8	0	0	8	0	8	0	0	0	0	0	8	0	
16,17	16	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	
18	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	0000	1000	0100	0010	0001	1100	1010	1001	0110	0101	0011	0111	1011	1101	1110	1111	τ

Table 4. The canonical representatives for the $n \leq 4$ AC classes in $\{+1, -1\}$ notation, but with all sign information removed.