

# Walsh Analysis

Darrell Whitley  
Colorado State AI Lab  
Colorado State University

## A Simple 3-Bit Function Decomposition

$$\begin{aligned} f(b_2b_1b_0) = & w_0 + \\ & w_1b_0 + w_2b_1 + w_4b_2 + \\ & w_3b_0b_1 + w_5b_0b_2 + w_6b_1b_2 + \\ & w_7b_0b_1b_2 \end{aligned}$$

## A Linear Example: Binary

000	001	010	011	100	101	110	111
0	1	2	3	4	5	6	7

$$f(000) = w_0 = 0$$

$$f(001) = w_0 + w_1 = 1$$

$$f(010) = w_0 + w_2 = 2$$

$$f(011) = w_0 + w_1 + w_2 + w_3 = 3$$

$$f(100) = w_0 + w_4 = 4$$

$$f(101) = w_0 + w_1 + w_4 + w_5 = 5$$

$$f(110) = w_0 + w_2 + w_4 + w_6 = 6$$

$$f(111) = w_0 + w_1 + w_2 + w_4 + w_3 + w_5 + w_6 + w_7 = 7$$

$w_0$	$w_1$	$w_2$	$w_3$	$w_4$	$w_5$	$w_6$	$w_7$
0	1	2	0	4	0	0	0

## A Nonlinear Example: Gray

000	001	010	011	100	101	110	111
0	1	3	2	7	6	4	5

$$f(000) = w_0 = 0$$

$$f(001) = w_0 + w_1 = 1$$

$$f(010) = w_0 + w_2 = 3$$

$$f(011) = w_0 + w_1 + w_2 + w_3 = 2$$

$$f(100) = w_0 + w_4 = 7$$

$$f(101) = w_0 + w_1 + w_4 + w_5 = 6$$

$$f(110) = w_0 + w_2 + w_4 + w_6 = 4$$

$$f(111) = w_0 + w_1 + w_2 + w_4 + w_3 + w_5 + w_6 + w_7 = 5$$

$w_0$	$w_1$	$w_2$	$w_3$	$w_4$	$w_5$	$w_6$	$w_7$
0	1	3	-2	7	-2	-6	4

# Walsh Analysis

## The Walsh Decomposition

$$f(x) = \sum_{j=0}^{2^L-1} w_j \psi_j(x)$$

Let  $bc(j)$  count the number of 1 bits in string  $j$ :

$$\psi_j(x) = (-1)^{bc(j \wedge x)}$$

If  $bc(j \wedge x)$  is odd, then  $\psi_j(x) = -1$

If  $bc(j \wedge x)$  is even, then  $\psi_j(x) = 1$ .

$$w_j = \frac{1}{2^L} \sum_{i=0}^{2^L-1} f(i) \psi_j(i)$$

## A Simple 3-Bit Function Decomposition

$$\begin{aligned} f(b_2b_1b_0) &= w_0 + \\ &w_1b_0 + w_2b_1 + w_4b_2 + \\ &w_3b_0b_1 + w_5b_0b_2 + w_6b_1b_2 + \\ &w_7b_0b_1b_2 \end{aligned}$$

## A Walsh Decomposition

$$\text{Let } x = b_2b_1b_0$$

$$\begin{aligned} f(x) &= w_0 + \\ &w_1\psi_1(x) + w_2\psi_2(x) + w_4\psi_4(x) + \\ &w_3\psi_3(x) + w_5\psi_5(x) + w_6\psi_6(x) + \\ &w_7\psi_7(x) \end{aligned}$$

The Walsh transform, denoted  $W$ , acts as an invertible function analogous to a Fourier transform.

$$w_j = W(f(x)) \quad f(x) = W^{-1}(w_j)$$

A Fast Walsh Transform exists—analogue to an FFT.

**The Walsh Transform in matrix form:**

$$\vec{w} = \frac{1}{2^L} \vec{f}^T M$$

where  $M_{i,j} = \psi_j(i)$

$$f(000) = 5$$

$$f(001) = 7$$

$$f(010) = 2$$

$$f(011) = 4$$

$$f(100) = 6$$

$$f(101) = 1$$

$$f(110) = 3$$

$$f(111) = 8$$

$$w_{000} = 4.5$$

$$w_{001} = -0.5$$

$$w_{010} = 0.25$$

$$w_{011} = 1.25$$

$$w_{100} = 0.00$$

$$w_{101} = -0.5$$

$$w_{110} = 1.25$$

$$w_{111} = -1.25$$

$$\vec{w} = \frac{1}{2L} \vec{f}^T M$$

$$\begin{bmatrix} 4.5 \\ -0.50 \\ 0.25 \\ 1.25 \\ 0.00 \\ -0.50 \\ 1.25 \\ -1.25 \end{bmatrix} = \frac{1}{8} \begin{bmatrix} 5 \\ 7 \\ 2 \\ 4 \\ 6 \\ 1 \\ 3 \\ 8 \end{bmatrix}^T \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

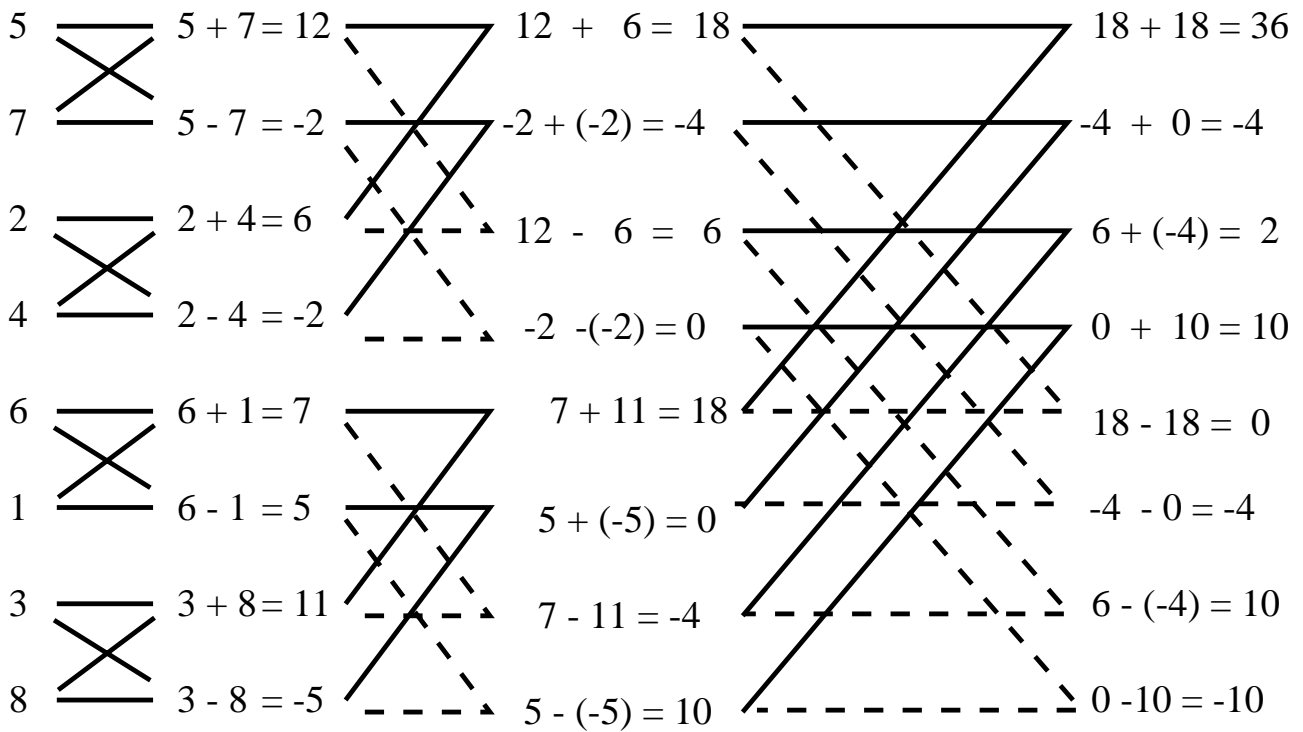
The Walsh Matrix can also be expressed as a recursive decomposition:

$$W(0) = 1$$

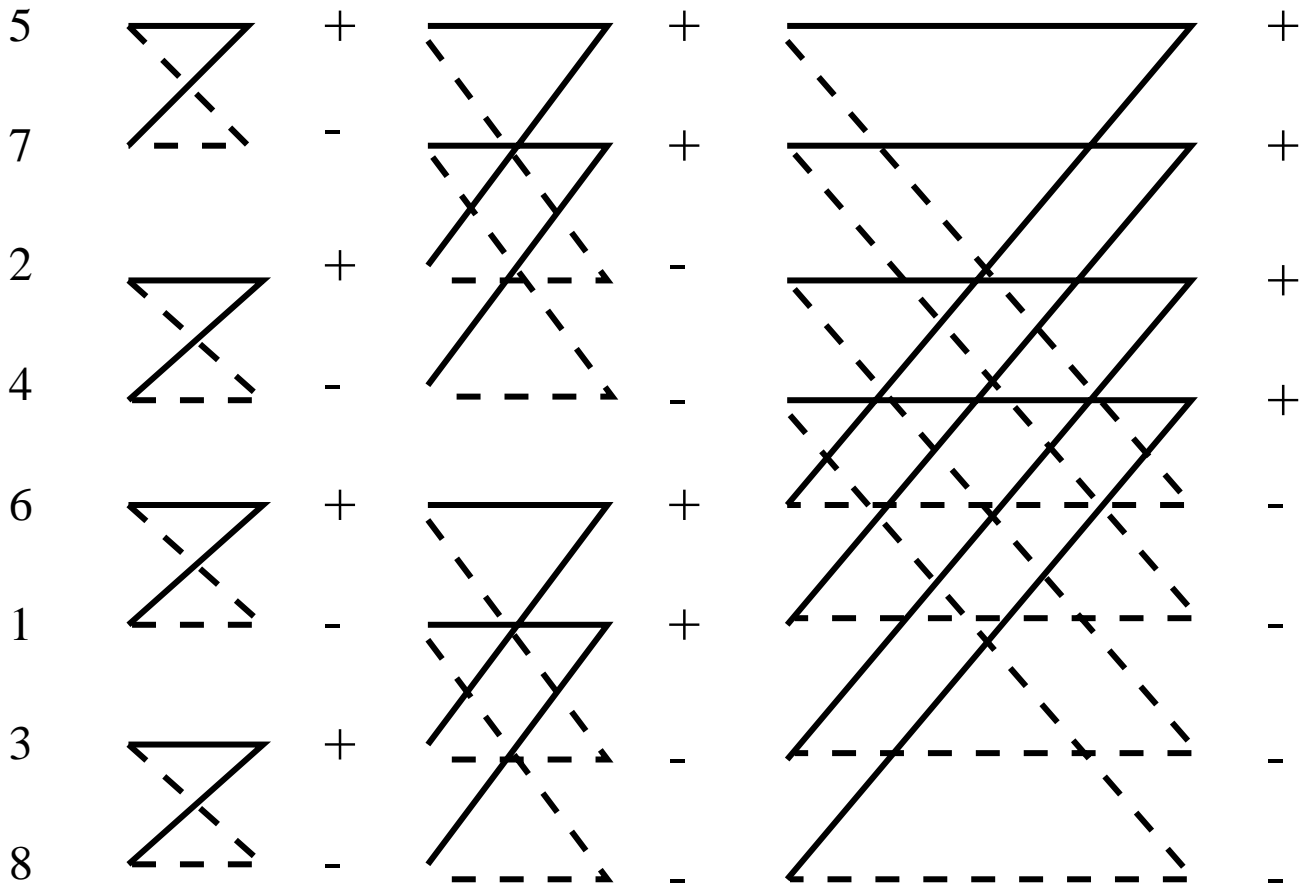
$$W(1) = \begin{matrix} 1 & 1 \\ 1 & -1 \end{matrix}$$

$$W(2) = \begin{matrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{matrix}$$

$$W(N) = \begin{matrix} W(N-1) & W(N-1) \\ W(N-1) & -W(N-1) \end{matrix}$$



The Fast Walsh Transform.



The Fast Walsh Transform.

## Walsh Analysis and Schema Fitness

$$f(* * 0) = (5 + 2 + 6 + 3)/4 = 4$$

$$f(* * 0) = w_{000} + w_{001} = 4.5 - 0.5 = 4$$

$$f(* * 1) = (7 + 4 + 1 + 8)/4 = 5$$

$$f(* * 1) = w_{000} - w_{001} = 4.5 + 0.5 = 5$$

$$f(0 * 1) = (7 + 4)/2 = 5.5$$

$$f(0 * 1) = w_{000} + w_{100} - w_{001} + w_{101} = 5.5$$

Functions  $\alpha$  and  $\beta$  are defined on a schema,  $h$ :

$$\alpha(h)[i] = \begin{cases} 0 & \text{if } h[i] = * \\ 1 & \text{if } h[i] = 0 \text{ or } 1 \end{cases}$$

$$\beta(h)[i] = \begin{cases} 0 & \text{if } h[i] = * \text{ or } 0 \\ 1 & \text{if } h[i] = 1 \end{cases}$$

$$f(h) = \frac{1}{|h|} \sum_{x \in h} f(x) = \sum_{j \subseteq \alpha(h)} w_j \psi_j(\beta(h))$$

$\alpha(h)$  is a mask used to select  $2^{o(h)}$  relevant coefficients.  
 $\beta(h)$  extracts the 1 bits from the respective coefficients.  
 An odd number of 1 bits yields a negative sign.

Example: Let  $h = **01**$  and compute  $f(h)$

$$\alpha(**01**) = 001100 \quad \text{and} \quad \beta(**01**) = 000100$$

$$j \in \{000000, 000100, 001000, 001100\}$$

$$f(**01**) = w_0 - w_4 + w_8 - w_{12}.$$

## Boolean Satisfiability

Given a logical expression consisting of Boolean variables, determine whether or not there is a setting for the variables that makes the expression TRUE.

**Literal:** a variable or the negation of a variable

**Clause:** a disjunct of literals

### A 3SAT Example

$$(\neg x_2 \vee x_1 \vee x_0) \wedge (x_3 \vee \neg x_2 \vee x_1) \wedge (x_3 \vee \neg x_1 \vee \neg x_0)$$

recast as a MAX3SAT Example

$$(\neg x_2 \vee x_1 \vee x_0) + (x_3 \vee \neg x_2 \vee x_1) + (x_3 \vee \neg x_1 \vee \neg x_0)$$

## Walsh Analysis of a Single Clause

Consider the example function consisting of a single clause

$$f(x) = \neg x_2 \vee x_1 \vee x_0$$

$$f(000) = 1 \quad (\neg x_2 T)$$

$$f(001) = 1 \quad (\neg x_2 T)$$

$$f(010) = 1 \quad (\neg x_2 T)$$

$$f(011) = 1 \quad (\neg x_2 T)$$

$$f(100) = 0 \quad (\neg x_2 F \wedge x_1 F \wedge x_0 F)$$

$$f(101) = 1 \quad (x_0 T)$$

$$f(110) = 1 \quad (x_1 T)$$

$$f(111) = 1 \quad (x_1 T)$$

$$\frac{1}{8} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}^T \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} = \begin{bmatrix} 0.875 \\ -0.125 \\ -0.125 \\ -0.125 \\ 0.125 \\ 0.125 \\ 0.125 \\ 0.125 \end{bmatrix}$$

- All  $\psi_j$ 's except  $\psi_0$  have 4 1's and 4  $-1$ 's.
- $\psi_0$  has all 1's.
- $\vec{f}$  for clauses of length 3 will contain 7 1's and 1 0

Let  $neg(f)$  return a  $K$ -bit string with 1 bits indicating which variables in the clause are negated.

$$f(100) = 0 \quad (\neg x_2 F \wedge x_1 F \wedge x_0 F)$$

$$neg(f) = 100$$

Then the Walsh coefficients for  $f$  are:

$$w_j = \begin{cases} \frac{2^K - 1}{2^K} & \text{if } j = 0 \\ -\frac{1}{2^K} \psi_j(neg(f)) & \text{if } j \neq 0 \end{cases}$$

## Walsh Analysis for MAXSAT Problems

An  $L$ -bit MAXSAT problem is a sum of  $C$  disjunctive clauses,  $f_i$ :

$$f(x) = \sum_{i=1}^C f_i(x)$$

where  $f, f_1, f_2, \dots, f_C : \mathcal{B}^L \rightarrow \mathcal{R}$ .

The Walsh transform is performed using a linear transformation so the Walsh transform of a MAXSAT problem is a sum of the Walsh transforms of the individual clauses.

$$W(f(x)) = \sum_{i=1}^C W(f_i(x))$$

The functions  $f_i$  contribute to only  $2^{K_i}$  Walsh coefficients.

## An Example Computation

$$f_1 = (\neg x_2 \vee x_1 \vee x_0)$$

$$f_2 = (x_3 \vee \neg x_2 \vee x_1)$$

$$f_3 = (x_3 \vee \neg x_1 \vee \neg x_0)$$

$x$	$w_i$	$W(f_1)$	$W(f_2)$	$W(f_3)$	$W(f(x))$
0000	$w_0$	0.875	0.875	0.875	2.625
0001	$w_1$	-0.125	0	0.125	0
0010	$w_2$	-0.125	-0.125	0.125	-0.125
0011	$w_3$	-0.125	0	-0.125	-0.250
0100	$w_4$	0.125	0.125	0	0.250
0101	$w_5$	0.125	0	0	0.125
0110	$w_6$	0.125	0.125	0	0.250
0111	$w_7$	0.125	0	0	0.125
1000	$w_8$	0	-0.125	-0.125	-0.250
1001	$w_9$	0	0	0.125	0.125
1010	$w_{10}$	0	-0.125	0.125	0
1011	$w_{11}$	0	0	-0.125	-0.125
1100	$w_{12}$	0	0.125	0	0.125
1101	$w_{13}$	0	0	0	0
1110	$w_{14}$	0	0.125	0	0.125
1111	$w_{15}$	0	0	0	0

## NK-Landspaces

A popular experimental model for correlated landscapes.

An NK-landscape can be expressed as an average of  $N$  functions as follows:

$$f(X) = \frac{1}{N} \sum_{j=0}^{N-1} g_j(j, b_{(1,j)}, b_{(2,j)}, \dots, b_{(k,j)})$$

There are  $N$  subfunctions.

And for fixed  $k$ ,  $g_j$  takes on  $2^k$  possible values.

In NK-Landspaces, the function  $g_j$  is random.

We can compute the Walsh coefficients for each subfunction, and sum across the subfunctions.

We can do Walsh Analysis and compute Schema Averages in Linear time when  $K$  (of the NK) is bounded, with constant of size  $2^K$ .

## Embedded-Landspaces

**Embedded landscapes** are a useful generalization of MAXSAT and NK-Landscapes. An embedded landscape,  $f : \mathcal{B}^N \rightarrow \mathcal{R}$ , can be expressed as the sum of  $Q$  embedded functions:

$$f(x) = \sum_{j=1}^Q g_j(\text{SELECT}(x, M_j))$$

where the function **SELECT** extracts the bits selected by the mask,  $M_j$ .

Each subfunction,  $g_j$ , accepts at most  $K$  bits of input.

$Q \leq L$  choose  $K$  and must be Polynomial.

Again, we can compute the Walsh coefficients for each subfunction, and sum across the subfunctions.

We can do Walsh Analysis and compute Schema Averages in PTIME (bounded by  $Q$ ) where the size of the each subfunction is bounded by a constant  $2^K$ .

Since  $\mu = w_0$ , and  $\psi_0(x) = 1 \forall x$ :

$$\mu_r = \frac{1}{2^L} \sum_{x=0}^{2^L-1} \left( \sum_{i=1}^{2^L-1} w_i \psi_i(x) \right)^r$$

Here,  $p(x) = \frac{1}{2^L}$  since we enumerate over all binary strings.

$$\begin{aligned} \mu_r &= \frac{1}{2^L} \sum_{a_1 \oplus a_2 \oplus \dots \oplus a_r = 0} w_{a_1} w_{a_2} \dots w_{a_r} 2^L, \quad a_i \neq 0 \forall i \\ &= \sum_{a_1 \oplus a_2 \oplus \dots \oplus a_r = 0} w_{a_1} w_{a_2} \dots w_{a_r}, \quad a_i \neq 0 \forall i \end{aligned} \quad (1)$$

This formula allows us to compute the variance, skew and kurtosis for any fitness distribution provided we are given the Walsh coefficients.

Given the set of nonzero Walsh coefficients, we can compute the variance, skew and kurtosis for any fitness distribution provided we are given the Walsh coefficients.

The statistics are computed in PTIME for EMBEDDED LANDSCAPES.

$$\textit{variance} = \mu_2 = \sigma^2 \quad \textit{skew} = \frac{\mu_3}{\sigma^3} \quad \textit{kurtosis} = \frac{\mu_4}{\sigma^4}$$

For example, since  $a_1 \oplus a_2 = 0$  if and only if  $a_1 = a_2$  then the variance for any function can be computed

$$\sum_{i=1}^{2^L - 1} w_i w_i$$

But the summary statistics tell us nothing we can use.

Consider the permutation set of  $N!$  functions constructed from the same  $N$  unique values.

The summary statistics are identical for every member of the permutation set.

It is obvious that the summary statistics tells us nothing about the structure of the search space.

And the summary statistics are identical for a permutation set over which No Free Lunch holds.