

# A TRANSFORMATION OF A STANDARD RANDOM WALK

Jason Goldstick  
University of Michigan  
July 20, 2005

## I. INTRODUCTION

This paper is a synopsis of the REU research conducted by myself during Summer 2005 at the University of Michigan, under the supervision of Professor Joe Conlon. The topic is a fusion of discrete state stochastic processes and numerical methods in computing. We begin by introducing the concept of a *standard random walk* and a few other details which leads to the definition of a particular function  $h(x^*, \theta)$  of some real number,  $\theta$ , and all  $m$ -step,  $n$ -dimensional standard random walks that begin at the origin and end on the line  $x = x^*$ ; a function that turns out to form a polynomial in  $\sinh^2 \theta$ . The conjecture is simple but non-trivial: although some particular walks may give a negative contribution to the one or more of the coefficients, when you sum over all of the walks, all of the coefficients turn out to be nonnegative. Herein we are not able to offer a rigorous proof of this conjecture but can offer rather strong evidence. We reason intuitively about why it is true, but rely primarily on numerical evidence using algorithms in Matlab. Such computations in more than 2 dimensions are pushing the capacity of Matlab, so this paper is narrowed to deal with the case where  $n = 1, 2$ . What follows is an in-depth description of the problem in 1 and 2 dimensions, followed by a section summarizing the conclusions. Copies, with comments, of the algorithms used in 1 and 2 dimensions are included in Section V. An appendix is also included and contains several checking programs, a complete enumeration of the coefficients for every computable case in 1 and 2 dimensions, and an alternate program for the 2-D case.

## II. BACKGROUND INFORMATION

### 2.1. Notation

Throughout we will use the following notation:

- $S$ : The state space of a discrete state stochastic process
- $X_n$ : of discrete time stochastic process after  $n$  steps,  $n \in \mathbb{N}$
- $\text{floor}(x)$ : The integer part of  $x$

### 2.2. Definitions

**Definition 2.2.1.** A discrete time stochastic process is a *discrete time Markov Chain* if it satisfies the following property:

$$P(X_{n+1} = x \mid X_0, X_1, X_2, \dots, X_n) = P(X_{n+1} = x \mid X_n) \quad x \in S$$

**Definition 2.2.2.** A standard random walk is a discrete time Markov Chain where the initial position is  $\mathbf{0}$  and the process can move to any adjacent location each with equal probability

### 2.3. Hyperbolic Trig Identities

Familiarity with a few hyperbolic trig identities is essential to the topics that follow. In particular, we are interested in the form of  $\sinh(k\theta)$  and  $\cosh(k\theta)$  for  $k = 2,3,4,\dots$

$$\cosh(x + y) = \cosh(x) \cosh(y) + \sinh(x) \sinh(y) \quad (2.1)$$

$$\sinh(x + y) = \cosh(x) \sinh(y) + \sinh(x) \cosh(y) \quad (2.2)$$

By using the two familiar identities above and making the proper simplifications we can write  $\sinh(k\theta)$  and  $\cosh(k\theta)$  either as polynomial in  $\sinh(\theta)$ , or as  $\cosh(\theta)$  times a polynomial in  $\sinh(\theta)$ . In all practically computable cases  $k$  does not exceed 6, and the following formulas are useful:

$$\sinh(2\theta) = 2 \cosh(\theta) \sinh(\theta) \quad (2.3)$$

$$\sinh(3\theta) = 3 \sinh(\theta) + 4 \sinh^3(\theta) \quad (2.4)$$

$$\sinh(4\theta) = \cosh(\theta)(4 \sinh(\theta) + 8 \sinh^3(\theta)) \quad (2.5)$$

$$\sinh(5\theta) = 5 \sinh(\theta) + 20 \sinh^3(\theta) + 16 \sinh^5(\theta) \quad (2.6)$$

$$\sinh(6\theta) = \cosh(\theta)(6 \sinh(\theta) + 32 \sinh^3(\theta) + 32 \sinh^5(\theta)) \quad (2.7)$$

$$\cosh(2\theta) = 1 + 2 \sinh^2(\theta) \quad (2.8)$$

$$\cosh(3\theta) = \cosh(\theta)(1 + 4 \sinh^2(\theta)) \quad (2.9)$$

$$\cosh(4\theta) = 1 + 8 \sinh^2(\theta) + 8 \sinh^4(\theta) \quad (2.10)$$

$$\cosh(5\theta) = \cosh(\theta)(1 + 12 \sinh^2(\theta) + 16 \sinh^4(\theta)) \quad (2.11)$$

$$\cosh(6\theta) = 1 + 18 \sinh^2(\theta) + 48 \sinh^4(\theta) + 32 \sinh^6(\theta) \quad (2.12)$$

Also note that by using  $\cosh^2(\theta) = 1 + \sinh^2(\theta)$  we can write any term of the form  $\cosh^k(\theta)$  in terms of a polynomial in  $\sinh^2(\theta)$  or as  $\cosh(\theta)$  times a polynomial in  $\sinh^2(\theta)$ . There are a few properties of the above identities that we will tacitly use later:

(1)  $\cosh(k\theta)$  yields a polynomial in  $\sinh^2(\theta)$  with degree  $\text{floor}(\frac{k}{2})$

\*For odd  $k$ , there is a leftover  $\cosh(\theta)$  term

(2)  $\sinh(k\theta)$  yields  $\sinh(\theta)$  times a polynomial in  $\sinh^2(\theta)$  with degree is  $\text{floor}(\frac{k+1}{2})$

\*For even  $k$ , there is a leftover  $\cosh(\theta)$  term

(3)  $\cosh^k(\theta)$  yields a polynomial in  $\sinh^2(\theta)$  with degree  $\text{floor}(\frac{k}{2})$

\*For odd  $k$ , there is a leftover  $\cosh(\theta)$  term

(4) For  $x, y, z \in \mathbb{Z}$ , The expression  $\cosh^x(\theta) \cosh(y\theta) \sinh(z\theta)$  yields a polynomial in  $\sinh^2(\theta)$ . When  $x + y + z$  is odd, the degree is  $\frac{x + y + z - 1}{2}$ ; when  $x + y + z$  is even, the degree is  $\frac{x + y + z - 2}{2}$ . Put simply, the degree is  $\text{floor}(\frac{x + y + z - 1}{2})$ .  
*Note:* for even  $x + y + z$ , there is a leftover  $\cosh(\theta)$  term.

### III. THE ONE DIMENSIONAL CASE

#### 3.1. Description of the problem

The one dimensional case was researched in detail by Professor Joe Conlon before I began my work and much of the material in this section was done previously by him.

**Definition 3.1.1.** In one dimension our definition of a standard random walk amounts to the following:

$$X_0 = 0 \quad t = 0, 1, 2, \dots$$

$$P(X_{t+1} = x + 1 \mid X_t = x) = P(X_{t+1} = x - 1 \mid X_t = x) = \frac{1}{2} \quad x \in S$$

**Definition 3.1.2.** Let  $v \in S$ , we define the quantities  $N^+(v)$  and  $N^-(v)$  as follows:

$N^+(v)$ : The number of times during the walk there was a move from  $v$  to  $v + 1$

$N^-(v)$ : The number of times during the walk there was a move from  $v$  to  $v - 1$

We now have all of the information we need to proceed to the definition of the function of interest. The interpretation of the function is beyond the scope of this paper, but it has to do with analysis of random walks where the parameter  $p$  is itself a random variable (in the case of standard random walks, the parameter  $p = 1/2$ ).

**Definition 3.1.3.** For  $\theta \in \mathbb{R}$ ,  $v \in S$ , our function over all  $m$ -step walks with endpoint  $x^*$  is defined as follows:

$$h(x^*, \theta) = \sum_{\substack{\text{WALKS} \\ X(0)=0 \\ X(m)=x^*}} \prod_{v \neq x^*} \cosh(\{N^+(v) - N^-(v)\}\theta) \sinh(\{N^+(x^*) - N^-(x^*)\}\theta)$$

The basic idea is to pick out all  $m$ -step standard random walks that begin at 0 and end at  $x^*$ . Compute the quantity  $\cosh(\{N^+(v) - N^-(v)\}\theta)$  for all locations in the state space that are not the endpoint, and multiply them all together. You then multiply this product with  $\sinh(\{N^+(x^*) - N^-(x^*)\}\theta)$ . Repeat this process for each walk and then sum the products. So the result will be a sum over cosh terms multiplied by sinh terms which, by the reasoning and formulas in Section 2.3., forms a polynomial of the following form:

$$-\sqrt{z}(a_0 + a_1z + a_2z^2 + a_3z^3 + \dots + a_rz^r) \quad \text{where } z = \sinh^2(\theta), \theta \in \mathbb{R} \quad (3.1.4)$$

The result is a polynomial in the odd powers of  $\sinh(\theta)$  because the  $\sinh(\theta)$  term from the above formula will be a polynomial in the odd powers, and the  $\cosh$  terms will be a polynomial in the even powers, so the resulting polynomial will be in the odd powers of  $\sinh(\theta)$ . In some cases there is a leftover  $\cosh$  term multiplied by the above polynomial. The quantity of interest here are the coefficients,  $a_i$ . Recall from the introduction, the conjecture is that although some walks give a negative contribution to some of the coefficients, the sum over all walks will give all nonnegative coefficients.

That is,

$$a_i \geq 0 \quad i = 0, 1, \dots, r \quad (3.1.5)$$

This can happen because terms from the walks which contribute negatively are cancelled out by the terms from the walks that contribute positively. We will see some examples later to make this clearer.

## 3.2. Properties and Miscellaneous

### 3.2.1. Accessibility of $x$ on an $m$ -step Walk

If an  $m$ -step standard random walk can end at the location  $x$ , then  $m - x \equiv 0 \pmod{2}$

*Proof.* Let  $r, l$  be the number of moves to the right, and left, respectively.

*Assume:* An  $m$ -step standard random walk can end at the location  $x$

$$\Rightarrow m = r + l, \quad x = r - l \quad \Rightarrow (r + l) - (r - l) = m - x$$

$$\Rightarrow 2l = m - x \quad \Rightarrow m - x \equiv 0 \pmod{2} \quad \square$$

### 3.2.2. Counting the Number of $m$ -step Walks That End at $x$

The number of  $m$ -step random walks that end at the location  $x$  can be counted with the following formula:

$$N(m, x) = \binom{m}{\frac{m-x}{2}} I\left\{\frac{m-x}{2} \in \mathbf{Z}\right\} \quad (\text{Eq. 1})$$

*Proof.*

$$m = r + l, \quad x = r - l$$

$$\Rightarrow 2l = m - x \quad \Rightarrow l = \frac{m - x}{2}$$

$\Rightarrow$  Of the  $m$  total steps, there are  $l$  possible locations for the movements to the left, and the remaining movement are to the right.

$$\Rightarrow \binom{m}{\frac{m-x}{2}}$$

$\Rightarrow$  By 3.2.1., if  $\frac{m-x}{2} \notin \mathbb{Z}$ , then the point  $x$  is not a possible endpoint for an  $m$ -step

walk, and also in that case, the quantity  $\binom{m}{\frac{m-x}{2}}$  is undefined, so we multiply by the

indicator function.  $\square$

*Note:* See the appendix for counter.  $m$ , which verifies this formula for all cases.

### 3.2.3. Some Properties of $N^+(v)$ and $N^-(v)$

**Property 1.**  $r$  and  $l$  are the number of moves to the left and right, respectively

$$\sum_v N^+(v) - N^-(v) = r - l$$

*Proof.*  $N^+(v) - N^-(v)$  measures the net movement to the right of the site  $v$ ; when you sum all of these quantities you get the total net movement to the right,  $r - l$ .

*Note:* This property holds for any number of dimensions

**Property 2.** For an  $m$ -step standard random walk we have the following:

$$\sum_v N^+(v) + N^-(v) = m$$

*Proof.* For each step there is some contribution to either  $N^+(v)$  or  $N^-(v)$ . Therefore if you sum all of these contributions you get back the number of steps taken.

### 3.3. Cancellations

**Definition 3.3.1.** The number of cancellations,  $c$ , during an  $m$ -step standard random walk can be computed as follows:

$$c = \frac{1}{2}(m - \sum_v N^+(v) - N^-(v))$$

A cancellation occurs when you move back through a site in the opposite direction that you moved into the site before. That is, the  $N^+(v)$  cancelled out the  $N^-(v)$  in the quantity  $N^+(v) - N^-(v)$ . Thinking of it this way makes the structure of the formula for  $c$  above clear.

Since it takes two steps in order to make a cancellation, there can be at most the integer part of  $m$  cancellations in a walk. The number of cancellations in a walk has everything to do with the magnitude of the walk's contribution, and what coefficients the walk contributes to. We will see more about this in the next sections.

### 3.4. Loops

The only walks that contribute to the function  $h(\theta, x^*)$  are ones where the quantity  $N^+(x^*) - N^-(x^*)$  is nonzero. This only occurs when there are what we refer to as “loops” around the endpoint. By this I mean the walk reaches the endpoint,  $x^*$ , at some point before the  $m^{\text{th}}$  step. Then the walk moves away from the point  $x^*$  and returns in a pattern. Often times these loops come up as alternations between either  $x^* + 1$  and  $x^*$  or  $x^* - 1$  and  $x^*$  in the following way:

$$x^* \rightarrow x^* + 1 \rightarrow x^* \rightarrow x^* + 1 \rightarrow \dots \rightarrow x^*$$

In other cases the loops extend further such as the following:

$$x^* \rightarrow x^* + 1 \rightarrow x^* + 2 \rightarrow x^* + 1 \rightarrow x^* \rightarrow \dots \rightarrow x^*$$

The only case where looping does not produce a nonzero contribution is when the loop is symmetric about the endpoint, and therefore each of the endpoint’s contributions are being cancelled with each move.

$$x^* \rightarrow x^* - 1 \rightarrow x^* \rightarrow x^* + 1 \rightarrow \dots \rightarrow x^*$$

This does not contribute because  $N^+(x^*) = N^-(x^*)$

A couple of things to notice regarding looping:

- (1) The more cycles there are in a loop, the greater the contribution the walk gives to the polynomial. This is because terms of the form  $\sinh(k\theta)$  and  $\cosh(k\theta)$  contribute more to the polynomial than powers of  $\cosh$ .
- (2) A loop may have only one cycle
- (3) Walks which loop “to the left” give negative contributions to the polynomial. Walks which loop “to the right” give positive contributions.

### 3.5. An Example

Definition 3.1.3. and some of the other things above seem very complicated when written symbolically, so we’re going to work through an example to make the concepts a bit clearer.

#### **Example 1.**

Let  $m = 5$ ,  $x^* = 1$ . Notice that the only walks that will make a contribution to  $h(x^*, \theta)$  are ones where the quantity  $\{N^+(x^*) - N^-(x^*)\}$  is nonzero. So a walk such as:

$$0 \rightarrow -1 \rightarrow -2 \rightarrow -1 \rightarrow 0 \rightarrow 1$$

does not contribute to the function because  $N^+(x^*) = 0$  and  $N^-(x^*) = 0$ .

By the counting formula in Section 3.2.3., we have:

$N(5,1) = 10$ , so there are exactly 10 5-step walks that end at the point 1.

We will now enumerate each of them for the purposes of this example

$$\begin{aligned} \mathbf{0} &\rightarrow \mathbf{-1} \rightarrow \mathbf{-2} \rightarrow \mathbf{-1} \rightarrow \mathbf{0} \rightarrow \mathbf{1} & \mathbf{(1)} \\ \mathbf{0} &\rightarrow \mathbf{-1} \rightarrow \mathbf{0} \rightarrow \mathbf{-1} \rightarrow \mathbf{0} \rightarrow \mathbf{1} & \mathbf{(2)} \\ \mathbf{0} &\rightarrow \mathbf{1} \rightarrow \mathbf{0} \rightarrow \mathbf{-1} \rightarrow \mathbf{0} \rightarrow \mathbf{1} & \mathbf{(3)} \\ \mathbf{0} &\rightarrow \mathbf{-1} \rightarrow \mathbf{0} \rightarrow \mathbf{1} \rightarrow \mathbf{0} \rightarrow \mathbf{1} & \mathbf{(4)} \\ \mathbf{0} &\rightarrow \mathbf{1} \rightarrow \mathbf{0} \rightarrow \mathbf{1} \rightarrow \mathbf{0} \rightarrow \mathbf{1} & \mathbf{(5)} \\ \mathbf{0} &\rightarrow \mathbf{1} \rightarrow \mathbf{2} \rightarrow \mathbf{1} \rightarrow \mathbf{0} \rightarrow \mathbf{1} & \mathbf{(6)} \\ \mathbf{0} &\rightarrow \mathbf{-1} \rightarrow \mathbf{0} \rightarrow \mathbf{1} \rightarrow \mathbf{2} \rightarrow \mathbf{1} & \mathbf{(7)} \\ \mathbf{0} &\rightarrow \mathbf{1} \rightarrow \mathbf{0} \rightarrow \mathbf{1} \rightarrow \mathbf{2} \rightarrow \mathbf{1} & \mathbf{(8)} \\ \mathbf{0} &\rightarrow \mathbf{1} \rightarrow \mathbf{2} \rightarrow \mathbf{1} \rightarrow \mathbf{2} \rightarrow \mathbf{1} & \mathbf{(9)} \\ \mathbf{0} &\rightarrow \mathbf{1} \rightarrow \mathbf{2} \rightarrow \mathbf{3} \rightarrow \mathbf{2} \rightarrow \mathbf{1} & \mathbf{(10)} \end{aligned}$$

In walks (1), (2), (6), and (8),  $\{N^+(x^*) - N^-(x^*)\} = 0$ , so those walks do not contribute

**Walk (3):**

$$\{N^+(x^*) - N^-(x^*)\} = -1, \{N^+(-1) - N^-(-1)\} = 1, \{N^+(0) - N^-(0)\} = 1$$

$$\Rightarrow -\cosh^2(\theta) \sinh(\theta) \quad (\text{loops to the left})$$

**Walk (4):**

$$\{N^+(x^*) - N^-(x^*)\} = -1, \{N^+(-1) - N^-(-1)\} = 1, \{N^+(0) - N^-(0)\} = 1$$

$$\Rightarrow -\cosh^2(\theta) \sinh(\theta) \quad (\text{loops to the left})$$

**Walk (5):**

$$\{N^+(x^*) - N^-(x^*)\} = -2, \{N^+(0) - N^-(0)\} = 3$$

$$\Rightarrow -\cosh(3\theta) \sinh(2\theta) \quad (\text{loops to the left})$$

**Walk (7):**

$$\{N^+(x^*) - N^-(x^*)\} = 1, \{N^+(-1) - N^-(-1)\} = 1, \{N^+(2) - N^-(2)\} = -1$$

$$\Rightarrow \cosh^2(\theta) \sinh(\theta) \quad (\text{loops to the right})$$

**Walk (9):**

$$\{N^+(x^*) - N^-(x^*)\} = 2, \{N^+(0) - N^-(0)\} = 1, \{N^+(2) - N^-(2)\} = -1$$

$$\Rightarrow \cosh(\theta) \cosh(2\theta) \sinh(2\theta) \quad (\text{loops to the right})$$

**Walk (10):**

$$\{N^+(x^*) - N^-(x^*)\} = 1, \{N^+(0) - N^-(0)\} = 1, \{N^+(3) - N^-(3)\} = -1$$

$$\Rightarrow \cosh^2(\theta) \sinh(\theta) \quad (\text{loops to the right})$$

Now we sum up each of these products to get our function value

$$\begin{aligned} \Rightarrow h(x^*, \theta) &= -\cosh(3\theta) \sinh(2\theta) - 2 \cosh^2(\theta) \sinh(\theta) + 2 \cosh^2(\theta) \sinh(\theta) \\ &\quad + \cosh(\theta) \cosh(2\theta) \sinh(2\theta) \end{aligned}$$

$$\Rightarrow h(x^*, \theta) = -\cosh(3\theta) \sinh(2\theta) + \cosh(\theta) \cosh(2\theta) \sinh(2\theta)$$

$$\Rightarrow h(x^*, \theta) = -2 \cosh^2(\theta) (1 + 4 \sinh^2(\theta)) \sinh(\theta) + 2 \cosh^2(\theta) (1 + 2 \sinh^2(\theta)) \sinh(\theta)$$

$$\Rightarrow h(x^*, \theta) = \sinh(\theta) (-2 - 10 \sinh^2(\theta) - 8 \sinh^2(\theta) + 2 + 6 \sinh^2(\theta) + 4 \sinh^4(\theta))$$

$$\Rightarrow h(x^*, \theta) = -\sinh(\theta) (4 \sinh^2(\theta) + 4 \sinh^4(\theta))$$

Other examples involve similar derivations and get increasingly difficult with increased  $m$ . We saw how complicated this example got and we only had  $m = 5$ . One can see why a numerical computing approach is preferred in attacking these problems.

A few things to notice about the example:

- (1) The polynomial fits the form of (3.1.5.) with  $a_1 = a_2 = 4$
- (2) Some walks give a negative contribution to the coefficients but when we sum over all walks we get a positive net contribution
- (3) The walk which contributed to the leading coefficient had no cancellations

### 3.6. *Some Properties of the Polynomial for Special Cases*

Although our primary means to explore this problem is by numerical computing there are some facts we can get at in a theoretical sense. In particular for the special cases of  $x^* = 1$  and  $x^* = 2$ , for given  $m$  we can derive both the degree polynomial, and the general form of several of the coefficients. These forms provide a reliable check to see if the computer generated data is correct.

**Lemma 3.6.1.** For large  $\theta$ ,  $\sinh(\theta) \approx \cosh(\theta)$

*Proof.*

$$\cosh(\theta) = \frac{e^\theta + e^{-\theta}}{2}, \quad \sinh(\theta) = \frac{e^\theta - e^{-\theta}}{2},$$

but for large  $\theta$ ,  $e^{-\theta}$  is approximately 0, so

$$\cosh(\theta) \approx \sinh(\theta) \approx \frac{e^\theta}{2} \quad \square$$

**Lemma 3.6.2.** If  $r$  is the degree of the polynomial, then a walk with  $\{N^+(x^*) - N^-(x^*)\} \neq 0$  and  $c$  cancellations can contribute to the coefficients up to  $a_{r-c}$ .

*Proof.* By 3.3.1., if there are  $c$  cancellations, then

$$\sum_v N^+(v) - N^-(v) = m - 2c$$

Then, by the reasoning in Section 2.3. property 4, the quantity

$\prod_{v \neq x^*} \cosh(\{N^+(v) - N^-(v)\}\theta) \sinh(\{N^+(x^*) - N^-(x^*)\}\theta)$  can yield at most a polynomial of degree  $\text{floor}(\frac{m-2c-1}{2})$  (since its a polynomial in  $\sinh^2(\theta)$ ). Therefore the highest order

coefficient  $a_r$  is of the order  $\text{floor}(\frac{m-1}{2})$  and is only contributed to by walks with zero

cancellations; the next highest coefficient  $a_{r-1}$  is of order  $\text{floor}(\frac{m-3}{2})$  and is contributed

to by walks with less than or equal to 1 cancellation; continue reasoning in this way and its clear that the claim is true.  $\square$

**Property 1.** The polynomial corresponding to an  $m$ -step walk has degree  $\text{floor}(\frac{m-1}{2})$ .

*Proof.* Follows directly from the proof of Lemma (3.6.2.); notice that the degree is independent of the endpoint.

**Property 2.** For  $x^* = 1$ , the leading coefficient,  $a_r$ , is  $2^{m-3}$

*Proof.* By 3.6.2., the only walks which contribute to the highest order coefficient are those that have no cancellations. In one-dimension this means that the walk goes directly to the endpoint, and then loops to the left or right.

Suppose the walk loops to the right. Then our resulting contribution will be:

$$\cosh(\theta) \cosh\left(\left(\frac{m-1}{2}\right)\theta\right) \sinh\left(\left(\frac{m-1}{2}\right)\theta\right)$$

By 3.2.1.,  $m$  must be odd, therefore  $r = \text{floor}(\frac{m-1}{2}) = \frac{m-1}{2}$

So then we have:

$$\cosh(\theta) \cosh(r\theta) \sinh(r\theta)$$

The walks which loop to the left contribute in the following way:

$$-\cosh((r+1)\theta) \sinh(r\theta)$$

These are the only walks that contribute to the highest order coefficient, so we have:

$$(*) \quad \cosh(\theta) \cosh(r\theta) \sinh(r\theta) - \cosh((r+1)\theta) \sinh(r\theta) = -a_r \sinh^{2r+1}(\theta)$$

$$\Rightarrow \quad \frac{e^{(2r+1)\theta}}{2^3} - \frac{e^{(2r+1)\theta}}{2^2} = -a_r \frac{e^{(2r+1)\theta}}{2^{2r+1}} \quad (\text{by Lemma 3.6.1})$$

$$\Rightarrow \quad -\frac{1}{2^3} = -a_r \frac{1}{2^{2r+1}} \quad \Rightarrow \quad a_r = 2^{2r-2} = 2^{m-3} \quad \square$$

*Note:* The lower order terms are left out of equation (\*)

**Property 3.** For  $x^* = 2$ , the leading coefficient,  $a_r$ , is  $2^{m-4}$ .

*Proof.* Similarly to the previous proof, the walk must go directly to the end and then loop. Suppose the walk loops to the right, then our contribution is the following:

$$\cosh^2(\theta) \cosh\left(\left(\frac{m-2}{2}\right)\theta\right) \sinh\left(\left(\frac{m-2}{2}\right)\theta\right)$$

Since  $m$  must be even (by 3.2.1.),  $r = \text{floor}(\frac{m-2}{2}) = \frac{m-2}{2}$ , so we have:

$$\cosh^2(\theta) \cosh(r\theta) \sinh(r\theta)$$

If the walk loops to the left, then we have:

$$\begin{aligned} & \cosh(\theta) \cosh((\frac{m-2}{2} + 1)\theta) \sinh((\frac{m-2}{2})\theta) \\ \Rightarrow & \cosh(\theta) \cosh((r+1)\theta) \sinh(r\theta) \end{aligned}$$

These are the only two walks that contribute to  $a_r$ , so we have the following:

$$\begin{aligned} (*) & \cosh^2(\theta) \cosh((r+1)\theta) \sinh(r\theta) - \cosh(\theta) \cosh((r+1)\theta) \sinh(r\theta) = -a_r \sinh^{2r+2}(\theta) \\ \Rightarrow & \frac{e^{(2r+2)\theta}}{2^4} - \frac{e^{(2r+2)\theta}}{2^3} = -a_r \frac{e^{(2r+2)\theta}}{2^{2r+2}} \quad (\text{by Lemma 3.6.1}) \\ \Rightarrow & -\frac{1}{2^4} = -a_r \frac{1}{2^{2r+2}} \quad \Rightarrow \quad a_r = 2^{2r-2} = 2^{m-4} \quad \square \end{aligned}$$

*Note:* The lower order terms are left out of equation (\*)

These calculations get increasingly cumbersome as you analyze the lower order terms, and would take an excessive amount of space in this paper so I have decided to omit them. A similar process is used to solve for the lower order coefficients. First you determine what walks can contribute to that coefficient in accordance with Lemma (3.6.2.). Then you write out the form of each of the contributing walks, sum them up and set them equal to  $a_{r-k} \sinh^{2(r-k)+1}$  for odd  $x^*$  and  $a_{r-k} \sinh^{2(r-k)+2}$  for even  $x^*$ . Then use Lemma (3.6.1.) and solve for  $a_{r-k}$ .

**Property 4.** For  $x^* = 1$ , the coefficient  $a_{r-1}$  is  $r * 2^{m-4}$

**Property 5.** For  $x^* = 1$ , the coefficient  $a_{r-2}$  is  $2^{m-10}(19r^2 - 13r - 12)$

**Property 6.** For  $x^* = 1$ , the coefficient  $a_1$  is  $\binom{m-1}{\frac{m-3}{2}}$

**Property 7.** For  $x^* = 2$ , the coefficient,  $a_1$  is  $\binom{m}{\frac{m-4}{2}}$

*Note:* All of the general forms given above are nonnegative, in accordance with our conjecture.

## IV. THE TWO-DIMENSIONAL CASE

This section contains the focus of my research and most of the material here was either derived by me or is a natural analog to the 1-D case.

### 4.1. Description of the Problem

**Definition 4.1.1.** In two dimensions our definition of a standard random walk amounts to the following:

$$\begin{aligned}
 X_0 &= (0, 0) \quad x, y \in \mathbb{Z}, n = 0, 1, 2, \dots \\
 P(X_{t+1} = (x+1, y) \mid X_t = (x, y)) &= P(X_{t+1} = (x-1, y) \mid X_t = (x, y)) = \\
 P(X_{t+1} = (x, y+1) \mid X_t = (x, y)) &= P(X_{t+1} = (x, y-1) \mid X_t = (x, y)) = \frac{1}{4}
 \end{aligned}$$

Or perhaps more clearly:

$$\begin{aligned}
 (x, y) &\xrightarrow{p=1/4} (x+1, y), (x, y) \xrightarrow{p=1/4} (x, y+1) \\
 (x, y) &\xrightarrow{p=1/4} (x-1, y), (x, y) \xrightarrow{p=1/4} (x, y-1)
 \end{aligned}$$

**Definition 4.1.2.** Let  $v = (x, y)$ ,  $x, y \in \mathbb{Z}$ , then we define  $N^+(v)$  and  $N^-(v)$  in two dimensions as follows:

$N^+(v)$  : The number of times during the walk there was a move from  $(x, y)$  to  $(x+1, y)$

$N^-(v)$  : The number of times during the walk there was a move from  $(x, y)$  to  $(x-1, y)$

*Note:* In general, if  $v$  is an ordered  $n$ -tuple,  $N^+(v)$  and  $N^-(v)$  can be thought of as the number of times a walk moves up or down, respectively, one in the  $x$ -direction from the site  $v$ .

We can now move to the 2-dimensional analog of the function of interest. Like the 1-D case, this function can be written in the form of Equation 3.1.4. The primary difference in the multi-dimensional analysis of this function is that now the walks analyzed do not have to end at a particular point. Instead, we are analyzing walks that end on the line  $x = x^*$ . For instance, if our  $x^*$  is 1, one of our walks may end at  $(1, 0)$  while another walk may end at  $(1, 2)$ . The only stipulation is that the  $x$  coordinate is  $x^*$ .

**Definition 4.1.3.** For  $\theta \in \mathbb{R}$ ,  $v \in S$ , our function over all  $m$ -step walks that end on the line  $x = x^*$  is given by:

$$h(x^*, \theta) = \sum_{\substack{\text{WALKS} \\ X(0)=(0,0) \\ X(m)=(x^*, y)}} \prod_{v \neq x^*} \cosh(\{N^+(v) - N^-(v)\}\theta) \sinh(\{N^+(x^*) - N^-(x^*)\}\theta)$$

*Note:* The  $y$  referred to in the subscript of the summation,  $X(m) = (x^*, y)$ , is just some  $y \in \{-m, -m+1, -m+2, \dots, m-2, m-1, m\}$ .

Although it is beyond the scope of this paper, this formula can readily be extended into  $n$  dimensions by starting the walk at the  $n$ -dimensional vector  $\mathbf{0}$  and having it end at some ordered  $n$ -tuple where the first entry in the vector is  $x^*$  and the rest of the entries  $\in \{-m, -m+1, -m+2, \dots, m-2, m-1, m\}$ . The conjecture is hypothesized to hold for any  $n$ .

## 4.2. Some Properties and Miscellaneous

One important difference between the 2-D case and the 1-D case is the addition of another parameter  $k$ , the number of horizontal steps. The 1-D case can be viewed as a special case of the 2-D where  $k = m$ . Most of the 2-D analogs of the properties in 1-D involve this extra parameter.

### 4.2.1. Accessibility of an endpoint on an $m$ -step walk with $k$ horizontal steps

If an  $m$ -step walk with  $k$  horizontal steps ends on the line  $x = x$ , then  $k - x \equiv 0 \pmod{2}$

*Proof.* See the proof of Section 2 – Property 1

### 4.2.2. Counting the $m$ -step Walks with $k$ horizontal steps that end at $x$

$$N(m, k, x^*) = 2^{m-k} \binom{m}{k} \binom{k}{\frac{x+k}{2}} I\left\{\frac{x+k}{2} \in \mathbf{Z}\right\}$$

*Proof.* There are  $m$  total steps, and  $k$  possible locations for the horizontal steps, hence the

$\binom{m}{k}$ , furthermore for each vertical step there are two possible ways the step could

go, hence the  $2^{m-k}$ . So now there are  $k$  spaces left unaccounted for.

If  $r$  is the number of horizontal moves to the right, and  $l$  is the number of moves to the left, then:

$$\begin{aligned} r + l &= k, & r - l &= x \\ \Rightarrow r &= \frac{k+x}{2}, & l &= \frac{k-x}{2} \end{aligned}$$

So there are  $\binom{k}{\frac{x+k}{2}}$

possible locations for the movements to the right; the remaining moves are to the left.

$$\Rightarrow 2^{m-k} \binom{m}{k} \binom{k}{\frac{x+k}{2}}$$

Also note that Property 1 above requires that  $\frac{x+k}{2} \in \mathbf{Z}$ , hence the indicator function added onto the end.

$$\Rightarrow 2^{m-k} \binom{m}{k} \binom{k}{\frac{x+k}{2}} I\{\frac{x+k}{2} \in \mathbf{Z}\} \quad \square$$

*Note:* See count.  $m$  in the appendix for a program which directly counts the number of walks; the output of the program agrees with this formula in all cases

#### 4.2.3. Some Properties of $N^+(v)$ and $N^-(v)$

**Property 1.** For an  $m$ -step 2-D walk with  $k$  horizontal steps we have the following:

$$\sum_v N^+(v) + N^-(v) = k$$

*Proof.* See the proof of section 3.2.3., Property 2.

*Note:* Section 3.2.3., Property 1, also holds for the 2-D case.

Also note that all of the discussion about Loops and Cancellations in Sections 3.3. and 3.4. carries over almost verbatim to the 2-D case. The only difference is that in the discussion about cancellations, replace  $m$  with  $k$ ; this gives:

$$c = \frac{1}{2}(k - \sum_v N^+(v) - N^-(v))$$

#### 4.3. An Example

The computations in 2-D are a bit more difficult so I will do a rather simple example to try and illustrate how this works in 2-D.

##### **Example 1.**

Let  $m = 4$ ,  $k = 3$ ,  $x^* = 1$ .

By the counting formula 4.2.3., the total number of walks that meet these criteria is 24.

By looking at these walks with my program I can see that there are only two possible ways that we could have a walk that reaches the line and still gives  $\{N^+(x) - N^-(x)\} \neq 0$ .

One such walk is one where you have  $\{N^+(x) - N^-(x)\} = 1$ , and you have two locations that have  $\{N^+(v) - N^-(v)\} = 1$ . One such walk looks like:

$$(0,0) \rightarrow (0,1) \rightarrow (1,1) \rightarrow (2,1) \rightarrow (1,1)$$

In this case  $x = (1,1)$ , and we have:

$$\{N^+(x) - N^-(x)\} = 1$$

$$\{N^+((0,1)) - N^-((0,1))\} = 1$$

$$\{N^+((2,1)) - N^-((2,1))\} = -1$$

This gives  $\cosh^2(\theta) \sinh(\theta)$

The other case is where  $\{N^+(x) - N^-(x)\} = -1$ , and  $\{N^+(v) - N^-(v)\} = 2$  for some location. One such walk looks like:

$$(0,0) \rightarrow (0,1) \rightarrow (1,1) \rightarrow (0,1) \rightarrow (1,1)$$

In this case  $x = (1,1)$ , and we have:

$$\{N^+(x) - N^-(x)\} = -1$$

$$\{N^+((0,1)) - N^-((0,1))\} = 2$$

This gives  $-\cosh(2\theta)\sinh(\theta)$

Using my program I found that there are exactly 2 walks that fit each of these scenarios, so this gives:

$$h(x^*, \theta) = -2\cosh(2\theta)\sinh(\theta) + 2\cosh^2(\theta)\sinh(\theta)$$

$$\Rightarrow = -2(\sinh(\theta)(1 + 2\sinh^2(\theta)) + 2((1 + \sinh^2(\theta))\sinh(\theta))$$

$$\Rightarrow = -4\sinh^3(\theta) - 2\sinh(\theta) + 2\sinh(\theta) + 2\sinh^3(\theta)$$

$$\Rightarrow = -2\sinh^3(\theta) = -\sinh(\theta)(2\sinh^2(\theta))$$

Notice that the one of the two walks gives a negative contribution but the net contribution is positive, in accord with the hypothesis.

#### 4.4. *Some Properties of the Polynomial for Special Cases*

Deriving the properties of the polynomial is much more difficult in 2-D but I was able to derive some facts that are pertinent and provide a check for the computer generated data. Note that Lemma 3.6.1 and 3.6.2. both hold for the two dimensional case as well. Also, when  $k = m$ , everything in section 3.6. holds for the 2-D case.

**Property 1.** The polynomial corresponding to an  $m$ -step walk has degree  $\text{floor}(\frac{k}{2})$ .

*Proof.* Follows directly from the proof of Lemma (3.6.2.) but replace  $m$  with  $k$ ; notice that the degree is independent of the endpoint.

**Property 2.** For  $x^* = 1$ ,  $k = m - 1$ , the leading coefficient,  $a_r$ , is  $2^{k-4}(k+1)$

*Proof.* By 3.6., the only walks which contribute to the leading coefficient are ones which have zero cancellations; that is:

$$\sum_v N^+(v) - N^-(v) = k$$

In one dimension this meant the walk went straight to the endpoint and then looped. But, in 2-D this isn't necessary true. For example:

$$(0,0) \rightarrow (-1,0) \rightarrow (-1,1) \rightarrow (0,1) \rightarrow (1,1) \rightarrow (2,1) \rightarrow (1,1)$$

In that walk we have that  $\sum_v N^+(v) - N^-(v) = k$ , but the walk obviously does not go

straight to the endpoint and loop. This is an example of when the 1-D projection of the walk would contain a cancellation, but in 2-D this is not the case. This is why the process of finding the highest order coefficient in 2-D is very different from 1-D.

By inspection I saw that every walk which satisfies the criteria in Property 2 fits one of the two following forms:

$$(1) \quad \cosh^{2z+1}(\theta) \cosh((r-z)\theta) \sinh((r-z)\theta)$$

$$(2) \quad \cosh^{2z}(\theta) \cosh((r-z+1)\theta) \sinh((r-z)\theta)$$

Also by inspection, walks of form (1) give positive contributions to the walk, walks of form (2) give negative contributions. In fact, walks of form (2) appear to form a bijection with walks of form (1) where the form (2) walks are simply the walks which loop to the left, and walks of form (1) loop to the right. I am unable to prove this but using an algorithm to count these walks for given  $m, k$ , I found that for all checked cases the number of form (1) equaled the number of form (2).

$r$  is the degree of the polynomial,  $z = 0, 1, \dots, r-1$

Notice that for any  $z$ , we have that  $m = 2r+1$ , as it should when  $x^* = 1$

Let  $f(z)$  denote the contribution of a walk of form (1) for given  $z$ .

Let  $h(z)$  denote the contribution of a walk of form (2) for given  $z$ .

$z$	0	1	2	...	$r-1$
$f(z)$	$2^{k-3}$	$2^{k-5}$	$2^{k-7}$		1
$h(z)$	$-2^{k-2}$	$-2^{k-4}$	$-2^{k-6}$		-2
$f(z) + h(z)$	$-2^{k-3}$	$-2^{k-5}$	$-2^{k-7}$		-1

Since we have this pattern and  $f(z) + h(z) = -f(z)$ , and by the reasoning above there are the same number of walks of each form for given  $z$ , so we can entirely capture the behavior of the sum by looking only at walks of form (1). Now all this is left is to count these walks.

If  $N(z)$  is the number of walks that fit the pattern for given  $z$  then we have:

$$a_r = \sum_{i=0}^{r-1} N(i) f(i)$$

But if you look you can see that walks of the form (2) with  $z = 0$  can only occur when the vertical step is first, so there is 2 walks that are of that form (since the vertical step can either be up or down). Walks of the form (2) with  $z = 1$  can only occur when the vertical step is 2<sup>nd</sup> or 4<sup>th</sup>, so there are 4 possible walks. By counting walks of these form (1) with a computer algorithm it appears that for given  $z$ , there are  $2^{2z}$  walks that fit the form. The exception is for  $z = 0$ , where the number is 2. Therefore we have:

$$\begin{aligned}
a_r &= 2 * 2^{k-3} + \sum_{i=0}^{r-1} 2^{2i} * 2^{k-3-2i} & \Rightarrow & & a_r &= 2 * 2^{k-3} + \sum_{i=0}^{\frac{k-3}{2}} 2^{2i} * 2^{k-3-2i} \\
\Rightarrow & & a_r &= & 2^{k-2} &+ \sum_{i=0}^{\frac{k-3}{2}} 2^{k-3} \\
\Rightarrow & & a_r &= & 2^{k-2} &+ 2^{k-4}(k-3) \\
\Rightarrow & & a_r &= & 2^{k-4} &(4+k-3) \\
\Rightarrow & & a_r &= & 2^{k-4} &(k+1) \quad \square
\end{aligned}$$

*Note:* The above form of  $a_r$  is always nonnegative

I also have made a couple observations that I was not able to prove but still feel are important to mention.

**Observation 1.** If the nonzero coefficients are symmetric in the 1-D case for given  $m = a$  and  $x^* = x$ , then the coefficients in the 2-D with  $k = a$  and  $x^* = x$  are also symmetric, regardless of  $m$ .

**Observation 2.** If the nonzero coefficients are equal in the 1-D case for give  $m = a$  and  $x^* = x$ , then the coefficients in the 2-D with  $k = a$  and  $x^* = x$  are also equal, regardless of  $m$ .

## V. METHODS

The primary methods used in this study were numerical methods which involve programs in Matlab. There are separate algorithms for the 1-dimensional and 2-dimensional cases. The 1-D program takes two inputs;  $n$ : the length of the walk and POSITION: the desired end position ( $x^*$  above). The 2-dimensional program takes 3 inputs;  $n$ : the length of the walk,  $h$ : the number of horizontal steps taken on the walk, and POSITION: the designated location on the  $x$ -axis that the walk ends. The following is the code for each of the two programs with comments to explain what each part is doing.

```

% Beginning of the 1-dimensional program
% AUTHOR: Joe Conlon
% define a 2^n by n matrix which contains all possible
% increments of random walks of length

```

```

function y=rw1(n, POSITION)
D= ones(2^n, n);
for j=1:n
for i=2^(j-1)+1: 2^j
D(i, :)=D(i-2^(j-1), :);
D(i, j)=-1;
end
end

```

```

% position of the walk started at zero determined by a row of D is
% given by the corresponding row of B

```

```

B= zeros(2^n, n+1);

```

```

for i=1: 2^n
for j=2: n+1
B(i, j)=B(i, j -1)+D(i, j -1);
end
end

% list in the 2n+1 dimensional ith row of W the quantity
%  $N_{(+)}(v) - N_{(-)}(v)$  for each site  $v$  the walk visits for
% the walk corresponding to the  $i$ th row of  $B$ ,  $-n \leq v \leq n$ ,
% with zero corresponding to the  $(n+1)$ st entry of  $W$ 

W=zeros(2^n, 2*n+1);
for i=1: 2^n
for j=1: n
W(i, n+1+B(i, j))=W(i, n+1+B(i, j))+D(i, j);
end
end

% the input is a vector V of length 2n+1 and an integer x
% where  $-n \leq x \leq n$ . the program computes the product of
%  $\cosh(V(j)z)$ , where  $j$  is not equal to  $n+1+x$ , times
%  $\sinh(V(n+1+x)z)$  as a polynomial in odd powers of  $\sinh(z)$ 
% or  $\cosh(z)$  times a polynomial in odd powers of  $\sinh(z)$ .

L=floor((n+1)/2);

if 2*L== n+1
L = L;
else L=L+1;
end

m=L-1;

N(1)=1;
for i=1: m
N(i +1)=N(i)+i +1;
end
a=zeros(1, N(m+1)); a(1)=1;
b=zeros(1, N(m+1)); b(1)=1;
c=zeros(1, N(m+1)); c(1)=2; c(2)=4; c(3)=8;
d=zeros(1, N(m+1)); d(1)=1; d(2)=1; d(3)=2;

for k=1: m
a(N(k+1))=2*a(N(k))+2*b(N(k));
a(N(k+1)-k)=a(N(k)+1-k)+2*b(N(k)+1-k);
b(N(k+1))=2*a(N(k))+2*b(N(k));
b(N(k+1)-k)=b(N(k)+1-k);

for j=1: k-1
a(N(k+1)-j)=2*a(N(k)-j)+2*b(N(k)-j) + a(N(k)+1-j)+2*b(N(k)+1-j);
b(N(k+1)-j)=2*a(N(k)-j)+2*b(N(k)-j) + b(N(k)+1-j);
end

end

for k=2: m
d(N(k+1))=2*c(N(k-1))+2*d(N(k));
d(N(k+1)-k)=d(N(k)+1-k);
d(N(k+1)+1-k)=2*c(N(k-1)+2-k)+2*d(N(k)+1-k)+d(N(k)+2-k);

for j=1: k-2
d(N(k+1)-j)=2*c(N(k-1)-j)+2*d(N(k)-j) + 2*c(N(k-1)+1-j)+d(N(k)+1-j);
end

```

```

c(N(k+1))=2*c(N(k))+2*d(N(k+1));
c(N(k+1)-k)=c(N(k)+1-k)+2*d(N(k+1)-k);

for j=1:k-1
c(N(k+1)-j)=2*c(N(k)-j)+2*d(N(k+1)-j) + c(N(k)+1-j);
end

end

% finding the polynomial corresponding to each of the 2^n random walks

A=zeros(1,L); p=0; SIGN=0; ABSVAL=0; COUNT=0; DEG=0;
x=0;
V=zeros(1,2*n+1); DUMVAR=0; OUTPUT=zeros(2*n+2,L);
SUM=0;

for INDEX=1:2^n

V=W(INDEX,:);
x=B(INDEX,n+1);
SIGN=sign(V(n+1+x));
ABSVAL=abs(V(n+1+x));
V=abs(V);
A=zeros(1,L);
DEG=0;
COUNT=0;

if SIGN==0
A=A;
else

% storing the polynomial representation of sinh(V(n+1+x)z)
% COUNT counts the extra power of cosh(z) entering in.
% DEG gives the degree of the polynomial in powers of sinh^2(z)

p=floor(V(n+1+x)/2); COUNT=0; DEG=0;

if V(n+1+x)== 0
A=A;

elseif V(n+1+x)== 1
A(1)=1;

elseif V(n+1+x)== 2
A(1)=2; COUNT=1;

elseif p== V(n+1+x)/2
COUNT=1; DEG=p-1;
for j=1: p
A(j)=c(N(p-1)+j);
end

else
DEG=p;
for j=1: p+1
A(j)=a(N(p)+j);
end

end

end

% computing the product of cosh(V(j)z), j not equal to
% n+1+x, with sinh(V(n+1+x))

```

```

DUMVAR=0;
V(n+1+x)=0;

for j=1: 2*n+1

p=floor(V(j)/2);

if V(j)==0
A=A;

elseif V(j)==1
COUNT=COUNT+1;

elseif p==V(j)/2
DEG=DEG+p;

for i=DEG+1: -1: 1
DUMVAR=0;
for k=max(1, i-p): i
DUMVAR=DUMVAR+A(k)*d(N(p)+i+1-k);
end
A(i)=DUMVAR;
end

else
COUNT=COUNT+1;
DEG=DEG+p;

for i=DEG+1: -1: 1
DUMVAR=0;
for k=max(1, i-p): i
DUMVAR=DUMVAR+A(k)*b(N(p)+i+1-k);
end
A(i)=DUMVAR;
end
end

% multiplying by cosh(z) raised to power COUNT
SUM=floor(COUNT/2);

for i=1: SUM
DEG=DEG+1;
A(DEG+1)=A(DEG);
for k=DEG: -1: 2
A(k)=A(k)+A(k-1);
end
end

OUTPUT(n+1+x, :)= OUTPUT(n+1+x, :)+SIGN*A;
end
end

for i=n+2: 2*n+1
OUTPUT(2*n+2, :)= OUTPUT(2*n+2, :)+ (i-n-1)*OUTPUT(i, :);
end
OUTPUT=-OUTPUT;
y=OUTPUT(n+1+POSITION, :)

% End of 1-dimensional program

```

```

% Beginning of the 2 dimensional program
% AUTHOR: Jason Goldstick
% The following takes inputs n for number of steps, h for number
% of horizontal steps, and POSITION for the end location on the
% x axis. The output is a polynomial in sinh(theta) where the
% coefficients are given in a vector of the form [a_0, a_1, ..., a_r]
% where r is the degree of the polynomial

function y=rw2(n, h, POSITION)

u=zeros(4^n, 2+2*n); v=zeros(4^n, 2+2*n); c=3; p=4^(n-1);

%the following generates all possible walks

while(p>=1)
    t=0; s=0;
    while(t<(4^n))
        for k=1:p
            u(s+k, c: c+1)=[1, 0]; t=t+1;
        end
        for k=(p+1): 2*p
            u(s+k, c: c+1)=[0, 1]; t=t+1;
        end
        for k=(2*p+1): 3*p
            u(s+k, c: c+1)=[-1, 0]; t=t+1;
        end
        for k=(3*p+1): 4*p
            u(s+k, c: c+1)=[0, -1]; t=t+1;
        end
        s=s+k;
    end
    c=c+2; p=p/4;
end

% DX and DY contain the movement along the x and y axis for each move
DX=u(:, 3: 2: 2+2*n); DY=u(:, 4: 2: 2+2*n);

% v contains the position of the walk after each step
for j=3: (2+2*n)
    v(:, j)=v(:, j-2)+u(:, j);
end

%X and Y contain the x and y coordinates of the walk after each step
X=v(:, 1: 2: 2+2*n); Y=v(:, 2: 2: 2+2*n);

% iiy contains the index of each walk that has h horizontal movements
% and ends on the line x=POSITION
dx=abs(DX'); horiz=sum(dx(:, :)); ix=(horiz==h); iix=find(ix==1);
iy=find(X(iix, n+1)==POSITION); iiy=iix(iy);

% finds the endpoint on the y-axis of the specified walk
for i=1: length(iiy)
    ENDPPOINT(i)=Y(iiy(i), n+1);
end

% SINH contains the quantity sinh(r*theta) in the r'th row and is
% written in terms of a polynomial in sinh(theta) where the columns
% denote the coefficients [a0, a1, a2, ...] where a_r is the coefficient
% of sinh^r. For even r, there is a leftover cosh(theta)
SINH=[0, 1, 0, 0, 0, 0, 0; 0, 2, 0, 0, 0, 0, 0; 0, 3, 0, 4, 0, 0, 0; 0, 4, 0, 8, 0, 0, 0; 0, 5, 0, 20,
... , 0, 16, 0; 0, 6, 0, 32, 0, 32, 0];

```

```

% COSH contains the quantity cosh(r*theta) in the r'th row and is
% written in terms of a polynomial in sinh(theta) similarly to the
% above. For odd r, there is a leftover cosh(theta)
COSH=[1, 0, 0, 0, 0, 0, 0; 1, 0, 2, 0, 0, 0, 0; 1, 0, 4, 0, 0, 0, 0; 1, 0, 8, 0, 8, 0, 0; 1, 0, 12, 0,
... , 16, 0, 0; 1, 0, 18, 0, 48, 0, 32];

```

```

% COSHTOPOWER contains the quantity cosh(theta)^r in the r'th row and
% is written in terms of a polynomial in sinh(theta) similarly to the
% above. For odd r, there is a leftover cosh(theta)
COSHTOPOWER=[0, 0, 0, 0, 0, 0, 0; 1, 0, 1, 0, 0, 0, 0; 1, 0, 1, 0, 0, 0, 0; 1, 0, 2, 0, 1, 0, 0; 1,
... , 0, 2, 0, 1, 0, 0; 1, 0, 3, 0, 3, 0, 1];

```

```

POLY=zeros(length(iiy), 21);

```

```

%following loop computes the product of cosh(v) for all v * sinh(x)

```

```

for INDEX=1:length(iiy)
    V=zeros(2*n+1, 2*n+1);
    % finds the quantity N+(v)-N-(v) for each site visited on the
    % specified walk
    for i=1:n
        V(X(iiy(INDEX), i)+n+1, Y(iiy(INDEX), i)+n+1)=V(X(iiy(INDEX), i)+n+1,
        ... , Y(iiy(INDEX), i)+n+1)+DX(iiy(INDEX), i);
    end

```

```

    EXTRACOSH=0; PRODUCT=zeros(1, 21);
    ENDCONTRIBUTION=zeros(1, 7);
    END=V(n+1+POSITION, n+1+ENDPOINT(INDEX));
    V=abs(V); CONTRIBUTION=zeros(((2*n+1)^2)+2, 21);

```

```

    if(END == 0)
        POLY=POLY;
    else
        k=1;

```

```

        % finds the contribution of each site on the walk

```

```

        for i=-n:1:n
            for j=-n:1:n
                if( (V(n+1+i, n+1+j) > 1) & ((sum( [n+1+i, n+1+j] ==
                ... [n+1+POSITION, n+1+ENDPOINT(INDEX)] ) / 2) ~= 1) )
                    CONTRIBUTION(k, 1:7)=COSH(V(n+i+1, n+j+1), :);
                    k=k+1;
                else
                    ;
                end
                if(rem(V(n+i+1, n+j+1), 2)==1)
                    EXTRACOSH=EXTRACOSH+1;
                else
                    EXTRACOSH=EXTRACOSH;
                end
            end
        end

```

```

        % finds the contribution of the endpoint

```

```

        if( END < 0 )
            ENDCONTRIBUTION=SI NH(-END, :);
        elseif( END > 0 )
            ENDCONTRIBUTION=SI NH(END, :);
        end

```

```

CONTRIBUTION(2*(2*n+1)+1, 1: 7)=ENDCONTRIBUTION;

if(abs(rem(END, 2))==1)
    EXTRACOSH=EXTRACOSH;
else
    EXTRACOSH=EXTRACOSH+1;
end

% finds contribution from the leftover cosh(theta) terms

EXTRACOSH=2*floor(EXTRACOSH/2);
if(EXTRACOSH ~= 0)
    EXTRA=COSHTOPOWER(EXTRACOSH, :);
else
    EXTRA=zeros(1, 7);
end
CONTRIBUTION(2*(2*n+1)+2, 1: 7)=EXTRA;

ix=find( (sum(CONTRIBUTION') ~= 0) );
CONTRIBUTION=CONTRIBUTION(ix, :);

% computes the product of all the contributing polynomials

PRODUCT=CONTRIBUTION(1, :);
s=size(CONTRIBUTION);
for z=2: s(1)
    b=zeros(1, 21);
    for i=0: 6
        for j=0: max(find(PRODUCT~=0))-1
            b(i+j+1)=b(i+j+1)+CONTRIBUTION(z, i+1)*PRODUCT(j+1);
        end
    end
    PRODUCT=b;
end

end

% gives the correct sign to the contribution

if(END < 0)
    PRODUCT=-PRODUCT;
elseif(END > 0)
    PRODUCT=PRODUCT;
end

POLY(INDEX, :)=PRODUCT;

end

% sums over all walks

COEFFICIENTS=sum(POLY);
ix=max(find(COEFFICIENTS ~= 0));
COEFFICIENTS=COEFFICIENTS(1: ix);

y=-COEFFICIENTS;

% End of two dimensional program

```

In order to minimize the chance of being mistakenly believing that the program's coefficients are correct when in fact they are not, I have written several checking algorithms for various parts of the program which can be found in the appendix. Also in the appendix is an alternate version of the 2-D program which outputs the exact same coefficients in every case.

## VI. CONCLUSIONS

The scope of this project was to make sure that the numerical evidence does not directly contradict the hypothesis, rather than trying to prove the hypothesis which is quite an arduous task that has not been done yet. However, from my findings I certainly cannot prove the hypothesis wrong, as all of the computations that are within the capabilities of Matlab all output positive coefficients in both the 1-D and 2-D cases. See the appendix for this data.

## REFERENCES

1. Conlon, Joseph. Personal meetings, conversations/advising sessions. May through July 2005.
2. Ross, Sheldon. Introduction to Probability Models. 8<sup>th</sup> Edition. Academic Press. San Diego, CA. 2003. p.181-184

## APPENDIX

```
% Beginning of the alternate 2-dimensional program
% AUTHOR: Jason Goldstick and Joe Conlon
% The locations of the walk are denoted in rows of length
% 2+2*n, and consist of ordered pairs(x, y)

function y=rw3(n, h, POSITION)
u=zeros(4^n, 2+2*n); v=zeros(4^n, 2+2*n); c=3; p=4^(n-1);

% The following generates all possible walks

while(p>=1)
    t=0; s=0;
    while(t<(4^n))
        for k=1:p
            u(s+k, c: c+1)=[1, 0]; t++;
        end
        for k=(p+1): 2*p
            u(s+k, c: c+1)=[0, 1]; t++;
        end
        for k=(2*p+1): 3*p
            u(s+k, c: c+1)=[-1, 0]; t++;
        end
        for k=(3*p+1): 4*p
            u(s+k, c: c+1)=[0, -1]; t++;
        end
        s=s+k;
    end
    c=c+2; p=p/4;
end
```

% The loop below generates the locations of the walk after % each step

```
for k=1: 4^n
    for j=3: (2+2*n)
        v(k, j)=v(k, j -2)+u(k, j);
    end
end
```

```
DX=u(:, 3: 2: 2+2*n); DY=u(:, 4: 2: 2+2*n);
X=v(:, 1: 2: 2+2*n); Y=v(:, 2: 2: 2+2*n);
```

% DX and DY contain the movement along the x and y axis for each move.  
% X and Y contain the x and y coordinates before each move

```
for i=1: 4^n
    horiz=0;
    for j=2: n+1
        if(Y(i, j) == Y(i, j -1))
            horiz=horiz+1;
        else
            horiz=horiz;
        end
    end
    if(horiz==h)
        ix(i)=1;
    else
        ix(i)=0;
    end
end
```

```
ix=find(ix==1); iy=find(X(ix, n+1)==POSITION);
iiy=ix(iy);
```

% iiy contains the locations of each walk that has h  
% horizontal movements and ends at POSITION

```
for i=1: length(iiy)
    ENDPOINT(i)=Y(iiy(i), n+1);
end
```

% records the end location of the walk,

% the following 33 lines computes the function h(POSITION, theta)

```
L=floor((h+1)/2);
if(2*L == h+1)
    L=L;
else
    L=L+1;
end
```

```
m=L-1; N(1)=1;
for i=1: m
    N(i+1)=N(i)+i+1;
end
```

```
a=zeros(1, N(m+1)); a(1)=1; b=zeros(1, N(m+1)); b(1)=1; c=zeros(1, N(m+1));
c(1)=2; c(2)=4; c(3)=8; d=zeros(1, N(m+1)); d(1)=1; d(2)=1; d(3)=2;
```

```
for k=1: m
```

```
    a(N(k+1))=2*a(N(k))+2*b(N(k));
    a(N(k+1)-k)=a(N(k)+1-k)+2*b(N(k)+1-k);
    b(N(k+1))=2*a(N(k))+2*b(N(k));
```

```

b(N(k+1)-k)=b(N(k)+1-k);

for j=1:k-1
a(N(k+1)-j)=2*a(N(k)-j)+2*b(N(k)-j) +a(N(k)+1-j)+2*b(N(k)+1-j);
b(N(k+1)-j)=2*a(N(k)-j)+2*b(N(k)-j) + b(N(k)+1-j);
end

end

for k=2:m

d(N(k+1))=2*c(N(k-1))+2*d(N(k)); d(N(k+1)-k)=d(N(k)+1-k);
d(N(k+1)+1-k)=2*c(N(k-1)+2-k)+2*d(N(k)+1-k)+d(N(k)+2-k);

for j=1:k-2
d(N(k+1)-j)=2*c(N(k-1)-j)+2*d(N(k)-j) + 2*c(N(k-1)+1-j)+d(N(k)+1-j);
end

c(N(k+1))=2*c(N(k))+2*d(N(k+1));
c(N(k+1)-k)=c(N(k)+1-k)+2*d(N(k+1)-k);

for j=1:k-1
c(N(k+1)-j)=2*c(N(k)-j)+2*d(N(k+1)-j) + c(N(k)+1-j);
end

end

A=zeros(1,L); p=0; SIGN=0; ABSVAL=0; COUNT=0; DEG=0; x=0;
V=zeros(2*n+1,2*n+1); DUMVAR=0; OUTPUT=zeros(2*n+2,L); SUM=0;

for INDEX=1:length(iiy) % runs through all walks h horizontal steps
V=zeros(2*n+1,2*n+1);

% The following computes N+(v)-N-(v) for each vertice on the walk
for j=1:n
V(X(iiy(INDEX),j)+n+1,Y(iiy(INDEX),j)+n+1)=...
=V(X(iiy(INDEX),j)+n+1,Y(iiy(INDEX),j)+n+1)+DX(iiy(INDEX),j);
end

SIGN=sign(V(n+1+POSITION,n+1+ENDPOINT(INDEX)));
ABSVAL=abs(V(n+1+POSITION,n+1+ENDPOINT(INDEX)));
V=abs(V); A=zeros(1,L); DEG=0; COUNT=0;

if(SIGN==0)
A=A;
else
p=floor(V(n+1+POSITION,n+1+ENDPOINT(INDEX))/2);
if(V(n+1+POSITION,n+1+ENDPOINT(INDEX))==0)
A=A;
elseif(V(n+1+POSITION,n+1+ENDPOINT(INDEX))==1)
A(1)=1;
elseif(V(n+1+POSITION,n+1+ENDPOINT(INDEX))==2)
A(1)=2; COUNT=1;
elseif(p==(V(n+1+POSITION,n+1+ENDPOINT(INDEX))/2))
COUNT=1; DEG=p-1;
for j=1:p
A(j)=c(N(p-1)+j);
end
else
DEG=p;
for j=1:p+1
A(j)=a(N(p)+j);
end
end
end
end

```

```

DUMVAR=0; V(n+1+POSITION, n+1+ENDPOINT(INDEX))=0;
for i=1: 2*n+1
    for j=1: 2*n+1
        p=floor(V(i, j)/2);
        if(V(i, j)==0)
            A=A;
        elseif(V(i, j)==1)
            COUNT=COUNT+1;
        elseif(p==V(i, j)/2)
            DEG=DEG+p;
            for t=DEG+1: -1: 1
                DUMVAR=0;
                for k=max(1, t-p): t
                    DUMVAR=DUMVAR+A(k)*d(N(p)+t+1-k)
                end
                A(t)=DUMVAR;
            end
        else
            COUNT=COUNT+1; DEG=DEG+p;
            for t=DEG+1: -1: 1
                DUMVAR=0;
                for k=max(1, t-p): t
                    DUMVAR=DUMVAR+A(k)*b(N(p)+t+1-k);
                end
                A(t)=DUMVAR;
            end
        end
    end
end
end
SUM=floor(COUNT/2);
for i=1: SUM
    DEG=DEG+1;
    A(DEG+1)=A(DEG);
    for k=DEG: -1: 2
        A(k)=A(k)+A(k-1);
    end
end
OUTPUT(n+1+POSITION, :)=OUTPUT(n+1+POSITION, :)+SIGN*A;
end

for i=n+2: 2*n+1
    OUTPUT(2*n+2, :)= OUTPUT(2*n+2, :)+ (i-n-1)*OUTPUT(i, :);
end
OUTPUT=-OUTPUT; y=OUTPUT(n+1+POSITION, :)

% beginning of counter.m
% AUTHOR: Jason Goldstick
function y=counter(n, x)
% generates all walks
D=ones(2^n, n);
for j=1: n
    for i=2^(j-1)+1: 2^j
        D(i, :)=D(i-2^(j-1), :);
        D(i, j)=-1;
    end
end
end

% finds the endpoint of each walk
B=zeros(2^n, n+1);
for i=1: 2^n
    for j=2: n+1
        B(i, j)=B(i, j-1)+D(i, j-1);
    end
end

```

```

end

% counts the number of walks that end at the location x
for j=1:2^n
    if(B(j, n+1)==x)
        ix(j)=1;
    else
        ix(j)=0;
    end
end

% computes function value from Eq. 1
if(rem(((m-x)/2), 1)==0)
    N=(nchoosek(m, (m-x)/2));
else
    N=0;
end

% tests if Eq. 1 agrees with the direct count
y=(sum(ix)==N);

% end of program

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% beginning of count. m
% AUTHOR: Jason Goldstick
function y=count(n, h, x)
u=zeros(4^n, 2+2*n);
v=zeros(4^n, 2+2*n); c=3; p=4^(n-1);

% generates all walks
while(p>=1)
    t=0; s=0;
    while(t<(4^n))
        for k=1:p
            u(s+k, c:c+1)=[1, 0];
            t=t+1;
        end
        for k=(p+1):2*p
            u(s+k, c:c+1)=[0, 1];
            t=t+1;
        end
        for k=(2*p+1):3*p
            u(s+k, c:c+1)=[-1, 0];
            t=t+1;
        end
        for k=(3*p+1):4*p
            u(s+k, c:c+1)=[0, -1];
            t=t+1;
        end
        s=s+k;
    end
    c=c+2; p=p/4;
end

% The loop below generates the locations of the walk after each step
for k=1:4^n
    for j=3:(2+2*n)
        v(k, j)=v(k, j-2)+u(k, j);
    end
end

```

```

D=u(:, 3: 2: 2+2*n); B=v(:, 1: 2: 2+2*n);
Y=v(:, 2: 2: 2+2*n);
for i=1: 4^n
    hori z=0;
    for j=2: n+1
        i f(B(i, j) ~= B(i, j -1))
            hori z=horiz+1;
        el se
            hori z=horiz;
        end
    end
    i f(horiz==h)
        i x(i)=1;
    el se
        i x(i)=0;
    end
end

% iix counts the number that have h horizontal steps and ends at x

% directly computes Eq. 4
iix=find(i x==1);
m=n; z=x; N=0; k=h;
i f((rem(((1/2)*(k+z)), 1)==0) & ( (k>=z) ) )
    N=( 2^(m-k) * nchoosek(m, m-k) * nchoosek(k, (1/2)*(k+z)) );
el se
    N=0;
end

% tests i f the direct counting equals Eq. 4, many different trials all
% output the value 1 (true)

y=(sum((B(i i x, n+1)==x)==N);

% end of program

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% beginning of check.m
% AUTHOR: Jason Goldstick
function y=check(n, h, POSITION)
u=zeros(4^n, 2+2*n);
v=zeros(4^n, 2+2*n); c=3; p=4^(n-1);

% generates all possible walks
while(p>=1)
    t=0; s=0;
    while(t<(4^n))
        for k=1: p
            u(s+k, c: c+1)=[1, 0]; t++;
        end
        for k=(p+1): 2*p
            u(s+k, c: c+1)=[0, 1]; t++;
        end
        for k=(2*p+1): 3*p
            u(s+k, c: c+1)=[-1, 0]; t++;
        end
        for k=(3*p+1): 4*p
            u(s+k, c: c+1)=[0, -1]; t++;
        end
        s=s+k;
    end
    c=c+2; p=p/4;
end

```

```

DX=u(:, 3: 2: 2+2*n); DY=u(:, 4: 2: 2+2*n);
X=v(:, 1: 2: 2+2*n); Y=v(:, 2: 2: 2+2*n);

% computes N+(v)-N-(v) for each site
W=zeros(length(iiy), 2*n+1, 2*n+1); THETA=1;
for i=1: 4^n
    for j=1: n
        W(i, X(i, j)+n+1, Y(i, j)+n+1)=W(i, X(i, j)+n+1, Y(i, j)+n+1)+DX(i, j);
    end
end

% directly computes function value
THETA=1;
for i=1: 4^n
    ENDPPOINT=[POSITION, Y(i, n+1)]; c=0;
    for j=-n: 1: n
        for k=-n: 1: n
            c=c+1;
            if( mean([i, j]==ENDPOINT)==1 )
                P(i, c)=1;
            else
                P(i, c)=cosh(W(i, j+n+1, k+n+1) * THETA);
            end
        end
    end
    F(i)=sinh( W(i, n+1+POSITION, n+1+Y(iiy(i))) * THETA ) * prod(P(i, :));
end

FUNCTIONVALUE = sum(F);

poly=rw3(n, h, POSITION);
ix=find(u!=0);
poly=poly(ix);

% computes the function value using the polynomial
for j=1: length(poly)
    POLYNOMIAL = POLYNOMIAL + poly(j)*(sinh(THETA))^(1+2*(j-1))
end

% tests the two quantities for equality
y=(FUNCTIONVALUE == POLYNOMIAL)

%%%%%%%%%%

```

The following is the 1-D coefficients  
m: length of walk  
x: point on the x-axis where the walk ends  
Note: Matlab will not run in the 1-D case if POSITION > 20

x=1	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
3	1								
5	4	4							
7	15	24	16						
9	56	120	128	64					
11	210	564	796	640	256				
13	792	2568	4480	4752	3072	1024			
15	3003	11460	23828	31702	26496	14336	4096		
17	11440	50416	121984	188480	197120	140800	65536	16384	
19	43758	219396	606956	1085488	1340816	1173504	721920	294912	65536

x=2

m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
4	1								
6	6	4							
8	28	36	16						
10	120	228	192	64					
12	495	1248	1532	960	256				
14	2002	6312	10168	9232	4608	1024			
16	8008	30396	60652	71936	51840	21504	4096		
18	31824	141604	337296	493808	463616	276992	98304	16384	
20	125970	644208	1784896	3110752	3598160	2795520	1426423	442368	65536

x=3

m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
5	1	1							
7	7	12	4						
9	37	85	64	16					
11	175	504	568	320	64				
13	781	2721	4064	3404	1536	256			
15	3367	13840	25820	28432	19024	7168	1024		
17	14196	67528	151644	207128	181392	101248	32768	4096	
19	58956	319548	841388	1375752	1488048	1084160	519680	147456	16384

x=4

m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
6	1	1							
8	8	14	4						
10	46	110	76	16					
12	232	696	752	384	64				
14	1092	3940	5740	4572	1856	256			
16	4928	20836	38116	40760	25808	8704	1024		
18	21624	105184	231788	309764	262880	138368	39936	4096	
20	93024	513216	1325336	2123936	2247376	1584384	714240	180224	16384

x=5

m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
7	1	2	1						
9	9	25	20	4					
11	56	194	227	104	16				
13	299	1231	1900	1416	512	64			
15	1471	7000	13412	13804	8124	2432	256		
17	6884	37176	85088	111292	90304	44048	11264	1024	
19	31161	188466	501929	794620	817856	549552	229248	51200	4096

x=6

m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
8	1	2	1						
10	10	28	22	4					
12	67	236	272	116	16				
14	378	1592	2436	1728	576	64			
16	1941	9486	18113	18024	10028	2752	256		
18	9402	52260	119662	152888	119288	54800	12800	1024	
20	43795	272860	728924	1135188	1135900	731840	286848	58368	4096

x=7

m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
9	1	3	3	1					
11	11	42	55	28	4				
13	79	361	603	449	144	16			
15	470	2486	5079	5136	2712	704	64		
17	2517	15087	36509	46991	35192	15212	3328	256	
19	12616	84430	236353	369939	359488	220896	81104	15360	1024

x=8									
m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
10	1	3	3	1					
12	12	46	60	30	4				
14	92	424	706	514	156	16			
16	576	3084	6296	6240	3152	768	64		
18	3214	19554	47394	60030	43428	17852	3648	256	
20	16664	113416	318600	492584	466028	275384	95824	16896	1024

x=9									
m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
11	1	4	6	4	1				
13	13	63	115	97	36	4			
15	106	598	1309	1400	751	184	16		
17	697	4467	11461	15175	11108	4456	896	64	
19	4048	28986	85461	136044	128362	73540	24668	4224	256

x=10									
m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
12	1	4	6	4	1				
14	14	68	124	104	38	4			
16	121	686	1501	1592	836	196	16		
18	834	5384	13826	18152	13004	5024	960	64	
20	5036	36392	107538	169928	157259	87264	28044	4544	256

x=11									
m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
13	1	5	10	10	5	1			
15	15	88	206	244	151	44	4		
17	137	917	2486	3502	2721	1133	224	16	
19	988	7406	22911	37980	36575	20712	6648	1088	64

x=12									
m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
14	1	5	10	10	5	1			
16	16	94	220	260	160	46	4		
18	154	1034	2804	3936	3030	1238	236	16	
20	1160	8736	27056	44680	42568	23264	7344	1152	64

x=13									
m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
15	1	6	15	20	15	6	1		
17	17	117	334	510	445	217	52	4	
19	172	1330	4299	7536	7750	4702	1595	264	16

x=14									
m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
16	1	6	15	20	15	6	1		
18	18	124	354	540	470	228	54	4	
20	191	1480	4786	8376	8575	5156	1720	276	16

x=15									
m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
17	1	7	21	35	35	21	7	1	
19	19	150	505	944	1065	734	295	60	4

x=16									
m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
18	1	7	21	35	35	21	7	1	
20	20	158	532	994	1120	770	308	62	4

x=17									
m	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
19	1	8	28	56	70	56	28	8	1

x=18  
m a\_1 a\_2 a\_3 a\_4 a\_5 a\_6 a\_7 a\_8 a\_9  
20 1 8 28 56 70 56 28 8 1

%%

The following is the 2-D coefficients

m: length of walk

k: # of horizontal steps

x: point on x-axis where walk ends

Note: Matlab will not run if POSITION > 10

A few points to notice:

- (1) Whenever k=m, the coefficients are identical to those for an m-step one dimensional walk
- (2) Whenever the coefficients generated by rw2(n, h, x) are equal, the coefficients generated by rw1(h, x) are also equal
- (3) Whenever the coefficients generated by rw2(n, h, x) are symmetric, the coefficients generated by rw1(h, x) are also symmetric
- (4) Polynomials generated by rw2(n, h, x) have the same degree as rw1(h, x)

case 1. k=m

x=1

m	a_0	a_1	a_2	a_3	a_4	a_5
3	0	1				
5	0	4	4			
7	0	15	24	16		
9	0	56	120	128	64	

x=2

m	a_0	a_1	a_2	a_3	a_4	a_5
4	0	1	0			
6	0	6	4	0		
8	0	28	36	16	0	
10	0	120	228	192	64	0

x=3

m	a_0	a_1	a_2	a_3	a_4	a_5
5	0	1	1			
7	0	7	12	4		
9	0	37	85	64	16	

x=4

m	a_0	a_1	a_2	a_3	a_4	a_5
6	0	1	1	0		
8	0	8	14	4	0	
10	0	46	110	76	16	0

x=5

m	a_0	a_1	a_2	a_3	a_4	a_5
7	0	1	2	1		
9	0	9	25	20	4	

x=6

m	a_0	a_1	a_2	a_3	a_4	a_5
8	0	1	2	1	0	
10	0	10	28	22	4	0

x=7						
m	a <sub>0</sub>	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>	a <sub>4</sub>	a <sub>5</sub>
9	0	1	3	3	1	

x=8						
m	a <sub>0</sub>	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>	a <sub>4</sub>	a <sub>5</sub>
10	0	1	3	3	1	0

case 2. k=m-1

x=1						
m	a <sub>0</sub>	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>	a <sub>4</sub>	a <sub>5</sub>
4	0	2				
6	0	12				
8	0	60	104	64		
10	0	280	680	720	320	

x=2						
m	a <sub>0</sub>	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>	a <sub>4</sub>	a <sub>5</sub>
5	0	4	0			
7	0	30	26	0		
9	0	168	272	144	0	

x=3						
m	a <sub>0</sub>	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>	a <sub>4</sub>	a <sub>5</sub>
6	0	6	6			
8	0	52	96	40		
10	0	320	832	736	224	

x=4						
m	a <sub>0</sub>	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>	a <sub>4</sub>	a <sub>5</sub>
7	0	8	8	0		
9	0	78	144	54	0	

x=5						
m	a <sub>0</sub>	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>	a <sub>4</sub>	a <sub>5</sub>
8	0	10	20	10		
10	0	108	308	268	68	

x=6						
m	a <sub>0</sub>	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>	a <sub>4</sub>	a <sub>5</sub>
9	0	12	24	12	0	

x=7						
m	a <sub>0</sub>	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>	a <sub>4</sub>	a <sub>5</sub>
10	0	14	42	42	14	

case 3. k=m-2

x=1						
m	a <sub>0</sub>	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>	a <sub>4</sub>	a <sub>5</sub>
5	0	10				
7	0	90	90			
9	0	600	1060	622		

x=2						
m	a <sub>0</sub>	a <sub>1</sub>	a <sub>2</sub>	a <sub>3</sub>	a <sub>4</sub>	a <sub>5</sub>
6	0	18	0			
8	0	198	162	0		
10	0	1476	2328	1118	0	

x=3						
m	a_0	a_1	a_2	a_3	a_4	a_5
7	0	30	30			
9	0	348	644	272		

x=4						
m	a_0	a_1	a_2	a_3	a_4	a_5
8	0	46	46	0		
10	0	570	1064	422	0	

x=5						
m	a_0	a_1	a_2	a_3	a_4	a_5
9	0	66	132	66		

x=6						
m	a_0	a_1	a_2	a_3	a_4	a_5
10	0	90	180	90	0	

case 4. k=m-3

x=1						
m	a_0	a_1	a_2	a_3	a_4	a_5
6	0	22				
8	0	256	256			
10	0	2100	3792	2202		

x=2						
m	a_0	a_1	a_2	a_3	a_4	a_5
7	0	58				
9	0	786	698	0		

x=3						
m	a_0	a_1	a_2	a_3	a_4	a_5
8	0	118	118			
10	0	1692	3192	1416		

x=4						
m	a_0	a_1	a_2	a_3	a_4	a_5
9	0	210	210	0		

x=5						
m	a_0	a_1	a_2	a_3	a_4	a_5
10	0	342	684	342		

case 5. k=m-4

x=1						
m	a_0	a_1	a_2	a_3	a_4	a_5
7	0	74				
9	0	1164	1164			

x=2						
m	a_0	a_1	a_2	a_3	a_4	a_5
8	0	190	0			
10	0	3360	2928	0		

case 6. k=m-5

x=1						
m	a_0	a_1	a_2	a_3	a_4	a_5
8	0	166				
10	0	3162	3162			

x=2  
m 9      a\_0      a\_1      a\_2      a\_3      a\_4      a\_5  
         0      546      0

x=3  
m 10      a\_0      a\_1      a\_2      a\_3      a\_4      a\_5  
         0      1398      1398

case 7. k=m-6

x=1  
m 9      a\_0      a\_1      a\_2      a\_3      a\_4      a\_5  
         0      474

x=2  
m 10      a\_0      a\_1      a\_2      a\_3      a\_4      a\_5  
         0      1566      0

case 8. k=m-7

x=1  
m 10      a\_0      a\_1      a\_2      a\_3      a\_4      a\_5  
         0      1062