Low Discrepancy Sequences and Quasi-Monte Carlo Integration

Erin Scott
SUNY Fredonia
Fredonia, NY
scot6856@ginko.ait.fredonia.edu

Dennis Simmons
University of California, Davis
Davis, CA
dgsimmons@ucdavis.edu

Ian Winokur
College of Mount Saint Vincent
Riverdale, NY
iwinokur@cmsv.edu

Faculty Advisors
Prof. Robert Burton and Prof. Thomas Schmidt

REU Program
Oregon State University
August 15, 1997

Chapter 1

Quasi-Monte Carlo Integration: A Good Gamble

1.1 Numerical Approximation of Integrals

Every polynomial can be integrated exactly on any interval $[a, b]$ where a and b are real. Unfortunately, there is more to life than polynomials. In fact, most functions cannot be integrated exactly. This is the reason numerical approximation methods are necessary.

Riemann Sums, the Trapezoidal Rule, and Simpson's Rule are several approximation methods. These methods usually give adequate approximations and they are very practical in lower dimensions.

However, each increase in dimension results in an exponential growth of the number of points that must be evaluated to attain any given degree of accuracy (such points will be referred to as 'nodes') [5] . (Note: The term "function call" is used to indicate each time a node is evaluated on some function.) This occurrence has been dubbed the "curse of dimensionality" [5]. Since function calls are very costly in terms of computer time, this "curse" can be quite troublesome.

1.2 Monte Carlo Integration

In the late 1940s, Monte Carlo Integration was developed as a method to foil the "curse of dimensionality." (References on the history of Monte Carlo

Integration can be found in [5].) The main benefit of Monte Carlo Integration is due to the fact that each node being used in the approximation method requires only one function call, regardless of dimension. Monte Carlo Integration represents a significant step toward improving the efficiency of approximating the integral of a function in higher dimensions.

We will now discuss two different Monte Carlo Methods. The first hinges on the fact that $\int_a^b f(x)\,dx$ gives us the area under f and between a and b. We will first give a simplified analogy of this Monte Carlo technique before we relate it to functions. This analogy was discussed in [4].

Let us suppose we want to calculate the area of a pond which is enclosed inside a polygon of known area, *poly* (see fig 1-1).

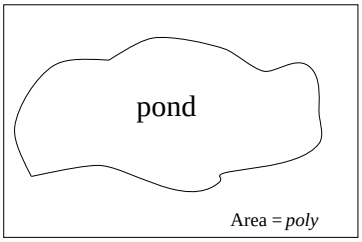

Figure 1-1

Let us then"randomly" toss n pebbles into the polygon (see figure 1-2). (Note: The word 'randomly' is in quotation marks because I am using a very loose definition of the word because generating truly random numbers on a computer is nearly impossible. I will omit the quotation marks from now on but this caveat remains in effect for the duration of the paper.)

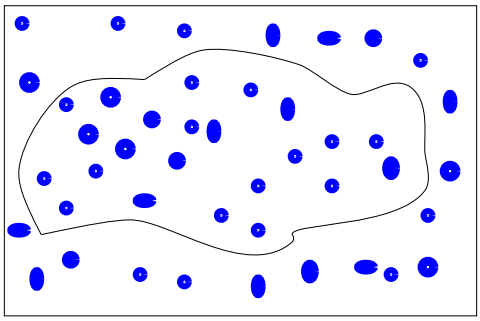

Figure 1-2

Let s be the number of pebbles that land in the pond (the letter 's' is chosen to represent the word "splash"). Intuitively, it is clear that the area of the pond is approximately equal to the area of the polygon (namely, *poly*) multiplied by the fraction of pebbles that land in the pond (namely, s/n). Therefore,

Area of the pond $\approx poly \cdot (s/m)$.

Let us now extend this concept to functions. Assume we want to integrate some function, f, on the interval $[a, b]$. Also, let h be some upper bound of f on $[a, b]$.

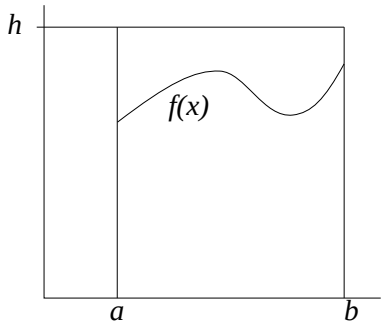

Figure 1-3

We again randomly toss pebbles, this time making sure that the pebbles land in the rectangle formed by joining the points $(a, 0)$, $(b, 0)$, (b, h), (a, h) (see figure 1-3). Let n be the number of pebbles tossed and let s denote the number of "splashes." We increment the number of splashes each time a pebble lands under the curve. This translates into the following line of pseudocode: if $f(x_i) \geq y_i$, then $s = s + 1$ where (x_i, y_i) are the coordinates of the ith pebble tossed where $1 \leq i \leq n$. The area under f can now be approximated in the same manner that the area of the pond was approximated. Thus,

$$\int_a^b f(x)\,dx \approx h(b-a)(s/n).$$

Of course, tossing in three pebbles (or any other ridiculously small number of pebbles) is not likely to give a good approximation. A significant number of pebbles usually needs to be used to attain an acceptable level of accuracy. This will be discussed more concretely in the section on error bounds.

In order to actually see how accurate this method is, we wrote a program (called mc1.c) that can be found in the 'programs' section of this paper.

Another Monte Carlo Integration method uses the following definition:

Definition 1.1 *The average value of a function, f, on the interval $[a, b]$ is defined to be:*

$$\frac{1}{b-a}\int_a^b f(x)\,dx.$$

Our strategy here is to come up with an intuitive definition for the average value of f, and then equate this intuitive notion with the actual definition. What we are going to do is randomly select n nodes that all lie in $[a, b]$. We will call each node x_i where $1 \leq i \leq n$. To find an approximation of the average value of f, we will evaluate each x_i, sum up these $f(x_i)$, and then divide by n. Therefore, the average value of f on $[a, b]$ is approximated by:

$$\frac{f(x_1) + f(x_2) + ... + f(x_n)}{n} = \frac{1}{n}\sum_{i=1}^{n} f(x_i).$$

Equating this with the actual definition yields:

$$\frac{1}{b-a}\int_a^b f(x)\,dx \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i).$$

We now multiply both sides by $(b-a)$ to obtain a second Monte Carlo Approximation Method:

$$\boxed{\int_a^b f(x)\,dx \approx \frac{b-a}{n}\sum_{i=1}^{n} f(x_i)}.$$

This method is utilized in the program mc2.c and can also be found in the 'programs' section. The following charts give the results of mc1.c and

function	a	b	n	area	mc1	error	mc2	error
$f(x) = x$	0	1	20	0.50	0.450	10.0	0.485	2.92
			50		0.560	12.0	0.459	8.21
			100		0.560	12.0	0.484	3.25
			1000		0.499	0.20	0.491	1.81
			10000		0.500	0.18	0.506	1.11
$f(x) = x^3$+2x-3	2	5	20	164.25	178.200	8.49	159.338	2.99
			50		213.840	30.19	148.878	9.36
			100		186.120	13.32	153.728	6.41
			1000		167.112	1.74	161.587	1.62
			10000		164.498	0.15	166.444	1.34
$f(x) = \cos x$	0	$\frac{\pi}{2}$	20	1.00	0.785	21.46	1.023	2.35
			50		1.037	3.67	1.071	7.13
			100		1.147	14.67	1.048	4.85
			1000		1.000	0.06	1.012	1.22
			10000		1.002	0.15	0.990	1.02

Table 1.1: Approximations using Monte Carlo Methods 1 and 2

mc2.c using various functions which can be integrated exactly. While these functions are rather simple, they are useful here because they show just how accurate Monte Carlo Methods can be.

Table 1.1 shows that Monte Carlo Methods can be quite accurate with large enough n.

However, Monte Carlo Integration has two distinct downfalls, as Niederreiter points out in [5]. First of all, because the nodes selected are chosen randomly, there is no guaranteed error bound. In many cases, Monte Carlo Methods produce accuracy that is more than sufficient, but the fact of the matter is that its error bound is only probabilistic.

Another problem with these methods is the fact that the accuracy of

the approximation is highly dependent upon how truly random the random nodes are. Usually, computers are used to generate these random nodes and it has been found that creating truly random numbers with a computer is an extraordinarily difficult task. Because it is nearly impossible for computers to generate random numbers and because of the lack of a guaranteed error bound, Monte Carlo methods became less and less attractive in favor of other approximation techniques that did not have these two downfalls. This brings us to a discussion of quasi-Monte Carlo Integration.

1.3 Quasi-Monte Carlo Integration

Quasi-Monte Carlo Integration originated in the 1950s as an attempt to overcome the disadvantages of Monte Carlo Integration. Quasi-Monte Carlo Integration was so named because, with the exception of how nodes are selected, it is exactly the same as the second Monte Carlo Method discussed above. The nodes used in quasi-Monte Carlo Integration are selected because they are expected to outperform randomly selected nodes.

Some definitions are now in order to make the criteria for selection of nodes for quasi-Monte Carlo Integration more precise. The following definitions are excerpted from [7].

Definition 1.2 *Let* $\{x_1, x_2, \ldots\}$ *be a sequence of points contained in the interval* $[0, 1)$. *Then this sequence is uniformly distributed if for any interval* $[a, b)$ *where* $0 \leq a < b < 1$

$$\lim_{N \to \infty} \frac{\#\{x_i \in [a, b) \mid 1 \leq i \leq N\}}{N} = (b - a).$$

Essentially, this says that a sequence is uniformly distributed if every interval in $[0, 1)$ gets its "fair share" of points from the sequence as the number of points in the sequence gets large.

Definition 1.3 *Let* $\{x_1, x_2, \ldots x_N\}$ *be a sequence in* $[0, 1)$. *Then the discrepancy of the sequence is defined as*

$$D_N = \sup_{0 \leq \alpha \leq \beta \leq 1} \left| \frac{\#\{x_i \epsilon [\alpha, \beta) s.t. 1 \leq i \leq N\}}{N} - (\beta - \alpha) \right|.$$

The above definition was also excerpted from [4].

To put it another way, discrepancy measures the largest difference between the number of points in any given interval and the length of that interval. Discrepancy will rise when elements in $\{x_1, x_2, \ldots\}$ are "very close together" or when they are "spread out too much." The lower the discrepancy, the more "evenly spaced" the elements in the sequence.

It turns out that nodes from uniformly distributed sequences with low discrepancy are the best ones to choose when using quasi-Monte Carlo Integration. When nodes of this type are used, quasi-Monte Carlo Integration overcomes some of the downfalls of Monte Carlo Integration.

The following advantages of quasi-Monte Carlo Integration are discussed in [5]. Since the nodes chosen are now deterministic (as opposed to probabilistic), a deterministic error bound exists. Also, there is no need to generate random numbers, the difficulty of which was discussed earlier. Thirdly, a much higher degree of accuracy can be reached with the same number of function calls as Monte Carlo Integration. These three advantages are a tremendous improvement over Monte Carlo Methods.

An application of quasi-Monte Carlo Integration using the same premise as mc2.c can be found in the 'programs' section under the heading qmc1.c. In testing out this program, I found that quasi-Monte Carlo Integration gave the exact results for linear functions when a particular type of uniformly distributed low discrepancy sequence was used. This observation led to the following theorem:

Theorem 1.1 *Lan's Theorem: If f is a non-negative linear function then quasi-Monte Carlo Integration yields the exact answer to $\int_a^b f(x)\,dx$ when the nodes used are of the form $\{a, a+c, a+2c, a+3c, ..., b-2c, b-c, b\}$ where c divides $b-a$.*

Proof: Let $f(x) = mx + d$ where $m \in \Re$ and $d \in \Re$. Either $f(a) = 0$ or $f(a) \geq 0$. If $f(a) = 0$ then the region under f is a triangle (see Figure 1-4) with

$$Area = \frac{1}{2} f(b)(b-a) = \frac{b-a}{2}(mb+d).$$

If $f(a) \geq 0$ then the region under f is a trapezoid (Figure 1-5) with

$$Area = \frac{1}{2}(b-a)(f(a)+f(b)) = \frac{b-a}{2}(ma+d+mb+d) = \frac{b-a}{2}(m(a+b)+2d).$$

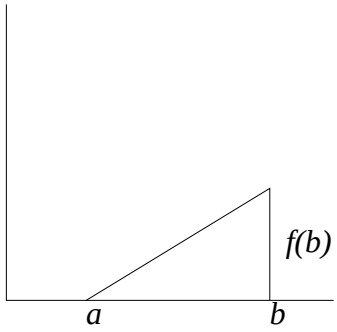

Figure 1-4

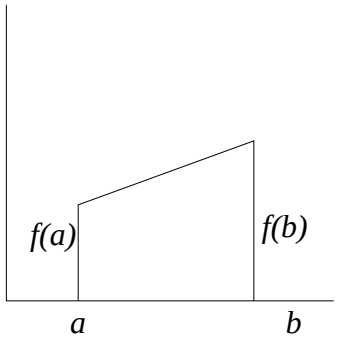

Figure 1-5

If $f(a) = 0$ then $ma + d = 0$. This information gives us the fact that the two area formulas are essentially the same as the only difference between them is the term $ma + d$.

It therefore suffices to show that quasi-Monte Carlo Integration yields the formula for the area of the trapezoid. So we want to show that

$$\frac{b-a}{n}\sum_{i=1}^{n} f(x_i) = \frac{b-a}{2}(m(a+b)+2d).$$

Recall that our nodes are of the form: $\{a, a+c, a+2c, a+3c, ..., b-2c, b-c, b\}$ where $x_1 = a$ and $x_n = b$. Therefore, $b = a + c(n-1)$ and $x_i = a + c(i-1)$ where $1 \leq i \leq n$.

We will now use quasi-Monte Carlo Integration on f.

$$\frac{b-a}{n}\sum_{i=1}^{n} f(x_i)$$

$$= \frac{b-a}{n}\sum_{i=1}^{n} f(a+c(i-1))$$

$$= \frac{b-a}{n}\sum_{i=1}^{n}(m(a+c(i-1))+d)$$

$$= \frac{b-a}{n}\sum_{i=1}^{n}(ma-cm+d)+icm)$$

$$= \frac{b-a}{n}(\sum_{i=1}^{n}(ma-cm+d)+mc\sum_{i=1}^{n} i)$$

$$= \frac{b-a}{n}(n(ma-cm+d)+\frac{mcn(n+1)}{2}$$

$$= \frac{b-a}{2}(2(ma-cm+d)+mc(n+1))$$

$$= \frac{b-a}{2}(2ma-2cm+2d+mcn+mc)$$

$$= \frac{b-a}{2}(m(2a-c+cn)+2d)$$

$$= \frac{b-a}{2}(m(a+a+c(n-1))+2d)$$

$$= \frac{b-a}{2}(m(a+b)+2d).$$

Since this formula obtained by using quasi-Monte Carlo Integration is the same as the formula for the area of the trapezoid, the proof is complete. ■

Table 1.2 presents a side-by-side comparison of five different approximation methods. Three of them have been discussed in detail in this paper. The other two are the Trapezoidal Rule and Simpson's Rule, both of which were

$f(x) = 4x + 2$ on $[0, 1]$. Exact area = 4.00

n	mc1	error	mc2	error	qmc1	error	simpson	error	trap	error
100	4.38	9.50	3.93	1.63	4.02	0.50	3.99	0.00	3.99	0.00
1000	4.01	0.35	3.96	1.59	4.00	0.05	3.99	0.00	3.99	0.00
10000	4.01	0.25	4.02	0.55	4.00	0.00	4.00	0.00	4.00	0.00

$f(x) = 3x^3 - x + 1$ on $[1, 3]$. Exact area = 58.00

n	mc1	error	mc2	error	qmc1	error	simpson	error	trap	error
100	67.94	17.14	53.25	8.20	59.25	2.15	57.99	0.00	58.00	0.00
1000	58.78	1.34	56.97	1.77	58.12	0.21	58.00	0.00	58.00	0.00
10000	57.65	0.60	58.91	1.57	58.00	0.01	58.02	0.03	58.02	0.03

$f(x) = x^5 + 3x^2 + 12$ on $[2, 5]$. Exact area = 2746.50

n	mc1	error	mc2	error	qmc1	error	simpson	error	trap	error
100	2987.16	8.76	2403.26	12.50	2838.21	3.34	2746.50	0.00	2746.73	0.01
1000	2736.62	0.36	2698.10	1.76	2755.50	0.33	2746.45	0.00	2746.46	0.00
10000	2730.84	0.57	2800.85	1.98	2746.87	0.01	2744.91	0.06	2744.91	0.06

Table 1.2: Approximations using five different methods

put into the 'programs' section. The nodes used in the quasi-Monte Carlo Integration are of the form $a, a + c, a + 2c, ..., b - 2c, b - c, b$ where c divides $b - a$.

From the results in Table 1.2, Simpson's Rule appears to be the best choice as it is generally the most accurate. (Note: we found that Simpson's Rule was usually more accurate than the Trapezoidal Rule by about the fourth digit after the decimal point.) However, it is noted in [4] that the accuracy of Simpson's Rule outweighs its large number of function calls only in one or two dimensions. In higher dimensions, [4] states that the "curse of dimensionality" becomes significant enough to make Monte Carlo and quasi-Monte Carlo Methods more attractive than Simpson's Rule. Again, the main

reason this is true is the low number of costly function calls that Monte Carlo and quasi-Monte Carlo Methods require.

1.4 Probability and Error Bounds

No discussion of approximation methods would be complete without a discussion of error bounds. We will first give a brief crash course in some probability theory.

A sample space is a list of possible outcomes of some experiment. For example, the sample space of two successive flips of a coin is {HH, HT, TH, TT} where H and T stand for Heads and Tails, respectively. Ω is generally used as the symbol for the sample space and ω is often used to represent an element of the sample space.

A random variable, X, is a function that assigns a numerical value to each $\omega \in \Omega$. An example of a random variable is $X(\omega) =$ the age (in years) of ω where Ω is a list of people. If Ω is a list of the people in this year's REU program then the random variable, X, given above, would satisfy:

$X(Michael) = 21.$
$X(Jill) = X(Margaret) = 19.$
$X(Jessica) = X(Erin) = X(Dennis) = X(Kevin) = X(Ian) = 20.$

The expected value of X is defined as

$$E(X) = \sum_{\omega \in \Omega} X(\omega) P(\{\omega\})$$

where $P(\{\omega\}) =$ the probability that ω occurs.

The mean of X is a special type of expected value. The mean is represented by μ and is defined to be $\mu = E(X)$. In the above example,

$$\mu = E(X) = \sum_{\omega \epsilon \Omega} X(\omega) P(\{\omega\}) = X(Jessica)P(\{Jessica\}) + X(Erin)P(\{Erin\}) +$$

$$X(Dennis)P(\{Dennis\}) + X(Kevin)P(\{Kevin\}) + X(Ian)P(\{Ian\}) +$$

$$X(Michael)P(\{Michael\}) + X(Jill)P(\{Jill\}) + X(Margaret)P(\{Margaret\})$$

$$= 20(\frac{1}{8}) + 20(\frac{1}{8}) + 20(\frac{1}{8}) + 20(\frac{1}{8}) + +20(\frac{1}{8})$$
$$+21(\frac{1}{8}) + 19(\frac{1}{8}) + 19(\frac{1}{8}) = 19.875.$$

Note: In this example, $P(\{\omega\}) = \frac{1}{8}$ for all $\omega \in \Omega$ because each person was listed only once and there were eight people altogether.

In the above example, it is clear that μ gives the average age of the group of people in question.

The variance, σ^2, is defined as:

$$\sigma^2 = E((X-E(X))^2) = E(X^2-2XE(X)+(E(X))^2) = E(X^2)-2E(X)E(X)+(E(X))^2$$
$$= E(X^2) - (E(X))^2.$$

and the standard deviation, σ, is defined as $\sigma = \sqrt{\sigma^2}$.

In the last example, the variance is calculated as follows:

$$\sigma^2 = [(20^2)\frac{1}{8}+(20^2)\frac{1}{8}+(20^2)\frac{1}{8}+(20^2)\frac{1}{8}+(20^2)\frac{1}{8}+(21^2)\frac{1}{8}+(19^2)\frac{1}{8}+(19^2)\frac{1}{8}]-[19.875]^2$$
$$= 395.375 - 395.015625 = 0.359375.$$

This number represents "the amount by which X tends to deviate from the average " [1].

The standard deviation is then: $\sigma = \sqrt{.359375} \approx 0.6$.

Figure 1-6 shows the distribution of X.

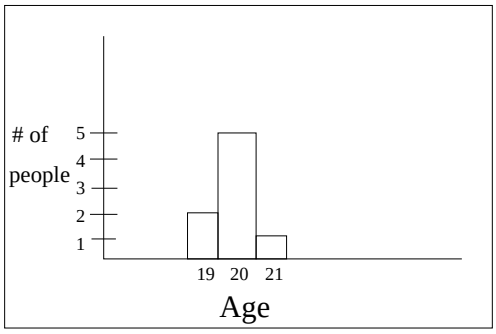

Figure 1-6

X has density function, f, if the probability that X takes on a value between a and b is given by

$$P(a \leq X \leq b) = \int_a^b f(x)\,dx.$$

N(0,1) is defined to be the standard normal distribution. It is a special type of distribution and it has density function given by

$$f(x) = \frac{1}{\sqrt{2\pi}} \exp[-\frac{x^2}{2}].$$

The graph of this function is the familiar bell-shaped curve (see figure 1-7).

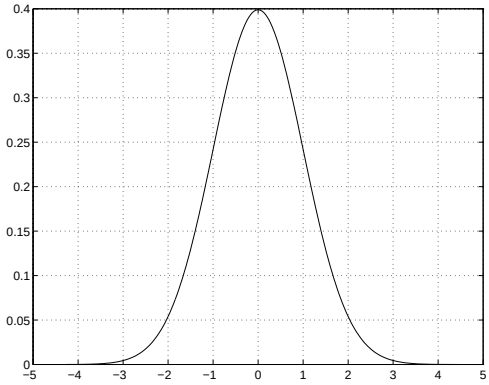

Figure 1-7

We are now ready for the central limit theorem which essentially says that "under appropriate conditions, certain random variables are approximately normally distributed " [8]. This theorem is excerpted directly from [8].

The Central Limit Theorem - Let $X_1, X_2, ...$ be independent random variables having a common distribution with mean μ and finite, positive standard deviation σ. Then

$$\lim_{n\to\infty} Prob(\frac{X_1 + X_2 + ... + X_n - n\mu}{\sigma\sqrt{n}} \leq x) = \frac{1}{\sqrt{2\pi}} \exp[-\frac{x^2}{2}], x \in \Re.$$

Using this theorem, Neiderreiter states that if n is the number of nodes used and s is the dimension we are integrating in, then Monte Carlo Integration has a probabilistic error bound of $O(\frac{1}{\sqrt{n}})$ (note that this error bound

is completely independent of the dimension) and quasi-Monte Carlo Integration "yields a much better result, giving us the deterministic error bound"

$$O(\frac{(\log n)^{(s-1)}}{n})$$

[5].

With large enough n, these methods generally yield suitable accuracy without using up excessive computer time. For this reason, I feel that Monte Carlo Methods and quasi-Monte Carlo methods are a good gamble.

1.5 Programs

This section contains one program which is basically just a concatenation of five smaller programs which apply different numerical integration approximation methods. A possible next step is to extend these programs to higher dimensions and add a 'function call counter' to see the main reason Monte Carlo and quasi-Monte Carlo Methods can be advantageous to use.

MAIN PROGRAM

```
/*ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c  Ian Winokur
c  August 6, 1997                              Last updated:  August 6, 1997
c
c  This program implements five different methods of approximating the
c  value of a definite integral.  The methods are:  the Trapezoidal
c  Rule, Simpson's Rule, two Monte Carlo methods, and one quasi-Monte
c  Carlo method.
c
c  Variable Directory:
c
c       a,b             the endpoints of the interval being
c                           integrated over
c       num_points      the number of points being used in the
```

```
c                              approximation
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

float a,b;
int num_points;

/*  function prototypes */

float mc1(),ran1(),mc2(),qmc1(),simpson(),trap(),f(float x);


void main(void)
{

scanf("%f %f",&a,&b); /*  read in a and b  */
scanf("%d",&num_points);  /*  read in num_points  */

/*    call functions and output results  */

printf("\n\n\n");
printf("Function being integrated:  f(x) = 4x + 2.\n");
printf("Interval:  [%f,%f]\n",a,b);
printf("Number of nodes used:  %d\n\n\n",num_points);
printf(" Method used       Approximation\n");
printf(" -----------       -------------\n\n");


printf("Monte Carlo 1: %f\n",mc1());
printf("Monte Carlo 2: %f\n",mc2());
printf("quasi-Monte Carlo: %f\n",qmc1());
printf("Simpson's Rule:                %f\n",simpson());
printf("Trapezoidal Rule:              %f\n",trap());
```

```
exit(0);
};
```

MONTE CARLO 1 FUNCTION:

```
float mc1()
/*ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c   Ian Winokur
c   July 12, 1997                          Last updated:  August 6, 1997
c
c   This program uses a Monte Carlo Method to approximate the integral of
c  a function, f, on some interval [a,b].  The program picks random
c  coordinates and decides whether they are under the curve or not.  The
c  value of the integral is approximated by the percentage of points under
c  the curve multiplied by the area under some upper bound of the function
c  over the interval.
c
c  Variable directory:
c
c       xi,yi           coordinates of random tosses
c       integral        the value of the integral of f on [a,b]
c       num_points      the number of random tosses
c       area            area between a and b and under bound
c       bound           an upper bound of the function on [a,b]
c       a,b             the start and end points of the interval
c       f               the function being integrated
c       splashes        the number of "splashes"
c       index           an index
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc*/
{
/*  variable declarations */

int  index, splashes;
float xi, yi, integral, area, bound;

/* initialize variables */
```

```
splashes = 0;
scanf ("%f",&bound);   /*  read in upper bound  */


/*  generate random numbers and count splashes  */

for(index=1; index <= num_points; ++index)
 {
  xi = ran1();
  xi = (b - a) * xi + a;    /* scale xi to [a,b]  */
  yi = bound * ran1();
  if (f(xi)>=yi)
 ++splashes;
 };

/*  calculate and output area and integral */

area = bound * (b-a); /* area under bound  */
integral = area * (float)splashes / (float)num_points;
return(integral);
};

RANDOM NUMBER GENERATOR FUNCTION

float ran1()
{
 /* This function generates random numbers in (0,1)  */
 static long int c = 100001;  /* c is the seed  */
 c = (c * 125) % 2796203;
 return (float) c / 2796203;

};  /*  end of definition of function ran1  */
```

```
MONTE CARLO 2 FUNCTION

float mc2()
/*ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c   Ian Winokur
c   July 19, 1997                                Last updated:  August 6, 1997
c
c   This program uses a Monte Carlo Method to approximate the integral of
c  a function, f, on some interval [a,b].  The program picks random
c  x - coordinates, evaluates them, sums up their function values,
c  averages their function values, and then multiplies this average by the
c  length of the interval.  This technique uses the definition of the
c  average value of a function to approximate an integral.
c
c  Variable directory:
c
c       x               x - coordinate of random tosses
c       integral        the value of the integral of f on [a,b]
c       num_points      the number of random tosses
c       a,b             the start and end points of the interval
c       f               the function being integrated
c       index           an index
c       sum             holds the sum of the evaluated xi's
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc*/
{
/*  variable declarations */

int index;
float x, integral, sum;

/* initialize variables */

sum = 0.0;
```

```
/*  generate random numbers and sum up function evaluated there */

for(index=1; index <= num_points; ++index)
 {
  x = ran1();

/*  scale x to [a,b]   */

  x = (b - a) * x + a;
  sum = sum + f(x);
 };

/*  calculate and output  integral */

integral = (float) (b - a) * sum / (float) num_points;
return(integral);
};
```

QUASI-MONTE CARLO FUNCTION

```
float qmc1()
/*ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c Ian Winokur
c Date started:  July 23, 1997              Last updated:  August 6, 1997
c
c This program is an application of a Monte Carlo Method of
c Integration using points from a low-discrepancy, uniformly distributed
c sequence instead of random points.  This program evaluates each point,
c averages the function values, and then multiplies this average by the
c length of the interval to obtain an approximation of the integral. This
c technique uses the definition of the average value of a function.
c
c Variable directory:
c
c         x               the name given to each of the points being used
c         n               counts the number of points being used
c         a,b             the left and right endpoints of the interval
c         sum             contains the sum of the f(x)'s
c         integral        the approximation of the integral of f
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc*/
{
 /*   variable declarations  */
int n;
float  sum = 0.0, x, integral;

 /*  read first number in the sequence  */

scanf("%f",&x);
for (n = 0; x != -99.0; ++n)
{
 /*  scale x values to be in [a,b]  */
 x = a + (b - a) * x;
```

```
 sum = sum + f(x);
 scanf ("%f",&x); /* get next number in the sequence */
}

/*  calculate and return integral  */

integral = (float) (b - a) * sum/ (float) n;
return (integral);
};
```

SIMPSON'S RULE FUNCTION

```
float simpson()
/*ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c  Ian Winokur
c  Date Started:  August 5, 1997                Last Updated:  August 6, 1997
c
c
c  This program approximates the integral of a function, f,  on some interval
c  [a,b] using Simpson's Rule.
c
c  Variable Directory:
c
c       index                   the index used in the main loop
c       simp_sum                the sum used in Simpson's Rule
c       f                       the function being integrated
c       a,b                     the endpoints of the interval that
c                                 f is being integrated on
c       num_points              the number of points being used in
c                                 the approximation
c       x                       the coordinate being evaluated
c       width                   measures the width of each sub-interval
c       simp_integral           holds the approximation using the trapezoidal
c                                 rule
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc*/
{
/*  variable declarations  */

int index;
float simp_sum, x, width, simp_integral;
/*  initialize variables  */

x = a;
simp_sum = f(x);
```

```
width = (b - a)/(float) num_points;

/*  calculate sum  */
for (index = 1; index < num_points; ++index)
{
 x = x + width;   /*  move to next sub-interval  */
 simp_sum = ((index % 2) == 0)? simp_sum + (2.0*f(x)):simp_sum+(4.0*f(x));
};
simp_sum = simp_sum + f(b);
simp_integral = ((width)/(3.0))*simp_sum;
return (simp_integral);
};
```

TRAPEZOIDAL RULE FUNCTION

```
float trap()
/*ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c  Ian Winokur
c  Date Started:  August 5, 1997                Last Updated:  August 6, 1997
c
c
c  This program approximates the integral of a function, f,  on some interval
c  [a,b] using the Trapezoidal Rule.
c
c  Variable Directory:
c
c       index                   the index used in the main loop
c       trap_sum                the sum used in the trapezoidal rule
c       f                       the function being integrated
c       a,b                     the endpoints of the interval that
c                                  f is being integrated on
c       num_points              the number of points being used in
c                                  the approximation
c       x                       the coordinate being evaluated
c       width                   measures the width of each sub-interval
c       trap_integral           holds the approximation using the trapezoidal
c                                  rule
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc*/
{
/*  variable declarations  */

int index;
float trap_sum, x, width, trap_integral;

/*  initialize variables  */

x = a;
```

```
trap_sum = f(x);
width = (b - a)/(float) num_points;

/*  calculate sum */

for (index = 1; index < num_points; ++index)
{
 x = x + width;  /*  move to next sub-interval */
 trap_sum = trap_sum + (2.0 * f(x));
};

/*  add last value to the sum */

trap_sum = trap_sum + f(b);

/*  calculate and return final approximation  */

trap_integral = ((width)/(2.0)) * trap_sum;
return(trap_integral);
};

FUNCTION FUNCTION

float f(float x)
{
 x = 4*x + 2;   /*  put function here */
 return (x);
};  /* end of definition of function f */
```

Chapter 2

Introduction to Low Discrepancy Sequences

The purpose of this report is to study the properties of uniformly distributed low discrepancy sequences. Before this is possible, we need to give some definitions and basic theorems.

2.1 Definitions and Examples

The sequence in Figure 2.1, from [7], is an example of a *uniformly distributed,* or *low discrepancy* sequence in one dimension. The points are evenly spaced throughout the interval, with none of the intervals between points significantly larger or smaller than the others. In order to study the properties of such sequences, we first need to define precisely which sequences have this property.

Definition 2.1 *Let* $\{x_1, x_2, \ldots\}$ *be a sequence of points in* $[0, 1)$ *and* $A \subseteq [0, 1)$*. For a fixed* N*, the function* $\#(A)$ *is defined as the cardinality of the set* $\{x_i \in A : 1 \leq i \leq N\}$*.*

Informally, the function # counts the number of the first N sequence points which are in A. This definition is needed to define the following important concept, found in [3].

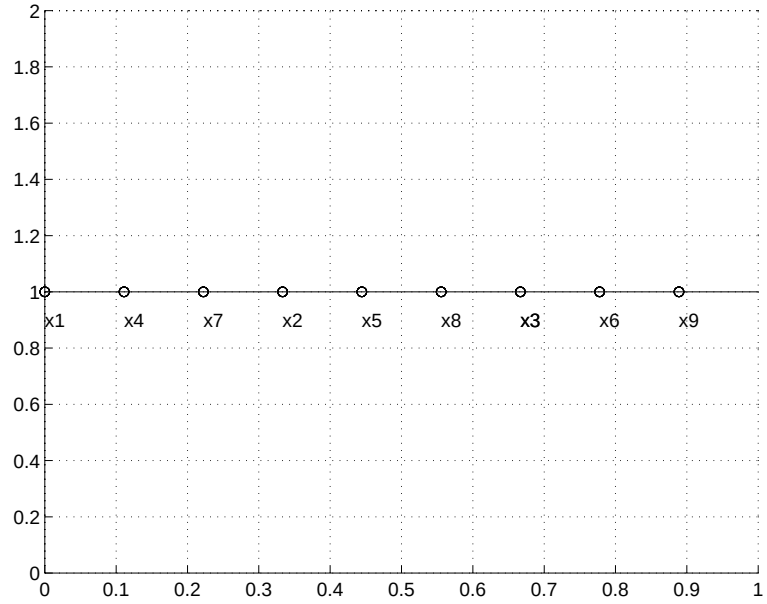

Figure 2.1: x_1 through x_9 of the p-adic sequence, with $p = 3$

Definition 2.2 *Let $\{x_1, x_2, \ldots\}$ be a sequence of points contained in the half-open unit interval $[0, 1)$. Then the sequence is uniformly distributed if for any $[a, b) \subseteq [0, 1)$,*

$$\lim_{N \to \infty} \frac{\#[a, b)}{N} = (b - a).$$

A uniformly distributed sequence is also called a *low discrepancy* sequence. The expression on the left hand side of the equal sign is the fraction of the N sequence points that lie within the interval, whereas the expression on the right hand side is the fraction of the length of $[0, 1)$ that lies in the interval. If a sequence is low-discrepancy, these quantities become close as N becomes large.

This leads to a question of how to measure quantitatively the "uniform-ness" or "non-uniform-ness" of a sequence. This quantity, called *discrepancy*, is important in determining the accuracy of quasi-Monte Carlo integration methods. The following definition is in [3].

Definition 2.3 *Let $\omega = \{x_1, x_2, \ldots, x_N\}$ be a sequence of points in $[0, 1)$.*

The discrepancy of the sequence, denoted $D_N(\omega)$, is equal to

$$D_N(\omega) = \sup_{0 \le \alpha < \beta \le 1} \left| \frac{\#[\alpha, \beta)}{N} - (\beta - \alpha) \right|.$$

Clearly, a sequence ω is uniformly distributed if $\lim_{N\to\infty} D_N(\omega) = 0$. Other useful definitions related to this one are given in [3], such as *star discrepancy* and *isotropic discrepancy*.

A similar definition holds in higher dimensions. The function # is defined in $\mathbf{R}^s$ the same way as in $\mathbf{R}$, except that the sequence is in $[0,1)^s = [0,1) \times [0,1) \times \cdots \times [0,1)$.

Definition 2.4 *Let $\{\mathbf{x}_1.\mathbf{x}_2.\ldots\}$ be a sequence of points in $[0,1)^s$. Then the sequence is uniformly distributed if for any $J = [a_1, b_1) \times \cdots \times [a_s, b_s) \subseteq [0,1)^s$,*

$$\lim_{N\to\infty} \frac{\#(J)}{N} = \lambda(J),$$

where $\lambda(\cdot)$ denotes s-dimensional Lebesgue measure.

Definition 2.5 *Let $\omega = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ be a sequence of points in $[0,1)^s$. The discrepancy of the sequence, denoted $D_N(\omega)$, is equal to*

$$D_N(\omega) = \sup_{J \in \mathcal{J}} \left| \frac{\#(J)}{N} - \lambda(J) \right|,$$

where $\mathcal{J} = \{[a_1, b_1) \times \cdots \times [a_s, b_s) : 0 \le a_i < b_i \le 1 \text{ for } 1 \le i \le s\}$ and $\lambda(\cdot)$ denotes s-dimensional Lebesgue measure.

2.2 Basic Theorems on Discrepancy

The following property follows from the definition of discrepancy. Kuipers and Niederreiter [3] prove it.

Theorem 2.1 *For any sequence ω, $\frac{1}{N} \le D_N(\omega) \le 1$.*

Niederreiter [5] also gives a theorem which gives an expression for the discrepancy as a maximum of a finite set, rather than a supremum over an infinite set.

Theorem 2.2 *For a sequence* $\omega = \{x_1, x_2, \ldots, x_N\}$ *in* $[0,1)$,

$$D_N(\omega) = \frac{1}{N} + \max_{1 \le i \le N} \left(x_i - \frac{i}{N} \right) - \min_{1 \le i \le N} \left(x_i - \frac{1}{N} \right).$$

Another theorem, sometimes called the Triangle Inequality for Discrepancies, relates the discrepancy of a sequence to the discrepancies of each element of a partition of the sequence. Our proof closely follows the proof in [3] for the corresponding theorem in $\mathbf{R}$.

Theorem 2.3 *For* $1 \le i \le k$, *let* ω_i *be a sequence of* N_i *elements in* $[0,1)^s$ *with discrepancy* $D_{N_i}(\omega_i)$. *Let* ω *be a sequence containing exactly the elements of* $\omega_1, \omega_2, \ldots, \omega_k$ *in some order. Let* $N = N_1 + N_2 + \ldots + N_k$ *be the number of elements in* ω. *Then*

$$D_N(\omega) \le \sum_{i=1}^{k} \frac{N_i}{N} D_{N_i}(\omega_i).$$

Proof:
Let $J = [a_1, b_1) \times \cdots \times [a_s, b_s)$ be a subinterval of $[0,1)^s$. Let $\#_i$ represent the function $\#$ for the subsequence ω_i. Then $\#(J) = \sum_{i=1}^{k} \#_i(J)$, by the definition of ω. Now note that

$$\left| \frac{\#(J)}{N} - \lambda(J) \right| = \left| \sum_{i=1}^{k} \frac{\#_i(J)}{N} - \lambda(J) \right|$$

$$= \left| \sum_{i=1}^{k} \left(\frac{N_i}{N} \cdot \frac{\#_i(J)}{N_i} \right) - \sum_{i=1}^{k} \frac{N_i}{N} \lambda(J) \right|$$

$$= \left| \sum_{i=1}^{k} \frac{N_i}{N} \left(\frac{\#_i(J)}{N_i} - \lambda(J) \right) \right|$$

$$\le \sum_{i=1}^{k} \frac{N_i}{N} D_{N_i}(\omega_i),$$

by the definition of discrepancy and the Triangle Inequality. Putting all these inequalities together gives

$$\left| \frac{\#(J)}{N} - \lambda(J) \right| \le \sum_{i=1}^{k} \frac{N_i}{N} D_{N_i}(\omega).$$

This holds for any J, so the supremum over all possible J of the left side of the inequality is still less than or equal to the right side. This supremum is exactly the definition of $D_N(\omega)$, giving the desired conclusion. ■

2.3 The p-adic Sequence

One common low-discrepancy sequence is the p-adic sequence, shown in Figure 2.1. Its terms are formed by reflecting the base-p representations of consecutive integers around the "decimal point." In other words, the coefficient c_i of p^i is "reflected" to become the coefficient of p^{-i-1}.

Mathematically, let $n-1 = \sum_{i=0}^{M} c_i p^i$ be the base-p representation of $n-1$. (In order to make $x_1 = 0$, $n-1$ is used instead of n.) Now, "reflect" these digits to give $x_n = \sum_{i=0}^{M} c_i p^{-i-1}$, the nth term of the p-adic sequence. Table 2.1 shows this process. An asymptotic bound for D_N for this sequence is proved in Chapter 3.

n	$n-1$	$n-1$ in base 3	x_n in base 3	x_n
1	0	0	0.0	0
2	1	1	0.1	1/3
3	2	2	0.2	2/3
4	3	10	0.01	1/9
5	4	11	0.11	4/9
6	5	12	0.21	7/9
7	6	20	0.02	2/9
8	7	21	0.12	5/9
9	8	22	0.22	8/9

Table 2.1: Values of the 3-adic sequence through $n = 9$

The following chapters give more examples of low-discrepancy sequences. Pace and Salazar-Lazaro [7] give a further exposition of the topic.

Chapter 3

Some One-Dimensional Sequences And Their Discrepancies

A number of sequences have been studied extensively. Asymptotic bounds have been proven for their discrepancies D_N as N goes to infinity. This chapter describes the properties of two such sequences.

3.1 The p-adic sequence

The p-adic sequence described in chapter 2 is one such sequence whose properties are well known. The following theorem gives an asymptotic discrepancy bound of $O(\frac{\log N}{N})$ for the p-adic sequence; the proof closely follows a theorem in [3] proving a bound for the case $p = 2$.

Theorem 3.1 *Let ω be the p-adic sequence with base p, formed as described above. The sequence satisfies*

$$ND_N(\omega) \leq (p-1)\left(\frac{\log(N+1)}{\log p} + 1\right).$$

Proof:
Let $\sum_{i=1}^{M} b_i p^i$ with $0 \leq b_i \leq p-1$ and $b_M \neq 0$ be the base-p representation of N. We now write N as a sum of s pure powers of p, where p^i appears as a term

b_i times in the sum; $N = p^{h_1} + p^{h_2} + \cdots + p^{h_s}$, with $h_1 \geq h_2 \geq \cdots \geq h_s \geq 0$. Note that $s = \sum_{i=0}^{M} b_i$.

Partition the set of integers $\{1, 2, \ldots, N\}$ into s subsets in the following way: For $1 \leq j \leq s$, let

$$M_j = \{x \in \mathbf{Z} : (p^{h_1} + p^{h_2} + \cdots + p^{h_{j-1}} + 1) \leq x \leq (p^{h_1} + p^{h_2} + \cdots + p^{h_j})\},$$

where here and throughout this proof, an empty sum equals 0. This means that the smallest element of M_1 is 1. Note that each M_j contains p^{h_j} elements.

An integer n in M_j can be written as

$$n = 1 + p^{h_1} + \cdots + p^{h_{j-1}} + \sum_{i=0}^{h_j - 1} a_i p^i,\ a_i \in \{0, 1, \ldots, p-1\}$$

Note that the values of n determined by all the possible values of the set of a_i are precisely the integers in M_j.

By definition of x_n, using the integer n given above,

$$x_n = p^{-h_1 - 1} + \cdots + p^{-h_{j-1} - 1} + \sum_{i=0}^{h_j - 1} a_i p^{-i-1}.$$

Define $y_j = p^{-h_1 - 1} + \cdots + p^{-h_{j-1} - 1}$. Then $x_n = y_j + \sum_{i=0}^{h_j - 1} a_i p^{-i-1}$, where y_j depends on p and j but not n. Since y_j is the base-p representation of a number for which the largest power is $p^{-h_{j-1}-1}$, it follows that $0 \leq y_j \leq p^{-h_j}$.

If n runs through all the integers in M_j, then the sum $\sum_{i=0}^{h_j - 1} a_i p^{-i-1}$, where the a_i are determined by n, runs through the fractions $0, p^{-h_j}, 2p^{-h_j}, \ldots, (p^{h_j} - 1)p^{-h_j}$. These p^{h_j} distinct values are all in the interval $[0, 1)$. Therefore, the sequence $\{y_j, y_j + p^{-h_j}, \ldots, y_j + (p^{h_j} - 1)p^{-h_j}\}$ is evenly spaced in $[0, 1)$ with difference p^{-h_j}.

Now partition the sequence ω into sub-sequences $\omega_j = \{z_{jn}\} = \{x_n : n \in M_j\}$. The number of elements of M_j is p^{h_j}, so each subsequence ω_j contains p^{h_j} elements. By Theorem 2.2,

$$D_{p^{h_j}}(\omega_j) = \frac{1}{p^{h_j}} + \max_{1 \leq i \leq p^{h_j}} \left(z_{ji} - \frac{i}{p^{h_j}} \right) - \min_{1 \leq i \leq p^{h_j}} \left(z_{ji} - \frac{i}{p^{h_j}} \right).$$

For each i, $1 \leq i \leq p^{h_j}$, z_{ji} equals $y_j + (i-1)p^{-h_j}$. Thus $z_{ji} - i/p^{h_j} = y_j + (i-1)p^{-h_j} - ip^{-h_j} = y_j - p^{-h_j}$ for each i. This value is constant over

i, so its maximum and minimum are equal. Hence $D_{p^{h_j}}(\omega_j) = 1/p^{h_j}$ and $p^{h_j} D_{p^{h_j}}(\omega_j) = 1$.

By Theorem 2.3,

$$ND_N(\omega) \leq \sum_{j=1}^{s} p^{h_j} D_{p_{h_j}}(\omega_j) = s.$$

We now bound s in terms of N. Recall that $N = p^{h_1} + p^{h_2} + \cdots + p^{h_s}$, a sum of s pure powers of p. N is thus greater than or equal to the smallest integer which can be expressed as the sum of s pure powers of p, with no single power appearing more than $p-1$ times. This number is

$$(p-1)p^0 + \cdots + (p-1)p^{\lfloor \frac{s}{p-1} \rfloor - 1} + tp^{\lfloor \frac{s}{p-1} \rfloor}$$

with $0 \leq t < p-1$. This number in turn is greater than

$$(p-1)p^0 + \cdots + (p-1)p^{\lfloor \frac{s}{p-1} \rfloor - 1}.$$

The sum of $(p-1)p^0 + \cdots + (p-1)p^k = p^{k+1} - 1$ for any positive integer k. Thus,

$$\begin{aligned} N &\geq (p-1)p^0 + \cdots + (p-1)p^{\lfloor \frac{s}{p-1} \rfloor - 1} \\ &= p^{\lfloor \frac{s}{p-1} \rfloor} - 1. \end{aligned}$$

Therefore,

$$N + 1 \geq p^{\lfloor \frac{s}{p-1} \rfloor},$$

and

$$\frac{\log(N+1)}{\log p} \geq \left\lfloor \frac{s}{p-1} \right\rfloor \geq \frac{s}{p-1} - 1,$$

by definition of the floor function. It then follows that

$$\frac{s}{p-1} \leq \frac{\log(N+1)}{\log p} + 1,$$

or

$$s \leq (p-1)\left(\frac{\log(N+1)}{\log p} + 1\right). \blacksquare$$

3.2 The rotation sequence

Another sequence with a known discrepancy bound is the *rotation sequence*. Let $\{x\}$ be the fractional part of x, $x - \lfloor x \rfloor$. For some irrational number α, the rotation sequence is given by $x_n = \{(n-1)\alpha\}$. Sometimes the sequence is indexed differently, with $x_n = \{n\alpha\}$. (If α is a rational fraction $\frac{p}{q}$, the rational parts will begin to repeat after q terms of the sequence, whereas an irrational α will give an infinite sequence, since according to [3] the terms of the sequence are dense in $[0, 1)$.) Figure 3.1 shows this process for α approximately equal to 0.47, with the interval $[0, 1)$ represented as both a circle and a line segment. Using the circular model, it is easier to visualize the operation of the fractional part function.

Pace and Salazar-Lazaro [7] give an algorithm to give the points in the rotation sequence of α without referring to fractional parts of multiples of α. Our algorithm is based on theirs. Let $\gamma_1 = \alpha$. Let $x_1 = 0$. In the first stage, the following points are each added at a distance of γ_1 to the right of the preceding point. When there is no room left to add another point in $[0, 1)$, call the leftover distance γ_2.

The second stage begins now. For each point of the sequence marked, in the order of their indices, mark a new point at a distance γ_2 to the left of the existing point, if no closer point exists in that direction. Continue this process for the points marked in stage 2 as well, until there is no room to mark new points without moving past an existing point. In each interval of length γ_1, a number of intervals of length γ_2 have been marked off; call the leftover distance γ_3.

This pattern continues, marking points in the positive direction from existing points in odd stages, and in the negative direction in even stages, with γ_{i+1} defined as the leftover distance after partitioning each sub-interval in the ith stage. The term *stage* i as used in this paper refers to the part of the algorithm when points are marked off at a distance of γ_i. This process gives a recurrence relation for γ_i:

$$\gamma_{i+1} = (-1)^{i+1} \left(|\gamma_{i-1}| - |\gamma_i| \left\lfloor \left| \frac{\gamma_{i-1}}{\gamma_i} \right| \right\rfloor \right),$$

where the sequence is alternating because the direction in which new points are added alternates. Notice that the sequence $\{|\gamma_i|\}$ is strictly decreasing.

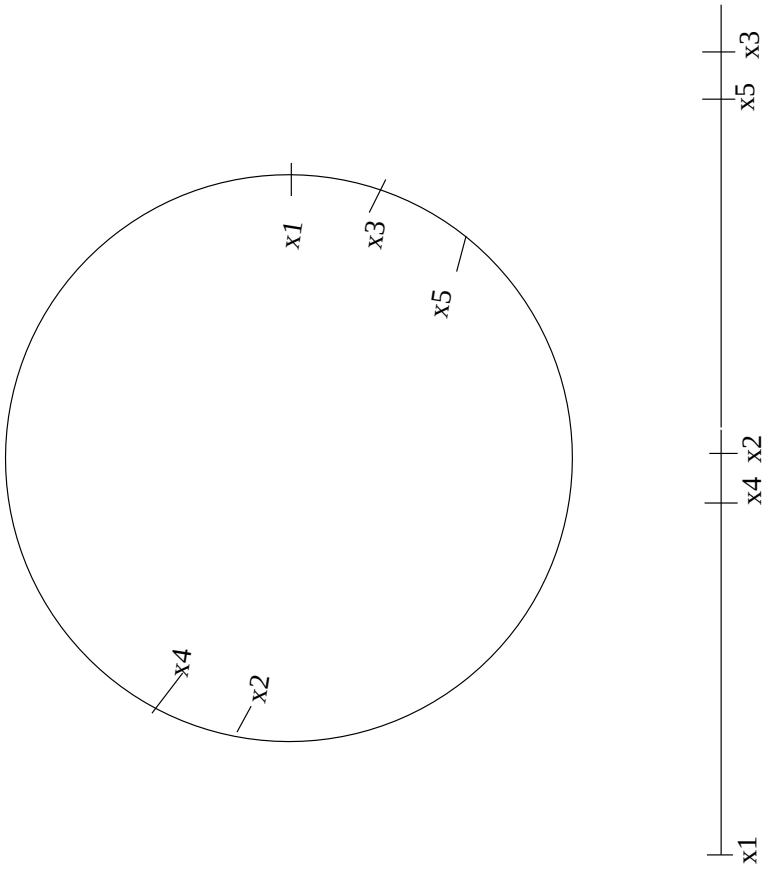

Figure 3.1: Rotation sequence for $\alpha \approx 0.47$

To state this precisely, let $r(i)$ and $b(i)$ represent the index of the point closest to x_i on the right and the distance $x_{r(i)} - x_i$, respectively. Similarly, let $l(i)$ and $c(i)$ represent the index of the point closest to x_i on the left and the distance $x_i - x_{l(i)}$, respectively. For a point near the ends of $[0, 1)$, there may be no point satisfying these conditions; in this case let $r(i)$ or $l(i)$ equal -1, as a signal value. In order to avoid special cases related to this, the values $b(-1)$ and $c(-1)$, which may be written during the running of the algorithm but are never read, are dummy values with no relevance.

N equals the number of points marked in the sequence. Each point is marked at a certain distance γ in a certain direction from an existing point, if there are no closer points in that direction. q denotes the existing point for which the distance to the nearest point in the given direction is currently being compared to γ.

- Let $x_1 = 0$, $b(1) = 1$, $c(1) = 0$, $l(1) = -1$, $r(1) = -1$, $N = 1$, $q = 1$, $stage = 1$, $done = FALSE$.
- LOOP:
  - Let $\gamma = \gamma_{stage}$.
  - If $\gamma > 0$, LOOP:
    - If $b(q) > \gamma$
      - Mark a new point: $x_{N+1} = x_q + \gamma$
      - $b(N + 1) = b(q) - \gamma$, $c(N + 1) = \gamma$
      - $b(q) = \gamma$, $c(r(q)) = b(N + 1)$
      - $r(N + 1) = r(q)$, $l(N + 1) = q$
      - $r(q) = N + 1$, $l(r(N + 1)) = N + 1$
      - $q = q + 1$, $N = N + 1$

      Else
      - If $q \neq N$, $q = q + 1$
      - Else, let $done = TRUE$
  - UNTIL $done = TRUE$
  - Else if $\gamma < 0$, LOOP:
    - If $c(q) > |\gamma|$

· Mark a new point: $x_{N+1} = x_q + \gamma$. Notice that γ is negative here;
· $b(N+1) = |\gamma|$, $c(N+1)$ $c(q) + \gamma$
· $c(q) = |\gamma|$, $b(l(q)) = c(N+1)$
· $l(N+1) = l(q)$, $r(N+1) = q$
· $l(q) = N+1$, $r(l(N+1)) = N+1$
· $q = q+1$, $N = N+1$

Else

· If $q \neq N$, $q = q+1$
· Else, let $done = TRUE$

– UNTIL $done = TRUE$
– Let $stage = stage + 1$, $q = 1$, $done = FALSE$, $N_i = N$

• End outer loop

Figure 3.2 shows the first three stages, along with the distances between points and the indices of the points, for the rotation sequence with $\alpha = \frac{\sqrt{5}-1}{2}$.

3.2.1 Continued Fractions

The rotation sequence is closely related to the theory of continued fractions. Continued fractions give an alternative to the standard decimal or base-p representation of real numbers. Any number can be expressed in the form

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cdots}}},$$

where each a_i is an integer and $a_i \geq 1$ for $i \geq 1$. Such an expression is called a *simple continued fraction*. The a_i are called *partial quotients*. The continued fraction expression is often denoted $[a_0, a_1, a_2, \ldots]$. A *finite continued fraction* is an expression of the form

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{\cdots + \cfrac{1}{a_k}}},$$

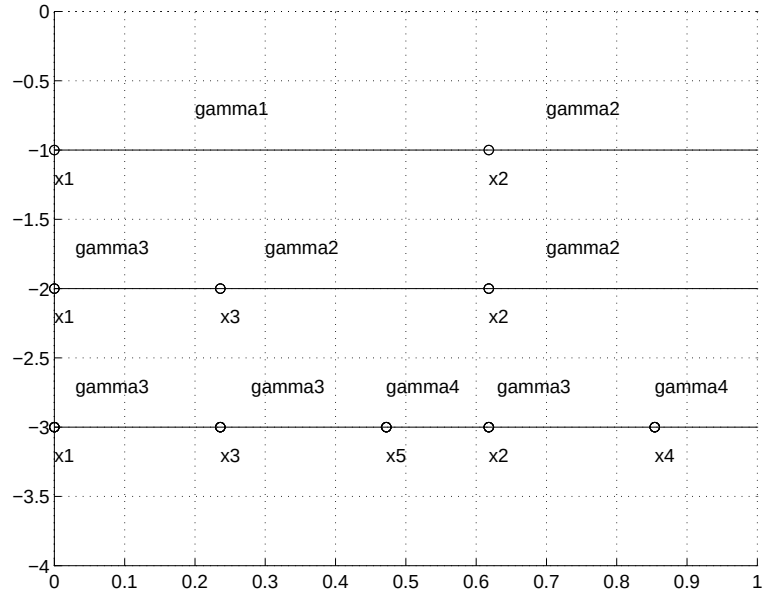

Figure 3.2: Stages 1, 2, 3 of rotation sequence for $\alpha = (\sqrt{5}-1)/2$

where the set of partial quotients is finite. It is denoted $[a_0, a_1, \ldots, a_k]$. The value of an infinite continued fraction is a converging limit of finite continued fractions.

Khinchin [2] shows the following result:

Lemma 3.1 *The number represented by $\alpha = [a_0, a_1, a_2, \ldots]$ is rational if and only if the expansion is finite.*

If $\alpha = [a_0, a_1, a_2, \ldots]$ is an infinite continued fraction, the number $r_k = [a_0, a_1, \ldots, a_k]$ is called the kth *convergent* of α, where $\lim_{k\to\infty} r_k = \alpha$. Since the convergents are finite continued fractions, each is a rational number and can be expressed as $r_k = \frac{p_k}{q_k}$ for some integers p_k, q_k. Khinchin [2] recalls that the numerators and denominators of the convergents satisfy a recurrence relation.

Lemma 3.2 *For all integers $k \geq 1$,*

$$p_k = a_k p_{k-1} + p_{k-2}$$

and

$$q_k = a_k q_{k-1} + q_{k-2}$$

with the initial conditions $p_{-1} = 1, p_0 = a_0, q_{-1} = 0, q_0 = 1$.

Clearly, the -1st convergent does not exist; the given values for p_{-1} and q_{-1} are bookkeeping devices only.

The *continued fraction transformation* T is given by $T(x) = \{\frac{1}{x}\}$, where $\{z\}$ denotes the fractional part of z, $z - \lfloor z \rfloor$. Let $\alpha \in [0, 1)$; clearly from the form of a continued fraction expression, $a_0 = 0$. If $\alpha = [0, a_1, a_2, \ldots]$ in continued fraction form, then

$$T(\alpha) = \left\{\frac{1}{\alpha}\right\} = \left\{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \cdots}}\right\}$$

$$= a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \cdots}} - \left\lfloor a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \cdots}} \right\rfloor$$

$$= a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \cdots}} - a_1$$

$$= \frac{1}{a_2 + \frac{1}{a_3 + \cdots}} = [0, a_2, a_3, \ldots].$$

Thus the transformation T shifts the partial quotients of α to the left.

Now consider the rotation sequence for the number α; assume without loss of generality that $\alpha \in [0, 1)$. From the rotation algorithm, we know that $|\gamma_{i+1}| = |\gamma_{i-1}| - \left\lfloor \left| \frac{\gamma_{i-1}}{\gamma_i} \right| \right\rfloor |\gamma_i|$. (Absolute values are used in this discussion because the direction is not important.) Recall that $\gamma_1 = \alpha = [0, a_1, a_2, \ldots]$.

Consider the number of intervals of width γ_1 that fit into an interval of width γ_0, $\left\lfloor \left| \frac{\gamma_0}{\gamma_1} \right| \right\rfloor$. Note that

$$\left\lfloor \left| \frac{\gamma_0}{\gamma_1} \right| \right\rfloor = \left\lfloor \frac{1}{[0, a_1, a_2, \ldots]} \right\rfloor = \left\lfloor a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \cdots}} \right\rfloor = a_1.$$

Thus a_1 is the number of intervals of width γ_1 fit into an interval or width γ_0. The leftover piece after a_1 intervals of length γ_1 are removed has length $|\gamma_2| = |\gamma_0| - \left\lfloor \left| \frac{\gamma_0}{\gamma_1} \right| \right\rfloor |\gamma_1| = |\gamma_0| - a_1|\gamma_1|$. But

$$|\gamma_0| - a_1|\gamma_1| = 1 - \frac{a_1}{a_1 + \frac{1}{a_2 + \cdots}}$$

$$= \frac{1}{a_1 + \frac{1}{a_2 + \cdots}} \left(\left(a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \cdots}} \right) - a_1 \right)$$

$$= \alpha \left(\frac{1}{a_2 + \frac{1}{a_3 + \cdots}} \right)$$

$$= \alpha [0, a_2, a_3, \ldots]$$

$$= \alpha T(\alpha).$$

A simple induction argument will show that the same relationship holds for higher indices.

Theorem 3.2 *Let $\alpha \in [0, 1)$ have the continued fraction representation $[0, a_1, a_2, \ldots]$. Let $T(\alpha)$ be the continued fraction transformation $T(\alpha) = \left\{ \frac{1}{\alpha} \right\}$, with T^i denoting i iterations of the transformation. Let γ_i be defined for the rotation sequence with $\gamma_0 = 1$ and $\gamma_1 = \alpha$. Then $|\gamma_i| = \alpha \cdot T(\alpha) \cdots T^{i-1}(\alpha) = [0, a_i, a_{i+1}, \ldots]$ and $a_i = \left\lfloor \left| \frac{\gamma_{i-1}}{\gamma_i} \right| \right\rfloor$.*

Proof:
The above paragraph gives the base case, $\gamma_1 = \alpha$ and $a_1 = \left\lfloor \left| \frac{\gamma_0}{\gamma_1} \right| \right\rfloor$. Also $\gamma_0 = 1 = T^0(\alpha)$. Now suppose that $\gamma_{i-1} = \alpha \cdot T(\alpha) \cdots T^{i-2}(\alpha)$, $\gamma_i = \alpha \cdot T(\alpha) \cdots T^{i-1}(\alpha)$, and $a_i = \left\lfloor \left| \frac{\gamma_{i-1}}{\gamma_i} \right| \right\rfloor$. From the recursive definition for γ_{i+1},

$$\begin{aligned}
|\gamma_{i+1}| &= |\gamma_{i-1}| - \left\lfloor \left| \frac{\gamma_{i-1}}{\gamma_i} \right| \right\rfloor |\gamma_i| \\
&= |\gamma_{i-1}| - a_i|\gamma_i| \\
&= \alpha \cdot T(\alpha) \cdots T^{i-2}(\alpha) \; - \; a_i(\alpha \cdot T(\alpha) \cdots T^{i-1}(\alpha)) \\
&= \alpha \cdot T(\alpha) \cdots T^{i-1}(\alpha) \left(\frac{1}{T^{i-1}(\alpha)} - a_i \right) \\
&= \alpha \cdot T(\alpha) \cdots T^{i-1}(\alpha) \left(\frac{1}{[0, a_i, a_{i+1}, \ldots]} - a_i \right) \\
&= \alpha \cdot T(\alpha) \cdots T^{i-1}(\alpha) \left(a_i + \frac{1}{a_{i+1} + \frac{1}{a_{i+2} + \cdots}} - a_i \right) \\
&= \alpha \cdot T(\alpha) \cdots T^{i-1}(\alpha) \left([0, a_{i+1}, a_{i+2}, \ldots] \right) \\
&= \alpha \cdot T(\alpha) \cdots T^{i-1}(\alpha) \cdot T^i(\alpha).
\end{aligned}$$

Additionally,

$$\left\lfloor \left| \frac{\gamma_i}{\gamma_{i+1}} \right| \right\rfloor = \left\lfloor \frac{\alpha \cdot T(\alpha) \cdots T^{i-1}(\alpha)}{\alpha \cdot T(\alpha) \cdots T^i(\alpha)} \right\rfloor.$$

By canceling terms in the numerator and denominator, this equals

$$\left\lfloor \frac{1}{T^i(\alpha)} \right\rfloor = \left\lfloor a_{i+1} + \frac{1}{a_{i+2} + \frac{1}{a_{i+3} + \cdots}} \right\rfloor = a_{i+1}. \blacksquare$$

Certain other properties relating continued fractions to the rotation algorithm can be proven from the algorithm. First, however, we state a definition and prove an important lemma.

Definition 3.1 *The ith rotation sequence partition of* $[0, 1)$ *is the partition of* $[0, 1)$ *with partition points* $\{x_n : n \in \mathbf{Z}, 1 \leq n \leq N_i\} \cup \{1\}$.

Lemma 3.3 *In the ith rotation sequence partition of* $[0, 1)$*, each partition interval has length equal to either* $|\gamma_i|$ *or* $|\gamma_{i+1}|$*.*

Proof:
The proof will be by induction on the number of the stage. In stage 1, points are marked off beginning with $x_1 = 0$ and moving a distance of γ_1 to the right. By Theorem 3.2, a_1 intervals of length γ_1 are marked off, and the remaining interval by definition has length $|\gamma_2|$. This establishes the base case.

Now assume that each interval of the ith rotation sequence partition of $[0, 1)$ has length equal to either $|\gamma_i|$ or $|\gamma_{i+1}|$. During stage $i + 1$, the placement of points depends on the value of $b(q)$ or $c(q)$, depending on whether $i + 1$ is odd or even. By the induction hypothesis, this value is either $|\gamma_i|$ or $|\gamma_{i+1}|$ (or, in the case of $c(1)$, zero). The value $b(q)$ or $c(q)$ is compared to $|\gamma_{i+1}|$ in stage $i + 1$. If it is less than $|\gamma_{i+1}|$, no point is marked. If the value is greater, which is the case for an interval of width $|\gamma_i|$, then a point is marked. Each of these intervals of width $|\gamma_i|$ is partitioned into intervals of width $|\gamma_{i+1}|$ and a leftover piece at the end of the stage. Since each of these intervals has the same length at the start of the stage, each leftover piece will have the same length, and by definition this length is $|\gamma_{i+2}|$. Therefore, the new partition formed by the points marked after stage $i + 1$ has intervals of length equal to either $|\gamma_{i+1}|$ or $|\gamma_{i+2}|$, establishing the induction case. ■

One result following from this lemma determines the number of intervals in the ith rotation sequence partition, from which follows an expression for N_i, the number of points given after stage i of the algorithm. The result is well-known, but the proof is ours.

Theorem 3.3 *The set of partition intervals of the ith rotation sequence partition of* $[0, 1)$ *contains* q_i *intervals of length* $|\gamma_i|$ *and* q_{i-1} *intervals of length* $|\gamma_{i+1}|$*.*

Proof:
Let d_i be the number of partition intervals in the ith rotation sequence partition of $[0, 1)$. It will be shown that d_i satisfies the same recurrence relation and initial conditions as q_i.

At the beginning of the algorithm, $x_1 = 1$ is the only point. Thus the 0th rotation sequence partition is the trivial partition, and its length is $1 = |\gamma_0|$. So $d_0 = 1$. By Theorem 3.2, $q_0 = 1$ as well.

The first rotation sequence partition of $[0, 1)$ produces a_1 intervals of width $|\gamma_1|$, by Theorem 3.2. So $d_1 = a_1$. The recurrence relation in Theorem 3.2 gives $q_1 = a_1 q_0 + q_{-1} = a_1$. These initial conditions for d_i are consistent with the corresponding values of q_i.

Now consider an arbitrary stage i, $i \geq 2$. Just before stage i begins, stage $i-1$ is completed, and the $(i-1)$st rotation sequence partition of $[0, 1)$ gives d_{i-1} intervals of length $|\gamma_{i-1}|$. During stage i, each of these intervals is further partitioned into a_i intervals of length $|\gamma_i|$, plus a leftover piece, giving a total of $a_i d_{i-1}$ intervals.

However, other intervals of length $|\gamma_i|$ exist. The $(i-2)$nd rotation sequence partition gives intervals of length $|\gamma_{i-2}|$ and $|\gamma_{i-1}|$ (Lemma 3.3). During stage $i-1$, each interval of length $|\gamma_{i-2}|$ are further partitioned into intervals of length $|\gamma_{i-1}|$ and one leftover piece of length $|\gamma_i|$. These intervals of length $|\gamma_i|$ remain intact during stage i. The number of these intervals is the same as the number of intervals of width $|\gamma_{i-2}|$ at the end of stage $i-2$, which equals d_{i-2} by definition.

The total number of intervals of length $|\gamma_i|$ after stage i equals

$$d_i = a_i d_{i-1} + d_{i-2},$$

which is precisely the recurrence relation for q_i. Since $d_0 = q_0$ and $d_1 = q_1$, it follows that $d_i = q_i$ for $i \geq 0$, proving the first claim of the theorem.

To prove the second claim, note that we have just shown that the $(i-1)$st rotation sequence partition of $[0, 1)$ gives q_{i-1} intervals of length $|\gamma_{i-1}|$ and some number of intervals of length $|\gamma_i|$. During stage i, each interval of length $|\gamma_{i-1}|$ is further partitioned into intervals of length $|\gamma_i|$ and one leftover piece of length $|\gamma_{i+1}|$, while the intervals of length $|\gamma_i|$ are not further partitioned. Therefore the number of intervals of length $|\gamma_{i+1}|$ is the same as the number of intervals of length $|\gamma_{i-1}|$ in the $(i-1)$st partition, which is q_{i-1}. This establishes the second claim. ■

Corollary 3.1 $N_i = q_i + q_{i-1}$.

Proof:
The number of partition points in the ith rotation sequence partition is clearly

$N_i + 1$, since the set of partition points is exactly the N_i points given by the algorithm and one extra point. The number of partition intervals is one less than the number of partition points, so there are N_i partition intervals. But, combining Lemma 3.3 and Theorem 3.3, the number of intervals is $q_i + q_{i-1}$. Thus $N_i = q_i + q_{i-1}$. ■

Theorem 3.4 *For any i, with γ_i and N_i given by the rotation algorithm, $|\gamma_{i+1}| < \frac{1}{N_i} < |\gamma_i|$.*

Proof:
Proof is by contradiction. Suppose that $|\gamma_{i+1}| \geq \frac{1}{N_i}$. The ith rotation sequence partition gives q_i intervals of length $|\gamma_i|$ and q_{i-1} intervals of length $|\gamma_{i+1}|$ (Theorem 3.3). Since the sum of the lengths of the partition intervals equals 1, the length of the entire interval $[0, 1)$, it follows that $1 = q_{i-1}|\gamma_{i+1}| + q_i|\gamma_i|$. Since $|\gamma_i| > |\gamma_{i+1}|$, $1 > (q_{i-1} + q_i)|\gamma_{i+1}| = N_i|\gamma_{i+1}|$. But the hypothesis implies that $N_i|\gamma_{i+1}| \geq 1$, a contradiction. Thus $|\gamma_{i+1}| < \frac{1}{N_i}$.

Similarly, suppose that $|\gamma_i| \leq \frac{1}{N}$. Then $1 = q_{i-1}|\gamma_{i+1}| + q_i|\gamma_i|$, just as above. Since $|\gamma_{i+1}| = |\gamma_i|$, $1 < (q_{i-1} + q_i)|\gamma_i| = N_i|\gamma_i|$. But the hypothesis implies that $N_i|\gamma_i| \leq 1$. This is a contradiction, so $|\gamma_i| > \frac{1}{N}$. The conclusion follows from these two results. ■

The points x_{N_i}, the values of the last points added in each stage, seem to approach some limit as N goes to infinity. In particular, the following property holds. Define $N_{-1} = 1$, $N_0 = 0$, $\gamma_0 = 1$ for bookkeeping devices.

Theorem 3.5 *For all N_i, $i \geq 1$,*

$$x_{N_{i-1}} < x_{N_i} < x_{N_{i-2}}, \quad \textit{for } i \textit{ odd,}$$

$$x_{N_{i-1}} > x_{N_i} > x_{N_{i-2}}, \quad \textit{for } i \textit{ even.}$$

Proof:
The base case is trivial. Since x_{N_1} is a point in $[0, 1)$, and clearly from the algorithm $x_{N_1} \neq 0$, it follows that $0 = x_{N_0} < x_{N_1} < x_{N_{-1}} = 1$.

Now assume that the claim holds at the end of stage $i-1$ for some positive integer i. If i is odd, $i - 1$ is even and $x_{N_{i-3}} < x_{N_{i-1}} < x_{N_{i-2}}$. From the recursion relation defining γ_i, $\gamma_i > 0$.

As stage i begins (or equivalently, stage $i-1$ ends), each of the intervals of the $(i-1)$st rotation sequence partition has length $|\gamma_{i-1}|$ or $|\gamma_i|$ (Theorem 3.3). Thus for all q where such a value exists, $b(q), c(q) \in \{|\gamma_{i-1}|, |\gamma_i|\}$.

When the point $x_{N_{i-1}}$ is added, $b(N_{i-1})$ is set to $|\gamma_{i-1}|$. The value of $b(q)$ for any q changes only when a new point is marked; since $x_{N_{i-1}}$ is the last point added in stage $i-1$, $b(N_{i-1}) = |\gamma_{i-1}|$ still at the start of stage i.

In stage i, q begins at 1. When $b(q) = \gamma_i$, no point is added according to the algorithm. When $b(q) = \gamma_{i-1}$, a point x_r is marked with $b(r) = b(q) - \gamma_i$ $= |\gamma_{i-1}| - \gamma_i$. For notation purposes, let $N_{i-1} = s_0$. When $q = N_{i-1}$, let $s_1 = N$ after N has been incremented. We know that a point is added to the right of $x_{N_{i-1}}$ because $b(N_{i-1}) = |\gamma_{i-1}|$. Therefore, the point added to the right of $x_{N_{i-1}}$ is x_{s_1}.

By Theorem 3.2, a total of a_i points are marked to the right of each x_q with $b(q) = |\gamma_{i-1}|$. Repeat the above process: For $j = 2, 3, \ldots, a_i$, let $s_{j-2} < q \leq s_{j-1}$ (where the s_js are defined recursively through this process). Note that $j\gamma_i \leq a_i\gamma_i < |\gamma_{i-1}|$, so $|\gamma_{i-1}| - j\gamma_i > 0$. Adding γ_i to both sides of this inequality gives $|\gamma_{i-1}| - (j-1)\gamma_i > \gamma_i$. But the left side of this inequality equals $b(q)$ for q in this range; therefore a new point x_r is added with $b(r) = |\gamma_{i-1}| - j\gamma_i$. When $q = s_{j-1}$, let $s_j = N$ after N has been incremented. Thus the point added to the right of $x_{s_{j-1}}$ is x_{s_j}.

For $s_{a_i-1} < q \leq s_{a_i}$, $b(q) = |\gamma_{i-1}| - a_i\gamma_i$. Since $a_i = \left\lfloor \left| \frac{\gamma_{i-1}}{\gamma_i} \right| \right\rfloor$, $|\gamma_{i-1}| = a_i\gamma_i < \gamma_i$ by the definition of the floor function. Thus no new points are added. When $q = s_{a_i} = N$, the stage ends. Thus $N_i = s_{a_i}$.

Since $x_{N_i} = x_{s_{a_i}} > x_{s_{a_i-1}} > \cdots > x_{s_0} = x_{N_{i-1}}$, clearly $x_{N_i} > x_{N_{i-1}}$.

To prove the other inequality, recall that $b(N_{i-1}) = |\gamma_{i-1}|$. Every point x_r to the right of $x_{N_{i-1}}$ satisfies $x_r - x_{N_{i-1}} = |\gamma_{i-1}|$, by definition of b. Thus, using the hypothesis that $x_{N_{i-2}} > x_{N_{i-1}}$, we get $x_{N_{i-2}} - x_{N_{i-1}} \geq |\gamma_{i-1}|$.

The distance between x_{s_i} and $x_{s_{i+1}}$, $i = 0, 1, \ldots, a_i - 1$, is clearly γ_i by construction. Thus, using a telescoping sum, $x_{s_{a_i}} - x_{s_0} = x_{N_i} - x_{N_{i-1}} = a_i\gamma_i$. Since $a_i\gamma_i < |\gamma_{i-1}|$.

$$x_{N_i} - x_{N_{i-1}} \leq |\gamma_{i-1}|.$$

We know from multiplying the induction hypothesis by -1 that

$$x_{N_{i-1}} - x_{N_{i-2}} < -|\gamma_{i-1}|.$$

Adding these two inequalities gives

$$x_{N_i} - x_{N_{i-2}} < 0.$$

The desired conclusion follows from this and the conclusion above that $x_{N_i} > x_{N_{i-1}}$.

The proof for the case with i even is similar, except the signs are reversed in the induction hypothesis, the absolute value signs are used for γ_i instead of γ_{i-1} since $\gamma_i < 0$ and $\gamma_{i-1} > 0$, and $c(q)$ is evaluated in stage i instead of $b(q)$. ■

Another theorem gives a result on the lengths of the ith rotation sequence partition intervals containing the endpoints of $[0, 1]$.

Theorem 3.6 *Let $[0, x_a]$ and $[x_r, 1]$ be the intervals of the ith rotation sequence partition containing 0 and 1, $i \geq 1$. If i is odd, then $x_a - 0 = \gamma_i$ and $1 - x_r = |\gamma_{i+1}|$. If i is even, then $x_a - 0 = \gamma_{i+1}$ and $1 - x_r = |\gamma_i|$.*

Proof:
The statement will be proven by mathematical induction. It is easily verified that the result holds for $i = 1$.

Now assume that i is odd and, with $[0, x_a]$ and $[x_r, 1]$ intervals of the $(i-1)$st rotation sequence partition, $x_a - 0 = \gamma_i$ and $1 - x_r = |\gamma_{i-1}|$, since $i-1$ is even. Note that $b(1) = \gamma_i$, so that no point is marked in the partition interval to the right of $x_1 = 0$. Thus in the ith rotation sequence partition, the interval containing 0 is still $[0, x_a]$, and $x_a - 0 = \gamma_i$.

For $q = r$, $b(r) = |\gamma_{i-1}| > \gamma_i$, so a point x_{r_1} is marked to the right of x_r. By Theorem 3.2, at the end of stage i, a_i points $x_{r_1}, x_{r_2}, \ldots, x_{r_{a_i}}$, will be marked to the right of x_r. The interval $[x_{r_{a_i}}, 1]$ is therefore a partition interval of the ith rotation sequence partition, and by definition its length is $|\gamma_{i+1}|$. This establishes the induction step for i odd; the proof for i even is similar. ■

3.2.2 Further Conjectures on the Discrepancy of the Rotation Sequence

While evaluating the discrepancy for a number of different rotation sequences, I initially believed the following statement to be true. In order to account for the possibility of counting points in an interval $[a, 1]$, which is not contained in $[0, 1)$, we define $\#[a, 1] = \#[a, 1) + 1$, as if a point in the sequence existed at 1.

This is consistent with the circle model of the rotation sequence (Figure 3.1), where the point at 1 is identified with the point at 0, a point of the sequence.

Conjecture 3.1 *The discrepancy D_{N_i} for the points marked at the end of stage i for a rotation sequence is given by*

$$D_{N_i} = \left| \frac{\#[a,b]}{N} - (b-a) \right|,$$

where $[a,b]$ is the smallest closed interval containing all the ith rotation sequence partition intervals of width $|\gamma_{i+1}|$.

Unfortunately, I discovered a counterexample for which this does not hold. Let $\alpha = \frac{\sqrt{5}-1}{2}$, and construct the rotation sequence of fractional parts of multiples of this value. It is easily verified that $N_4 = 8$; the first eight values of the sequence (rounded to six decimal places) are given in Table 3.1 and Figure 3.3.

$$\begin{aligned} x_1 &= 0.000000 \\ x_2 &= 0.618034 \\ x_3 &= 0.236068 \\ x_4 &= 0.854102 \\ x_5 &= 0.472136 \\ x_6 &= 0.090170 \\ x_7 &= 0.708204 \\ x_8 &= 0.326238 \end{aligned}$$

Table 3.1: First eight values of rotation sequence for $\alpha = (\sqrt{5}-1)/2$

The smallest closed interval containing all the intervals of length γ_5 is $[x_1, x_7]$. By the conjecture, the discrepancy should equal

$$\left| \frac{\#[x_1, x_7]}{8} - (x_7 - x_1) \right|$$

$$= \left| \frac{7}{8} - (0.708204) \right| = 0.166796.$$

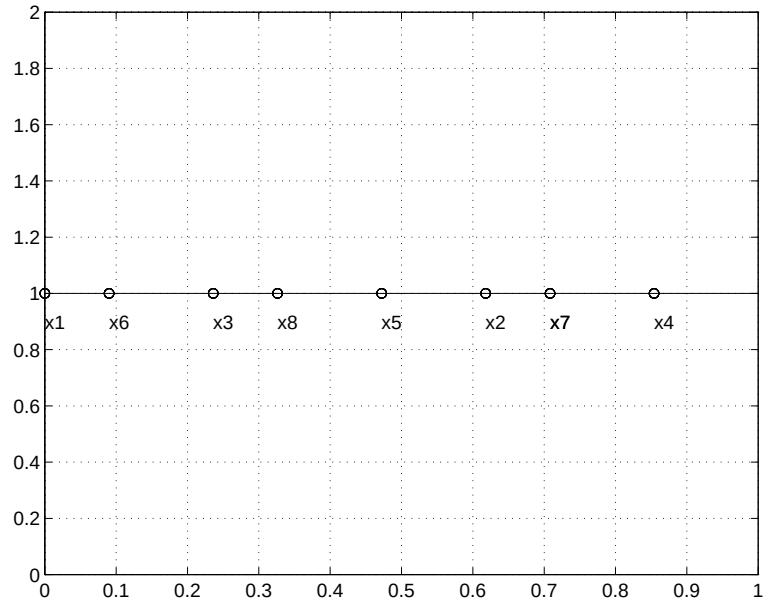

Figure 3.3: Plot of first eight values of rotation sequence for $\alpha = (\sqrt{5}-1)/2$

However,

$$\left|\frac{\#[x_1, x_8+\epsilon)}{8} - (x_8+\epsilon-x_1)\right|$$

$$= \left|\frac{4}{8} - (0.326238) - \epsilon\right| = 0.173762 - \epsilon$$

for arbitrarily small ϵ. Since discrepancy is a supremum over a set containing the value $0.173762 - \epsilon$, the discrepancy cannot equal 0.166796 as the conjecture implies. ■

After experimenting with more rotation sequences with different irrational numbers, I found that the following result seemed to be true, although I could not prove it. Again, $\#[a,1]$ is defined as $\#[a,1) + 1$, as if a point of the sequence existed at 1.

Conjecture 3.2 *Let ω be a rotation sequence. Then the discrepancy D_{N_i} at the end of stage i is given by*

$$\frac{\#[x_{N_i}, 1]}{N_i} - (1 - x_{N_i}), \quad \textit{for } i \textit{ odd},$$

$$\frac{\#[0, x_{N_i}]}{N_i} - (x_{N_i} - 0), \quad \textit{for } i \textit{ even}.$$

Pace and Salazar-Lazaro [7] attempted to study sequences generated by the rotation sequence algorithm but with the γ_i sequence defined differently. By changing the rotation algorithm slightly, they give a generalized algorithm which does not depend on the rotation properties of the γ_i, only that the sequence $\{\gamma_i\}$ is decreasing in absolute value. Their report gives data on discrepancy for a number of sequences formed with γ_i unrelated to the rotation sequence. Most of these examples did not appear to give a low-discrepancy sequence. They found an alternate recurrence relation for γ_i, similar to the one used in the rotation sequence, for which the sequences appeared to be low-discrepancy but the results were inconclusive.

3.2.3 Discrepancy Bound for the Rotation Sequence

Kuipers and Niederreiter [3] prove a result that the discrepancy of such a sequence is $O(\frac{\log N}{N})$ in the case that the partial quotients of the continued fraction sequence are bounded.

Theorem 3.7 *Let α be an irrational number with bounded partial quotients; i.e., for $\alpha = [a_0, a_1, a_2, \ldots]$, there exists an integer K with $a_i \leq K$ for $i \geq 1$. Let ω be the sequence formed from the fractional parts of the integer multiples of α; i.e., $\omega = \{\omega_i\}$ where $\omega_i = \{i\alpha\}$. Then the discrepancy $D_n(\omega)$ satisfies*

$$ND_N(\omega) \leq 3 + \left(\frac{1}{\log \xi} + \frac{K}{\log(K+1)}\right) \log N,$$

where $\xi = \frac{1+\sqrt{5}}{2}$.

Chapter 4

Two Dimensional Low Discrepancy Sequences

4.1 Defining Discrepancy in Two Dimensions

Before we define discrepancy in $\mathbf{R}^2$ we need to introduce some notation.

$$(\alpha^{(1)}, \beta^{(1)}) \times (\alpha^{(2)}, \beta^{(2)}) = \{(x, y) : \quad \alpha^{(1)} < x < \beta^{(1)} \quad \alpha^{(2)} < y < \beta^{(2)}\}$$

is an open rectangle in $\mathbf{R}^2$.

$$[\alpha^{(1)}, \beta^{(1)}] \times [\alpha^{(2)}, \beta^{(2)}] = \{(x, y) : \quad \alpha^{(1)} \leq x \leq \beta^{(1)} \quad \alpha^{(2)} \leq y \leq \beta^{(2)}\}$$

is a closed rectangle in $\mathbf{R}^2$. Similiarly, we can define a half open rectangle in $\mathbf{R}^2$. For $J = [\alpha^{(1)}, \beta^{(1)}) \times [\alpha^{(2)}, \beta^{(2)})$, we have

$$\#(J) \text{ is the cardinality of } \{x_i \in J : 1 \leq i \leq N\}$$

$$area(J) = (\beta^{(1)} - \alpha^{(1)})(\beta^{(2)} - \alpha^{(2)})$$

We can now define discrepancy for $\mathbf{R}^2$.

Definition 4.1 *Let $\{x_1, x_2, x_3, ..., x_N\}$ be a sequence in $\mathbf{R}^2$. Let $J = [\alpha^{(1)}, \beta^{(1)}) \times [\alpha^{(2)}, \beta^{(2)}) \in [0, 1)^2$. Then D_N is defined as follows.*

$$D_N = \sup_J \left| \frac{\#(J)}{N} - area(J) \right|$$

A definition of D_N^* for $\mathbf{R}^2$ can be found in [3].

4.2 Creating the Two Dimensional Cut and Stack Sequence (2DCS Sequence)

The first two dimensional sequence that we looked at is generated by a process of cutting and stacking pieces of $[0,1)^2$. We have called this the Two Dimensional Cut and Stack Sequence but we shall hereafter refer to this sequence as the 2DCS Sequence.

For a given integer p greater than 1 we "cut" $[0,1)^2$ into p^2 even parts. We then "stack" these parts on top of each other such that the piece that contains the origin is always at the bottom of the stack. We always stack in the same order. We can continue to cut and stack in this manner. The sequence points are found by mapping the point $(0,0)$ to the points directly above it in the stack. For instance, x_1 would be the point directly above $(0,0)$ and x_2 would be the point directly above x_1. Continuing in this manner we generate all of the points in the sequence.

The following is a figure of the 2DCS Sequence for $p = 2$ after three cuts.

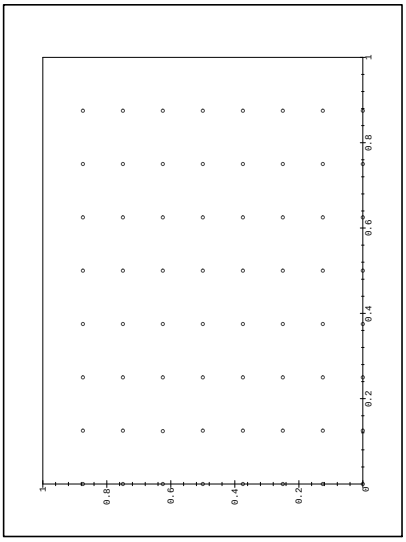

Figure 4.1: 2DCS Sequence for $p = 2$.

4.3 Algorithm to Create the 2DCS Sequence

When writing the algorithm for this sequence we thought of the x_m point as a vector move of some x_n point where $n < m$. The algorithm is written in two parts. The first part generates the vectors and the second part uses those vectors to generate the points of the sequence.

The following is the algorithm to create the sequence. For the time being ignore all sections surrounded by ** or @@.

2DCS Algorithm

Let p be an integer greater than 1. The pointer, R, points to the vector, V_i, that is being used. The counter, T, counts which round we are on. Initially, set $R = 1$, $T = 1$, and $V_0 = (0, 0)$. Then follow the following algorithm to create the vectors.

Loop
  $V_R = [V_{R-1} + (\frac{1}{p}, \frac{p-1}{p})]\text{mod}(1, 1)$
  $R = R + 1$
until $R\text{mod}p = 0$
Loop
  $V_R = V_0 + (0, \frac{T}{p})$
  $R = R + 1$
  loop
    $V_R = [V_{R-1} + (\frac{1}{p}, \frac{p-1}{p})]\text{mod}(1, 1)$
    $R = R + 1$
  until $R\text{mod}p = 0$
  $T = T + 1$
until $R = p^2$

Let p be an integer greater than 1. The pointer, q, points to the x_i that is being used and endlist, N, points to the last x_i added to the sequence. The pointer, y, points to which V_y you're using and s is a counter. **θ is the angle of rotation.** Initially, set $x_0 = (0, 0), q = 0, N = 0, y = 1$, and $s = 0$. **Choose $0 < \theta < \frac{\pi}{2}$.** Then follow the following algorithm for $n = 1, 2, 3, ...$ to create the sequence.

Loop
  $M = V_y$
  loop
    $x_{N+1} = x_q + (\frac{1}{p^{n-1}})M$

$$@@x_{N+1} = [[x_q \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} + (\tfrac{1}{p^{n-1}})M] \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}] \mathrm{mod}(1,1)@@$$

$N = N + 1$

$q = q + 1$

until $q - 1 = s$

set $q = 0$

$y = y + 1$

until $y = p^2$

set $s = N$

set $y = 1$

4.4 Calculating D_N of the 2DCS Sequence

Using only the information that we have so far we would need to use an infinite number of half open rectangles in order to calculate the discrepancy of this sequence. Since this is impossible to do we have proved a theorem that lets us use only a finite number of open and closed rectangles. In order to prove this we first prove several lemmas which require the following definitions and notation.

Definitions and Notation

$\mathcal{A} = \{[\alpha, \beta) \times [\gamma, \delta) : [\alpha, \beta), [\gamma, \delta) \subseteq [0,1)\}$;

$\{x_1, x_2, ..., x_N\} = \omega$ is a sequence of points in $[0, 1)^2$;

N is a positive number;

$x_0 = (0, 0), x_{N+1} = (1, 1)$;

For a fixed N, #(A) is the cardinality of $\{i \in \mathbf{N} : 1 \leq i \leq N, x_i \in A\}$, intuitively the number of points of ω in A

The discrepancy $D_N(\omega) = \sup_{A \in \mathcal{A}} \left| \frac{\#(A)}{N} - area(A) \right|$;

$a^{(i)}$ represents the i^{th} coordinate of a; i.e., for a in $\mathbf{R}^2$, $a = (a^{(1)}, a^{(2)})$;

$\mathcal{B}_1 = \{[x_i^{(1)}, x_j^{(1)}] \times [x_k^{(2)}, x_l^{(2)}] : i, j, k, l \in \{1, 2, ..., N\}, x_i^{(1)} \leq x_j^{(1)}, x_k^{(2)} \leq x_l^{(2)}\}$

$\mathcal{B}_2 = \{(x_i^{(1)}, x_j^{(1)}) \times (x_k^{(2)}, x_l^{(2)}) : i, j, k, l \in \{1, 2, ..., N\}, x_i^{(1)} < x_j^{(1)}, x_k^{(2)} < x_l^{(2)}\}$

Lemma 4.1 *If* $X \in \mathcal{B}_1$, $\left| \frac{\#(X)}{N} - area(X) \right| \leq D_N(\omega)$.

Proof: Let $X = [x_i^{(1)}, x_j^{(1)}] \times [x_k^{(2)}, x_l^{(2)}]$. Thus, $\left|\frac{\#(X)}{N} - area(X)\right|$
$= \left|\frac{\#(X)}{N} - (x_j^{(1)} - x_i^{(1)})(x_l^{(2)} - x_k^{(2)})\right|$. Now for $\epsilon > 0$, define $X_\epsilon = [x_i^{(1)}, x_j^{(1)} + \epsilon) \times [x_k^{(2)}, x_l^{(2)} + \epsilon)$. For small enough ϵ, $X_\epsilon \subseteq [0,1)^2$ and $\#(X_\epsilon) = \#(X)$. Fix an ϵ that satisfies these. Since $X_\epsilon \subseteq [0,1)^2, \epsilon < 1$.

Now, $\left|\frac{\#(X)}{N} - area(X)\right|$
$= \left|\frac{\#(X_\epsilon)}{N} - (x_j^{(1)} + \epsilon - x_i^{(1)})(x_l^{(2)} + \epsilon - x_k^{(2)}) + (-x_i^{(1)} + x_j^{(1)} - x_k^{(2)} + x_l^{(2)})\epsilon + \epsilon^2\right|$
$\leq \left|\frac{\#(X_\epsilon)}{N} - area(X_\epsilon)\right| + \left|(x_i^{(1)} - x_j^{(1)} + x_k^{(2)} - x_l^{(2)})\epsilon\right| + \epsilon^2$
(Triangle Inequality)
$\leq \left|\frac{\#(X_\epsilon)}{N} - area(X_\epsilon)\right| + 5\epsilon$ (since $0 < x_i^{(1)}, x_j^{(1)}, x_k^{(2)}, x_l^{(2)}, \epsilon, \epsilon^2 < 1$)
$\leq D_N(\omega) + 5\epsilon$ (definitions of discrepancy and supremum, since $X_\epsilon \in \mathcal{A}$).

Since $\left|\frac{\#(X)}{N} - area(X)\right| \leq D_N(\omega) + 5\epsilon$ for an arbitrary small ϵ, it must follow that for $X \in \mathcal{B}_1$, $\left|\frac{\#(X)}{N} - area(X)\right| \leq D_N(\omega)$. ■

Lemma 4.2 *If $X \in \mathcal{B}_2$, $\left|\frac{\#(X)}{N} - area(X)\right| \leq D_N(\omega)$.*

Proof: Let $X = (x_i^{(1)}, x_j^{(1)}) \times (x_k^{(2)}, x_l^{(2)})$. Thus, $\left|\frac{\#(X)}{N} - area(X)\right|$
$= \left|\frac{\#(X)}{N} - (x_j^{(1)} - x_i^{(1)})(x_l^{(2)} - x_k^{(2)})\right|$. Now for $\epsilon > 0$, define $Y_\epsilon = [x_i^{(1)} + \epsilon, x_j^{(1)}) \times [x_k^{(2)} + \epsilon, x_l^{(2)})$. For small enough ϵ, $Y_\epsilon \subseteq X \subseteq [0,1)^2$ and $\#(Y_\epsilon) = \#(X)$. Fix an ϵ that satisfies these. Since $Y_\epsilon \subseteq [0,1)^2, \epsilon < 1$.

Now, $\left|\frac{\#(X)}{N} - area(X)\right|$
$= \left|\frac{\#(Y_\epsilon)}{N} - (x_j^{(1)} - \epsilon - x_i^{(1)})(x_l^{(2)} - \epsilon - x_k^{(2)}) - (-x_i^{(1)} + x_j^{(1)} - x_k^{(2)} + x_l^{(2)})\epsilon + \epsilon^2\right|$
$\leq \left|\frac{\#(Y_\epsilon)}{N} - area(Y_\epsilon)\right| + \left|-(-x_i^{(1)} + x_j^{(1)} - x_k^{(2)} + x_l^{(2)})\epsilon\right| + \epsilon^2$
(Triangle Inequality)
$= \left|\frac{\#(Y_\epsilon)}{N} - area(Y_\epsilon)\right| + (-x_i^{(1)} + x_j^{(1)} - x_k^{(2)} + x_l^{(2)})\epsilon + \epsilon^2$ (since $0 \leq x_i^{(1)}, x_j^{(1)}, x_k^{(2)}, x_l^{(2)}, \epsilon, \epsilon^2 < 1$)
$\leq \left|\frac{\#(Y_\epsilon)}{N} - area(Y_\epsilon)\right| + 5\epsilon$ (as in Lemma 4.1)

Since $\left|\frac{\#(X)}{N} - area(X)\right| \leq D_N(\omega) + 5\epsilon$ for an arbitrary small ϵ, it must follow that for $X \in \mathcal{B}_2$, $\left|\frac{\#(X)}{N} - area(X)\right| \leq D_N(\omega)$. ■

Lemma 4.3 *Given N and ω let $M = D_N(\omega)$. Then for all ϵ such that $0 \leq \epsilon \leq M$, there exists $S \in (\mathcal{B}_1 \cup \mathcal{B}_2)$, such that $M - \epsilon < \left|\frac{\#(S)}{N} - area(S)\right| \leq M$.*

Proof:

Fix $0 < \epsilon < M$. From the definitions of discrepancy and supremum there exists $T = [a^{(1)}, b^{(1)}) \times [a^{(2)}, b^{(2)}) \subseteq \mathcal{A}$ such that $M - \epsilon < \left| \frac{\#(T)}{N} - area(T) \right| \leq M$.

Since $M > \frac{1}{N} > 0$ [3] we have $\frac{\#(T)}{N} - area(T) \neq 0$. There are two cases.

Case 1. $\frac{\#(T)}{N} - area(T) > 0$. Since area is always non-negative there must be at least one point of the sequence inside T. Let $i, j \in \{1, 2, ..., N\}$ be such that $[x_i^{(1)}, x_j(1)] \subset [a^{(1)}, b^{(1)})$ and for all other $k, l \in \{1, 2, ..., N\}$ with $[x_k^{(1)}, x_l^{(1)}] \subset [a^{(1)}, b^{(1)}), x_j^{(1)} - x_i^{(1)} \geq x_l^{(1)} - x_k^{(1)}$. Similiarly, let $p, q \in \{1, 2, ..., N\}$ be such that $[x_p^{(2)}, x_q^{(2)}] \subset [a^{(2)}, b^{(2)})$ and for all other $k, l \in \{1, 2, ..., N\}$ with $[x_k^{(2)}, x_l^{(2)}] \subset [a^{(2)}, b^{(2)}), x_q^{(2)} - x_p^{(2)} \geq x_l^{(2)} - x_k^{(2)}$. Intuitively, $S = [x_i^{(1)}, x_j^{(1)}] \times [x_p^{(2)}, x_q^{(2)}]$ is a rectangle which, when the sides are extended, they contain at least one sequence point with $S \subset T$. ($i = j$, $p = q$ are possible; in this case S would be a rectangle with area 0).

Now we have $\#(S) = \#(T)$ and $area(S) < area(T)$.

Thus, $M - \epsilon < \frac{\#(T)}{N} - area(T) < \frac{\#(S)}{N} - area(S) \leq M$ from Lemma 4.1.

Since $\frac{\#(S)}{N} - area(S) > 0$ we have $M - \epsilon < \left| \frac{\#(S)}{N} - area(S) \right| \leq M$.

Case 2. $area(T) - \frac{\#(T)}{N} > 0$. If $a^{(1)} \neq 0$ then let $i, j \in \{0, 1, ..., N+1\}$ be such that $(x_i^{(1)}, x_j^{(1)}) \supset [a^{(1)}, b^{(1)})$, and for all other $k, l \in \{0, 1, ..., N+1\}$ with $(x_k^{(1)}, x_l^{(1)}) \supset [a^{(1)}, b^{(1)}), x_j^{(1)} - x_i^{(1)} \leq x_l^{(1)} - x_k^{(1)}$. If $a^{(1)} = 0$ then let $i = 0$ and j be defined such that $x_j^{(1)} > b^{(1)}, x_j^{(1)} - b^{(1)} < x_k^{(1)} - b^{(1)}$ for $k \in \{0, 1, ..., N+1\}$. Similiarly, if $a^{(2)} \neq 0$ let $p, q \in \{0, 1, ..., N+1\}$ be such that $(x_p^{(2)}, x_q^{(2)}) \supset [a^{(2)}, b^{(2)})$ and, for all other $k, l \in \{0, 1, ..., N+1\}$ with $(x_k^{(2)}, x_l^{(2)}) \supset [a^{(2)}, b^{(2)}), x_q^{(2)} - x_p^{(2)} \leq x_l^{(2)} - x_k^{(2)}$. If $a^{(2)} = 0$ then let $p = 0$ and q be defined such that $x_q^{(2)} > b^{(2)}, x_q^{(2)} - b^{(2)} < x_k^{(2)} - b^{(2)}$ for $k \in \{0, 1, ..., N+1\}$. Let $S = (x_i^{(1)}, x_j^{(1)}) \times (x_p^{(2)}, x_q^{(2)})$.

Now we have $\#(S) \leq \#(T)$ and $area(s) \geq area(T)$.

Thus, $M - \epsilon < area(T) - \frac{\#(T)}{N} \leq area(S) - \frac{\#(S)}{N} \leq M$ from Lemma 4.2.

Since $area(S) - \frac{\#(S)}{N} > 0$ we have $M - \epsilon < \left| \frac{\#(S)}{N} - area(S) \right| \leq M$.

Combining these two cases we have $\exists S \in (\mathcal{B}_1 \cup \mathcal{B}_2)$ such that $M - \epsilon < \left| \frac{\#(S)}{N} - area(S) \right| \leq M$. ■

Theorem 4.1 *For any sequence, ω, and any N,*

$D_N(\omega) = \max_{S \in (\mathcal{B}_1 \cup \mathcal{B}_2)} \left\{ \left| \frac{\#(S)}{N} - area(S) \right| \right\}$

Proof: We know $\left\{\left|\frac{\#(S)}{N} - area(S)\right| : S \in (\mathcal{B}_1 \cup \mathcal{B}_2)\right\}$ is finite so it has a maximum element P.

Let $S_P \in (\mathcal{B}_1 \cup \mathcal{B}_2)$ be such that $\left|\frac{\#(S_P)}{N} - area(S_P)\right| = P$. From Lemma 4.1 and 4.2, we have $P \leq D_N(\omega)$.

Assume that $P < D_N(\omega)$. Then $D_N(\omega) - P = \epsilon > 0$. From Lemma 4.3 there exists $S \in (\mathcal{B}_1 \cup \mathcal{B}_2)$ such that $P = D_N(\omega) - \epsilon < \left|\frac{\#(S)}{N} - area(S)\right| \leq D_N(\omega)$. This is a contradiction because P is the maximum element of a set containing $\left|\frac{\#(S)}{N} - area(S)\right|$ it cannot be less than that element. Therefore $P = D_N(\omega)$. So we have that $D_N(\omega) = \max_{S \in (\mathcal{B}_1 \cup \mathcal{B}_2)} \left\{\left|\frac{\#(S)}{N} - area(S)\right|\right\}$. ■

We wrote a Matlab$^{\text{TM}}$ script that takes as input a sequence of N points in $\mathbf{R}^2$, imnplemented as an $N \times 2$ matrix, and returns D_N for that sequence. The program can be found at /amaterasu/sd2d/reu97/simmons/disc2.m.

4.5 Theorem for discrepancy of the 2DCS Sequence

Using the above results we can now prove the following theorem for discrepancy of our sequence.

Theorem 4.2 *Fix a positive integer p greater than 1. Let ω be the 2DCS Sequence on $[0,1)^2$ for p. Let $N = p^{2n}$, then the discrepancy, $D_N(\omega)$ is $\frac{2\sqrt{N}-1}{N}$. Furthermore, $D_N(\omega) = \left|\frac{\#(S)}{N} - area(S)\right|$ for S either $(0,1)^2$ or $\left[0, \frac{\sqrt{N}-1}{\sqrt{N}}\right]^2$.*

Proof:

We first show that for any open rectangle, $S \in \mathcal{B}_2$, the estimate of D_N will be the largest when $S = (0,1)^2$.

Let $f(m,n) = \left|\frac{\#(S)}{N} - area(S)\right|$ where S is a $\frac{m}{\sqrt{N}} \times \frac{n}{\sqrt{N}}$ rectangle such that $S \in \mathcal{B}_2$; $m, n \in \mathbf{Z}$ and $1 \leq m, n \leq \sqrt{N}$. Now, for any such S, $f(m,n) = \left|\frac{(m-1)(n-1)}{N} - \frac{nm}{N}\right| = \frac{m+n-1}{N}$. $f(m,n)$ is a maximum when $m, n = \sqrt{N}$. So the maximum value of $f(m,n)$ is $\frac{2\sqrt{N}-1}{N}$. For $m, n = \sqrt{N}$, S is a 1×1 rectangle so S must be $(0,1)^2$.

Now show that for any closed rectangle, $S \in \mathcal{B}_1$, the estimate of D_N will be the largest when $S = \left[0, \frac{\sqrt{N}-1}{\sqrt{N}}\right]^2$. This will require looking at several cases. Let $g(m,n) = (\frac{\#(S)}{N} - area(S))$ where S is a $\frac{m}{\sqrt{N}} \times \frac{n}{\sqrt{N}}$ rectangle such that $S \in \mathcal{B}_1$; $m, n \in \mathbf{Z}$ and $1 \leq m, n \leq \sqrt{N}$. Let $l^2 = N$.

Case 1: Suppose we have an $\frac{m}{\sqrt{N}} \times \frac{n}{\sqrt{N}}$ rectangle S such that $S \in \mathcal{B}_1$, S has a vertex at (0,0); $m, n \in \mathbf{Z}$ and $0 \leq m, n \leq l-1$. Then $g(m,n) = \frac{(m+1)(n+1)}{N} - \frac{mn}{N} = \frac{m+n+1}{N}$. Notice in this case $g(m,n) \geq 0$ so we do not need to take the absolute value to get an estimate of discrepancy. Here $g(m,n)$ is maximized when $m, n = l-1 = \sqrt{N}-1$. The maximum value, is then $\frac{2\sqrt{N}-1}{N}$. Notice for $m, n = \sqrt{N}-1, S$ is $[0, \frac{\sqrt{N}-1}{N}]^2$.

Case 2: Suppose (0,0) were not a vertex of our $\frac{m}{\sqrt{N}} \times \frac{n}{\sqrt{N}}$ rectangle S with $S \in \mathcal{B}_1; m, n \in \mathbf{Z}$ and $0 \leq m, n \leq l-1$. We can compare the estimate of discrepancy that we get with S with the rectangle "shifted" so that a vertex is at (0,0). This brings about two situations.

Situation 1: Suppose S is such that none of its sides lays along the line from (0,1) to (1,1) or (1,0) to (1,1). Then the estimate for discrepancy will be the same as the "shifted" estimate since the two areas are equal as are the number of points in both rectangles.

Situation 2: Suppose S is such that at least one of its sides lays along the line from (0,1) to (1,1) or (1,0) to (1,1). Then there is at least one vertex of S that is not a point of the sequence. So the "shifted" rectangle has a greater number of points and therefore has a higher discrepancy estimate. Notice that the "shifted" rectangle corresponds to a rectangle from Case 1.

Note that Case 1 yields the highest discrepancy estimate so far.

Case 3: Let m and n be equal to l. So S is a 1×1 closed rectangle or, in other words, S is $[0,1]^2$. For such an S, $g(m,n) = \frac{N}{N} - 1 = 0$. Since our discrepancy estimate is zero we have that the highest descrepancy estimate is still found in Case 1.

Case 4: Let m or n be equal to l. Without loss of generality assume that we have an mxl rectangle S such that $S \in \mathcal{B}_1, m \in \mathbf{Z}$ and $0 \leq m \leq l-1$. In this case $g(m,n) = \frac{\sqrt{N}(m+1)}{N} - \frac{m\sqrt{N}}{N} = \frac{\sqrt{N}}{N}$. For $N \geq 1$ (since $N = p^{2n}$ this will always hold) the maximum estimate of discrepancy from Case 1 is greater than or equal to this estimate.

In conclusion, we have that for any closed rectangle $S \in \mathcal{B}_1$, the estimate of discrepancy is maximized when $S = \left[0, \frac{\sqrt{N}-1}{\sqrt{N}}\right]^2$.

From Theorem 4.1 we have that at the end of the n^{th} level with $N = p^{2n}, D_N(\omega) = \frac{2\sqrt{N}-1}{N}$. ■

Now that we have a measure for the discrepancy of our sequence we can use it for Quasi-Monte Carlo Integration.

4.6 Evaluating the quality of the discrepancy for the 2DCS Sequence

We know that this sequence is a low-discrepancy sequence by Definition 2.4 since $\lim_{N\to\infty} \frac{2\sqrt{N}-1}{N} = 0$. Despite the fact that this is a low-discrepancy sequence the next step was to ask ourselves how "good" our measure of discrepancy was. Although it is true that we can make the discrepancy as low as we want by simply increasing the number of points in the sequence, this is not very practical. As the number of points that we have to use for Quasi-Monte Carlo Integration increases so does the computation time. This naturally leads to many questions.

Is our discrepancy good enough?

Is there some other sequence such that for all N its discrepancy is lower?

4.7 Transformations of the 2DCS Sequence

At this point we decided to look at doing transformations on the 2DCS Sequence that might yield sequences of lower discrepancy.

The transformation that we tried was to rotate the 2DCS Sequence by θ where $0 < \theta < \frac{\pi}{2}$.

The algorithm for the Rotated Sequence is the same as the algorithm for the 2DCS Sequence with some minor revisions. Insert all information surrounded by **. For the line surrounded by @@ delete the previous line and insert this one.

At the end of this chapter are some examples of rotated sequences.

Notice that these rotated sequences still have the points forming squares but these squares are at an angle of θ to the x-axis so that when we try to find discrepancy by using closed and open rectangles in $(\mathcal{B}_1 \cup \mathcal{B}_2)$ we see that

it looks like these rectangles cannot have sequence points on more than two vertices which makes it more difficult to find discrepancy of this sequence. Our hope is that the discrepancy is better for this sequence.

Unfortunately, we did not have enough time to calculate the discrepancy of this sequence.

4.8 Unanswered Questions

We still have many questions about our sequences including these.

What is the discrepancy of the rotated sequence?

Is it a better discrepancy than that of the 2DCS Sequence?

Would other transformations on the 2DCS Sequence yield lower or higher discrepanies?

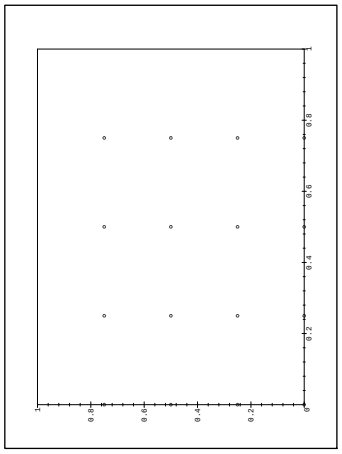

Figure 4.2: 2DCS, p=2, 2 cuts.

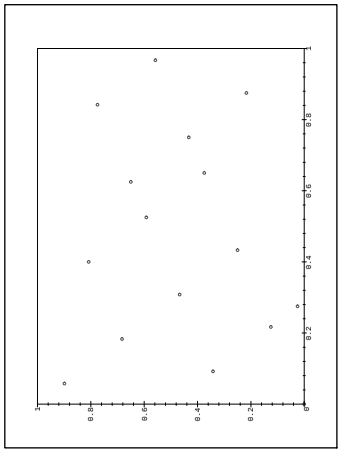

Figure 4.3: 2DCS, p=2, 2 cuts, rotated 30 Degrees.

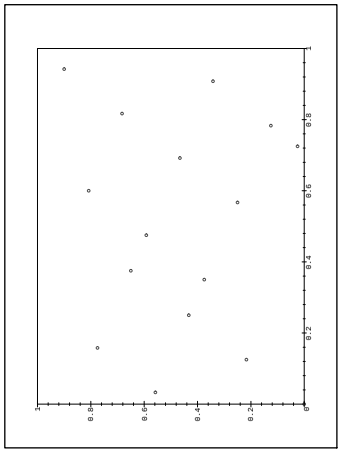

Figure 4.4: 2DCS, p=2, 2 cuts, rotated 60 Degrees.

Chapter 5

Using Low-Discrepancy Sequences in Quasi-Monte Carlo Integration

The purpose of this chapter is to show applications of Quasi-Monte Carlo Integration using low-discrepancy sequences described in Chapters 3 and 4.

The first sequence that we used in the application of Quasi-Monte Carlo Integration was the rotation sequence for $\frac{\sqrt{5}-1}{2}$, the fractional part of the golden mean. Using a program from [7], we found that the discrepancy for this sequence with $N = 300$ points is 0.0127269. We integrated three different functions in one dimension and used the sequence points as the nodes. The results are listed in Table 5.1. Note that $[a, b]$ is the interval over which we are integrating.

function	a	b	QMCI	exact	error
$f(x) = \frac{e^{\frac{-x^2}{2}}}{\sqrt{2\pi}}$	-1	1	0.682309	0.6826	0.04
$f(x) = \cos x$	$-\frac{\pi}{2}$	$\frac{\pi}{2}$	1.996209	2.0	0.19
$f(x) = x^6 + 4x^3 + 5x + 2$	0	2	48.210545	$\frac{338}{7}$	0.16

Table 5.1: Quasi-Monte Carlo integration of some functions using the rotation sequence for $(\sqrt{5}-1)/2$ and 300 points

function	a	b	c	d	QMCI	exact	error
$f(x,y) = xy$	0	1	0	1	0.191396	.25	23.44
$f(x,y) = \cos x + \sin y$	$\frac{\pi}{6}$	$\frac{\pi}{3}$	$-\frac{\pi}{2}$	$\frac{\pi}{2}$	0.981513	$\frac{(\sqrt{3}-1)\pi}{2}$	14.64
$f(x,y) = x^2 + y^2 + 4$	0	1	0	2	10.734367	$\frac{34}{3}$	7.93

Table 5.2: Quasi-Monte Carlo integration of some functions using the 2DCS sequence for $p = 2$ and three cuts

The second sequence that we used in the application of Quasi-Monte Carlo Integration was the Two Dimensional Cut and Stack Sequence for $p = 2$ after three cuts. The discrepancy of this sequence is 0.234375. Again we integrated three different functions, only in two dimensions instead of one. The results are listed in Table 5.2. Note that $[a, b]$ is the interval over which x is integrated and $[c, d]$ is the interval over which y is integrated.

The discrepancy for the sequence of 2DCS nodes is much larger than the discrepancy for the rotation sequence. Also, only 64 nodes of the 2DCS sequence were used, whereas 300 nodes were used for the rotation sequence. As expected, the percent errors for integration using the 2DCS sequence are larger than those using the rotation sequence.

Below is a program which implements Quasi-Monte Carlo Integration for two dimensional functions. The program that does this for one dimension can be found in Chapter 1.

```
/*ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c       Ian Winokur
c       Date started:  July 23, 1997           Last updated:  July 30, 1997
c
c       This program is an application of a Monte Carlo Method of
c       Integration using points from a low-discrepancy, uniformly distributed
c       two dimensional sequence instead of using random points.  This program
c       evaluates each point, averages the function values, and then multiplies
c       this average by the area length of the region being integrated to
c       obtain an approximation of the integral. This technique uses the
c       definition of the average value of a function.
c
c       Variable directory:
c
c               x           the name of each of the x-coordinates being used
c               y           the name of each of the y-coordinates being used
c               n           counts the number of points being used
c               a,b         the endpoints of the interval along the x-axis
c               c,d         the endpoints of the interval along the y-axis
c               sum         contains the sum of the f(xi,yi)
c               integral    the approximation of the integral of f
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc*/
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

float f(float x,float y);         /*  f is the function being integrated */

void main(void)
{
 /*   variable declarations  */

int n;
float a, b, c, d, sum = 0.0, x, y, integral;
```

```
scanf("%f %f",&a,&b);   /*  read in a and b  */
scanf("%f %f",&c,&d);   /*  read in c and d  */

 /*  read first point in the sequence  */

scanf("%f %f",&x,&y);
for (n = 0; x != -99.0; ++n)
{
 /*  scale values  */

 x = a + (b - a) * x;
 y = c + (d - c) * y;

 sum = sum + f(x,y);
 scanf ("%f %f",&x,&y); /* get next point in the sequence */
}

/*  calculate integral */

integral =  (b - a) * (d - c) * sum/ (float) n;

/*  output results  */

printf("Function being evaluated:   z = x^2 + y^2 + 4\n");
printf("Number of nodes used:       %d\n",n);
printf("Interval along the x-axis:  [%f,%f]\n",a,b);
printf("Interval along the y-axis:  [%f,%f]\n",c,d);
printf("\n\nApproximation:  %f\n\n",integral);
exit (0);
};

float f(float x,float y)
/*  this function is the one being integrated  */
{
  return(x*x+y*y+4);                /* put function here  */
}; /*  end of function f  */
```

Bibliography

[1] Grimmett, G.R., and D.R. Stirzaker. *Probability and Random Processes,* 2nd ed. Oxford Science Publications, 1992.

[2] Khinchin, A. Ya. *Continued Fractions.* 3rd ed. Chicago: The University of Chicago Press, 1964.

[3] Kuipers, Lauwerens, and Harald Niederreiter. *Uniform Distribution of Sequences.* New York: John Wiley & Sons, 1974.

[4] Mount Holyoke College. *Laboratories in Mathematical Experimentation, A Bridge to Higher Mathematics.* Springer, 1997.

[5] Niederreiter, Harald. *Random Number Generation and Quasi-Monte Carlo methods.* Philadelphia: Society for Industrial and Applied Mathematics, 1992.

[6] Niven, Ivan, Herbert S. Zuckerman, and Hugh L. Montgomery. *An Introduction to the Theory of Numbers.* 5th ed. New York: John Wiley & Sons, 1991.

[7] Pace, Laura A., and Carlos Salazar-Lazaro. Uniformly distributed sequences and their discrepancies. REU paper, Oregon State University, 1996.

[8] Stone, C.J. *A Course in Probability and Statistics.* Duxbury Press, 1996.