Numerical Integration of High-Dimensional Walsh Series and Concrete Parallel Calculations

Wolfgang Ch. Schmid

RIST++
Research Institute for Software Technology
University of Salzburg

0 Abstract

Functions which can be represented as Walsh series play an important role in the context of image processing and signal processing. For several applications good methods for the numerical integration of such functions are needed.

We present a method for the numerical integration of such functions by number theoretical means which was developed in analogy to the classical theory of good lattice points in [7] and which was optimized for higher dimensions in [6].

This method shows excellent theoretical error estimates but it turned out to be numerically highly intensive so that parallel methods had to be used for testing and applying the method in concrete calculations.

The implementation (see sections 5 and 6) is part 4 of the NEWTON- (New Technology of Numerics) Project, which is devoted to the development and practical applications of Number Theoretical Numerics. NEWTON encompasses theoretical investigations and implementations onto parallel computers.

1 Introduction

The fundamental concept in the theory of number-theoretical methods for numerical integration is the concept of good lattice points. Good lattice point methods are used for the numerical integration of functions which are representable by, in some sense rapidly converging, Fourier series. The basic result in this theory is the following:

Let s be any dimension with $s \geq 2$. For every $\alpha > 1$ and $c \in \mathbb{R}$ let $\mathrm{E}_s^{\alpha}(c)$be the class of all functions $f : [0,1)^s \longrightarrow \mathbb{R}$, periodic with period one in each variable, which are representable by absolutely converging Fourier series with Fourier coefficients $\hat{f}(h_1, \ldots, h_s)$ with the following property:

$$|\hat{f}(h_1, \ldots, h_s)| \leq \frac{c}{(\overline{h}_1 \cdots \overline{h}_s)^{\alpha}} \text{ for all } h_1, \ldots, h_s \in \mathbb{Z} \ ,$$

where $\overline{h} := \max(1, |h|)$.

Then we have:
For every $\alpha > 1$ there is a constant $c'(s, \alpha, c)$ such that for all $N \in \mathbb{N}$ there exist a so-called good lattice point $a_1, \ldots, a_s \in \mathbb{Z}_s$ such that the integration error

$$\begin{aligned} R_N(f) \ &:= \ \left| \int_{[0,1)^s} f(\underline{x})\, d\underline{x} - \frac{1}{N} \sum_{k=1}^{N} f\left(\left\{ k\frac{a_1}{N} \right\}, \ldots, \left\{ k\frac{a_s}{N} \right\} \right) \right| \leq \\ &\leq \ c'(s, \alpha, c) \frac{(\log N)^{\alpha \cdot (s-1)}}{N^{\alpha}} \end{aligned}$$

for all $f \in \mathrm{E}_s^{\alpha}(c)$. This holds for $N = p$ prime [1] and in a slightly weaker form for arbitrary N [11].

Therefore good lattice points provide very well distributed point-sets which are excellent for the numerical integration of rapidly converging Fourier series.

One main problem in this theory is that the above results are just existence results. For $s \geq 3$ no explicit fast construction methods are known and one essentially has to use tabulated values.

In recent time, especially in the context of image processing and signal processing, and also for Image-Data-Compression (see for example [13, 8]), it has become necessary to give good effective methods for the numerical integration of functions which are representable by Walsh series. We give the basic concepts and the basic result of this theory:

Definition 1 *The Walsh functions to base $b \geq 2$ are defined by:*

$$wal_0^{(b)}(x) := 1$$

and, if n is represented to base b by $n = a_0 + a_1 b^1 + \ldots + a_m b^m$, then

$$wal_n^{(b)}(x) := \phi^{a_0}(x) \cdot \phi^{a_1}(bx) \cdots \phi^{a_m}(b^m x) \ ,$$

where

$$\phi(x) := e^{\frac{2\pi i k}{b}} \text{ for } \frac{k}{b} \le x < \frac{k+1}{b} \ ; \ k = 0, \ldots, b-1$$

means the first Rademacher function to base b . For dimension $s \ge 2$ and $k_1, \ldots, k_s \ge 0$ we define

$$wal_{k_1, \ldots, k_s}^{(b)}(x_1, \ldots, x_s) := wal_{k_1}^{(b)}(x_1) \cdots wal_{k_s}^{(b)}(x_s) \ .$$

The set $W = \{\mathrm{wal}_{k_1, \ldots, k_s}^{(b)}(x_1, \ldots, x_s) \mid k_i \ge 0; i = 1, \ldots, s\}$ is an orthonormal and complete system of functions in $L^2([0,1)^s)$.

In analogy to the good lattice point method we give the following definition:

Definition 2 *For base $b \ge 2$, dimension $s \ge 1$ and $\alpha > 1, c > 0 (b, s \in \mathbb{N}; c, \alpha \in \mathbb{R})$ let ${}_b\bar{E}_s^{\alpha}(c)$ be the class of all functions $f : [0,1)^s \longrightarrow \mathbb{C}$ which are representable by absolutely converging Walsh series*

$$\sum_{h_1, \ldots, h_s = 0}^{\infty} \hat{W}(h_1, \ldots, h_s) \cdot wal_{h_1, \ldots, h_s}^{(b)}(x_1, \ldots, x_s) \ ,$$

with Walsh coefficients $\hat{W}(h_1, \ldots, h_s)$ with the following property:

$$|\hat{W}(h_1, \ldots, h_s)| \le \frac{c}{(\overline{h}_1 \cdots \overline{h}_s)^{\alpha}} \text{ for all } (h_1, \ldots, h_s) \ne (0, \ldots, 0) \ .$$

($\overline{h} := \max(1, |h|)$)

Then, for given N, we are asking for point-sets $\underline{x}_1, \ldots, \underline{x}_N$ in $[0,1)^s$ such that the integration error

$$R_N(f) := \left| \int_{[0,1)^s} f(\underline{x})\, d\underline{x} - \frac{1}{N} \sum_{k=1}^{N} f(\underline{x}_k) \right|$$

is small for all $f \in {}_b\bar{E}_s^{\alpha}(c)$.

It turnes out that point-sets satisfying these requirements in a best possible way are provided by so-called (t, m, s)-nets to base b which can be defined by the following definitions [9]:

Definition 3 *For an integer $b \geq 2$ an elementary interval to base b is defined to be an interval of the form*

$$\prod_{i=1}^{s} \left[\frac{a^{(i)}}{b^{d_i}}, \frac{a^{(i)}+1}{b^{d_i}} \right)$$

with integers $d_i, a^{(i)} \geq 0$ and $a^{(i)} < b^{d_i}$ for $1 \leq i \leq s$.

Definition 4 *Let $b \geq 2$ and $0 \leq t \leq m$ be integers. A (t,m,s)-net to base b is a set of b^m points in $[0,1)^s$ for which we have that in every elementary interval to base b of volume $\frac{1}{b^{m-t}}$ there are exactly b^t points of the set.*

Such (t,m,s)-nets have excellent distribution properties if t is small (the optimal case of course is $t=0$) as was pointed out in detail in [9].

If we use (t,m,s)-nets for the numerical integration of Walsh series, we get the following result from [7]:

Theorem 1 *If $\underline{x}_1, \ldots, \underline{x}_N$ is a (t,m,s)-net to base b, then the integration error*

$$R_N(f) \leq c'(s,\alpha,c) \cdot b^{t(\alpha-1)} \cdot \frac{(\log N)^{s-1}}{N^{\alpha-1}} \quad ,$$

for all $f \in {}_b\bar{E}_s^{\alpha}(c)$.

2 Digital construction of nets

Niederreiter [9] introduced a general method for the construction of (t,m,s)-nets in base b, the so-called digital method. We will explain this method in a slightly simplified form:

Let the integers $m \geq 1$, $s \geq 1$, and $b \geq 2$ be given, and let $B = \{0, 1, \ldots, b-1\}$. We choose the following:

(i) a commutative ring $R = \{\overline{0}, \overline{1}, \ldots, \overline{b-1}\}$;

(ii) a bijection $\psi : B \to R$ with $\psi(a) = \overline{a}$ for all $a \in B$;

(iii) a bijection $\lambda : R \to B$ with $\lambda(\overline{a}) = a$ for all $\overline{a} \in R$;

(iv) elements $c_{jr}^{(i)} \in R$ for $1 \leq i \leq s$, $1 \leq j \leq m$, and $0 \leq r \leq m-1$.

For $0 \le n \le b^m - 1$, let

$$n = \sum_{r=0}^{m-1} a_r(n) b^r \ , \quad a_r(n) \in B$$

be the digit expansion of n in base b. With

$$x_{nj}^{(i)} = \lambda\left(\sum_{r=0}^{m-1} c_{jr}^{(i)} \cdot \psi(a_r(n))\right) = \left(\sum_{r=0}^{m-1} c_{jr}^{(i)} \cdot \overline{a}_r(n)\right) \quad \in B$$

for $0 \le n \le b^m - 1,\ \ 1 \le i \le s,\ \ 1 \le j \le m$, we put

$$x_n^{(i)} = \frac{1}{b^m}\left(\sum_{j=1}^{m} x_{nj}^{(i)} b^{m-j}\right) \quad \in [0,1) \quad \text{for } 0 \le n \le b^m - 1, \ \ 1 \le i \le s \ \ ,$$

and define the point set

$$\underline{x}_n = \left(x_n^{(1)}, \ldots, x_n^{(s)}\right) \quad \in [0,1)^s \quad \text{for } 0 \le n \le b^m - 1 \ \ .$$

If this point set is a (t,m,s)-net to base b, then it will be called a digital (t,m,s)-net.

Niederreiter [9] gives the following explicit examples of digital $(0,m,s)$-nets to any prime-power base b, provided that $s \le b+1$.

Construction method: Let $\mathrm{GF}(b)$ be the Galois field of order b. Then we define the elements $c_{jr}^{(i)} \in \mathrm{GF}(b),\ j = 1,\ldots,m,\ \ r = 0,\ldots,m-1,\ \ i = 1,\ldots,s$, in the following way:
Chose $s-1$ pairwise different elements β_i , $i = 1,\ldots,s-1$, from $\mathrm{GF}(b)$. Let

$$\begin{array}{lclllll}
c_{jr}^{(i)} & = & 0 & \text{for} \ \ 1 \le i \le s-1 & , & 2 \le j \le m & , \quad 0 \le r < j-1, \\
c_{jr}^{(i)} & = & \binom{r}{j-1} \cdot \beta_i^{r-j+1} & \text{for} \ \ 1 \le i \le s-1 & , & 1 \le j \le m & , \quad j-1 \le r \le m-1, \\
c_{jr}^{(s)} & = & \delta_{r,m-j} & \text{for} \ \ 1 \le j \le m & , & 0 \le r \le m-1 & .
\end{array}$$

Here, $\beta^0 = 1$ for all $\beta \in \mathrm{GF}(b)$, and δ is the Kronecker symbol. It follows that the point set corresponding to these matrices always is a $(0,m,s)$-net to base b [9].

Some figures of two-dimensional digital $(0,m,s)$-nets are shown in [12].
Two basic examples:

$$s = 2,\ b \ge 2 : C_1 = \begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & 0 & \\ & & & \ddots & \\ & 0 & & \ddots & \\ & & & & 1 \end{pmatrix} \ , \ \ C_2 = \begin{pmatrix} & & & & 1 \\ & 0 & & \cdot^{\cdot^{\cdot}} & \\ & & \cdot^{\cdot^{\cdot}} & & \\ & & \cdot^{\cdot^{\cdot}} & 0 & \\ & 1 & & & \\ 1 & & & & \end{pmatrix} \ .$$

This choice of the C_i provides the so-called Hammersley sequence to base b, which is well known in uniform distribution.

$$s = 3,\ b = 2 : C_2 = \begin{pmatrix} \binom{0}{0} & \binom{1}{0} & \cdot & \cdots & \binom{m-1}{0} \\ & \binom{1}{1} & \binom{2}{1} & \cdots & \binom{m-1}{1} \\ & & \cdot & & \vdots \\ & & & \cdot & \vdots \\ & 0 & & & \vdots \\ & & & & \binom{m-1}{m-1} \end{pmatrix} \quad \text{in } F_2 \ ,$$

with C_1 as above and C_3 equal to the second matrix from the first example gives a $(0, m, 3)$-net to base 2.

For these special nets we get the following improvement for Theorem 1 from [7]:

Theorem 2 *If b is prime, R the Galois field $GF(b)$ and $\underline{x}_1, \ldots, \underline{x}_N$ a digital (t, m, s)-net to prime base b defined over R, then the integration error*

$$R_N(f) \leq c'(s, \alpha, c) \cdot b^{t\alpha} \cdot \frac{(\log N)^{s-1}}{N^\alpha} \quad ,$$

for all $f \in {}_b\bar{E}_s^\alpha(c)$.

For the optimal case of digital nets with $t = 0$ we get

$$R_N(f) \leq c'(s, \alpha, c) \cdot \frac{(\log N)^{s-1}}{N^\alpha} \quad .$$

In some sense this is the best possible result that can be obtained in any case (see [7]).

Till now these results are proven for b prime only, but numerical investigations make us conjecture that these error-estimates also hold for prime-power base.

For most applications the most important case is the case of base $b = 2$. Therefore in the following we will concentrate on practical aspects for this case. As we have mentioned above we have a concrete construction method for digital $(0, m, s)$-nets to base 2 for dimension $s = 2$ or 3.

For dimension $s \geq 4$, however, it was shown by Niederreiter [9] that there cannot exist a $(0, m, s)$-net to base 2.

It need not be mentioned that methods for numerical integration are of increasing interest the higher the dimension is they work for. That is: We would need efficient construction methods for (t, m, s)-nets to base 2 with small t for higher dimensions.

3 Representing a function to different bases

So we have tried to do the following: We use $(0, m, s)$-nets to a base 2^h (with $h \in \mathbb{N}$; $h \geq 2$) for the integration of base-2 Walsh series.

That is: We have to integrate a base-2 Walsh series of dimension $s \geq 4$. We choose h minimal such that $2^h \geq s-1$. Then by Niederreiter's construction method we construct a digital $(0, m, s)$-net to base 2^h and use this net for the numerical integration of the function.

Then of course we need an error-estimate for the integration of base-2-functions by base-2^h-digital-nets. For that reason we use the following result, which gives a relation between the Walsh coefficients of a function represented in different bases.

Theorem 3 *For $h,\ s \in \mathbb{N},\ c > 0$ we have:*

a) *If $f \in {}_2\bar{E}_s^{\alpha}(c)$, then $f \in {}_{2^h}\bar{E}_s^{\alpha-\beta_h}(c \cdot 2^{hs\alpha})$, provided that $\alpha > 1 + \beta_h$. Here,*

$$\beta_h = \frac{h-1}{2h} + \frac{\sum_{k=0}^{h-2} \log \sin \left(\frac{\pi}{4} + \frac{\pi}{2} \left\{ \frac{4^{\left[\frac{k}{2}\right]}-1}{3 \cdot 2^{k+1}} \right\} \right)}{h \cdot \log 2} \quad ,$$

and β_h is monotonically increasing with

$$\beta := \lim_{h \to \infty} \beta_h = \frac{1}{2} + \frac{\log \sin \frac{5\pi}{12}}{\log 2} = 0.4499\ldots \quad .$$

b) *The "exponents" $\alpha - \beta_h$ in the above result are best possible.*

Remark. We have $\beta_2 = 0.25,\ \beta_3 = 0.2952\ldots,\ \beta_4 = 0.3394\ldots,\ \beta_5 = 0.3588\ldots.$

For the proof of Theorem 3 see [6] or [12].

From Theorem 1 and Theorem 3 the following result immediately follows:

Theorem 4 *If $\underline{x}_1, \ldots, \underline{x}_N$ is a $(0, m, s)$-net to base 2^h, then*

$$R_N(f) \leq c'(s, \alpha, c \cdot 2^{hs\alpha}) \cdot \frac{(\log N)^{s-1}}{N^{\alpha-\beta_h-1}} \quad ,$$

for all $f \in {}_2\bar{E}_s^{\alpha}(c)$ with $\alpha > 1 + \beta_h$.

Assuming again that Theorem 2 also is true for prime power base b we get the following conjecture:

Conjecture *If $\underline{x}_1, \ldots, \underline{x}_N$ is a digital $(0, m, s)$-net to base 2^h defined over the Galois field $GF(b)$, then*

$$R_N(f) \leq c'(s, \alpha, c \cdot 2^{hs\alpha}) \cdot \frac{(\log N)^{s-1}}{N^{\alpha - \beta_h}} \quad ,$$

for all $f \in {}_2\bar{E}_s^{\alpha}(c)$ with $\alpha > 1 + \beta_h$.

The integration error with this method is larger by a factor N^{β_h} at most. But on the other hand it provides a concrete efficient method to integrate base-2 Walsh series in arbitrary dimension with arbitrary large N and an integration error of at most $\sim \frac{(\log N)^{s-1}}{N^{\alpha-\beta_h-1}}$ and most probably of at most $\sim \frac{(\log N)^{s-1}}{N^{\alpha-\beta_h}}$, which still is a very useful order if α is not too small.

4 Construction of testing functions

As testing functions we use the following class of functions. (There are some advantages of these functions. So for example variables cannot be separated, and almost all Walsh coefficients $\hat{W}(k_0, \ldots, k_{s-1})$ of these functions f are of order $(\overline{k}_0 \cdots \overline{k}_{s-1})^{-\alpha}$ if $f \in {}_2\bar{E}_s^{\alpha}$. Furthermore their variations are not too small. Compare with [7] and [6].) For the construction of the testing function let $g_{\alpha'} : [0, 1) \longrightarrow I\!R$ with

$$g_{\alpha'} := \begin{cases} 0 & \text{if } x = 0 \\ \frac{1}{2^{(k-1)\alpha'}} & \text{if } \frac{1}{2^k} \leq x < \frac{1}{2^{k-1}} \ ; \quad k = 1, 2, \ldots \end{cases}$$

($\alpha' > \frac{\log 3}{\log 2} - 1$ arbitrary).

With

$$\begin{aligned} G_{\alpha'}(x_0, \ldots, x_{s-1}) &:= g_{\alpha'}(x_0) \cdots g_{\alpha'}(x_{s-1}), \text{ and} \\ h_{\alpha'}^{\gamma}(x_0, \ldots, x_{s-1}) &:= (g_{\alpha'}(x_0 \oplus \ldots \oplus x_{s-1}))^{\gamma} \end{aligned}$$

(where $\oplus$ means digital summation mod2) we take $f_{\alpha'} : [0, 1)^s \longrightarrow I\!R$ to be

$$f_{\alpha'}(x_0, \ldots, x_{s-1}) := G_{\alpha'}(x_0, \ldots, x_{s-1}) - h_{\alpha'}^{s + \frac{s-1}{\alpha'}}(x_0, \ldots, x_{s-1}) \quad .$$

Then $f_{\alpha'} \in {}_2\bar{E}_s^{\alpha'+1}(c)$ with a certain constant c (see [12]).

In the two-dimensional case some figures of the testing function for $\alpha' = 1$ are shown in [12].

We use these $f_{\alpha'}$ as testing functions and approximate the integral of $f_{\alpha'}$ on $[0,1)^s$ by $\frac{1}{N}\sum_{k=0}^{N-1} f_{\alpha'}(\underline{x}_k)$, where $\underline{x}_0,\ldots,\underline{x}_{N-1}$ form a digital $(0,m,s)$-net to a base b. We have

$$\int_{[0,1)^s} f_{\alpha'}(\underline{x})\,d\underline{x} = \left(\frac{2^{\alpha'}}{2^{\alpha'+1}-1}\right)^s - \left(\frac{2^{\alpha's+s-1}}{2^{\alpha's+s}-1}\right) \quad .$$

In Table 1 we show the integration error of our testing function for $\alpha' = 2.0$ and for $\alpha' = 3.0$ for bases 4 and 8 and dimensions 3 and 7.

Some exact results of the integral are:

for $\alpha' = 2.0$	and $s = 3$:	-0.3143895522984145...
for $\alpha' = 2.0$	and $s = 7$:	-0.4801057095355622...
for $\alpha' = 3.0$	and $s = 3$:	-0.3484183964183964...
for $\alpha' = 3.0$	and $s = 7$:	-0.4877258612133539...
for $\alpha' = 3.0$	and $s = 8$:	-0.4934537917701267...
for $\alpha' = 3.0$	and $s = 9$:	-0.4965086888892553...
for $\alpha' = 3.0$	and $s = 10$:	-0.4981379674041771...

5 Implementation

For using the program on different types of computers (portability) and for doing parallalization with minimal effort (see next section) the author had to decide between the programming languages Fortran and C. Experiences in structured programming (pascal, modula) caused us to implement the construction of $(0,m,s)$-nets and the numerical integration of Walsh series in the programming language C.

The digital construction of a $(0,m,s)$-net to base 2^h provides discrete points of the form $\frac{x'}{2^{hm}}$ with $x' \in \{0,\ldots,2^{hm}-1\}$. It is easy to see that we can write the above defined function $g_{\alpha'} : [0,1) \longrightarrow \mathbb{R}$ as

$$g_{\alpha'}\left(\frac{x'}{2^{hm}}\right) := \begin{cases} 0 & \text{for } x' = 0 \ , \\ 2^{\alpha'\cdot\left(\left\lfloor\frac{\log x'}{\log 2}\right\rfloor - hm + 1\right)} & \text{for } x' \neq 0 \ . \end{cases}$$

We have $\left\lfloor\frac{\log x'}{\log 2}\right\rfloor \in \{0,\ldots,hm-1\}$. If we initialize a global vector `g_hilf[i]` with

$$\texttt{g\_hilf}[\texttt{i}] = 2^{\alpha'\cdot(i-hm+1)} \text{ for all } i \in \{0,\ldots,hm-1\} \ ,$$

then we get:

$$g_{\alpha'}\left(\frac{x'}{2^{hm}}\right) := \begin{cases} 0 & \text{for } x' = 0\ , \\ \texttt{g\_hilf}\left[\left\lfloor \frac{\log x'}{\log 2}\right\rfloor\right] & \text{for } x' \neq 0\ . \end{cases}$$

For $x = \frac{x'}{2^{hm}}$ and $y = \frac{y'}{2^{hm}}$ with $x', y' \in \{0, \ldots, 2^{hm}-1\}$ we have

$$x \oplus y = \frac{x' \oplus y'}{2^{hm}} = \frac{x' \wedge y'}{2^{hm}}\quad ,$$

where $\oplus$ means digital summation mod 2 and $(\wedge)$ means the same for positive integers in the notation of the programming language C (logical XOR-operator).

Then we get

$$f_{\alpha'}\left(\frac{x'_0}{2^{hm}}, \ldots, \frac{x'_{s-1}}{2^{hm}}\right) = \prod_{i=0}^{s-1} g_{\alpha'}\left(\frac{x'_i}{2^{hm}}\right) - \left(g_{\alpha'}\left(\frac{x'_0 \wedge \ldots \wedge x'_{s-1}}{2^{hm}}\right)\right)^{s+\frac{s-1}{\alpha'}} \quad .$$

(For details see [12].)

A problem in high-dimensional numerical computation is the so-called floating-point-error. To avoid this error we only work with positive integers $\alpha' \geq 1$. Then all values of the testing function are nonnegative integer-exponents of $\frac{1}{2}$. Instead of the above mentioned vector `ghilf[i]` we initialize two vectors

- `expo[i]` $= -\alpha' \cdot (i - hm + 1)$, and
- `bit_expo[i]` $= -\alpha' \cdot (i - hm + 1)(s + \frac{s-1}{\alpha'})$.

(With $l(x') := \left\lfloor \frac{\log x'}{\log 2}\right\rfloor$, $x' \in \{0, \ldots, 2^{hm}-1\}$, we have $g_{\alpha'}\left(\frac{x'}{2^{hm}}\right) = \left(\frac{1}{2}\right)^{\texttt{expo[l(x')]}}$ and $\left(g_{\alpha'}\left(\frac{x'}{2^{hm}}\right)\right)^{s+\frac{s-1}{\alpha'}} = \left(\frac{1}{2}\right)^{\texttt{bit\_expo[l(x')]}}$.)

Additionally we initialize a global vector `anzahl[k]` $= 0.0$. For a positive integer $k \leq (hm-1)(\alpha' s + s - 1)$, `anzahl[k]` is the number of n, $0 \leq n \leq 2^{hm} - 1$, with $f(\underline{x}_n) = \left(\frac{1}{2}\right)^k$.

In the following we give the implementation of the testing function:

```
void f(Stuetzstelle x, int s)
      /* long Stuetzstelle[MAX_S] */
{
  int i, exponent, non_zero;
```

```
  long x_bit;

  x_bit = 0;
  non_zero = 1;               /* values 0 or 1; (0 for x' = 0) */
  exponent = 0;
  for(i=0;i<s;i++)
  {
    x_bit = x_bit ^ x[i];     /* XOR */
    if(x[i] != 0)             /* the exponents will be added */
      exponent+= expo[(int)floor(log((double)x[i])/log_2)];
    else
      non_zero = 0;
  };
  if (non_zero)               /* all x[i] are different from 0 */
    anzahl[exponent]++;

  if (x_bit != 0)             /* ... - h_alpha^() */
    anzahl[bit_expo[(int)floor(log((double)x_bit)/log_2)]]--;
}
```

So the result of the numerical integration is the following:

$$\frac{1}{2^{hm}} \sum_{i=0}^{\max} \texttt{anzahl[i]} \cdot \left(\frac{1}{2}\right)^i \quad .$$

Since the number of points is $2^{hm} \leq 2^{30}$, and $2^{30}\left(\frac{1}{2}\right)^{100} = \left(\frac{1}{2}\right)^{70} \leq 10^{-21}$, we can take $\max = 100$.

To avoid "expensive" operations we use a recursion ($\texttt{divis}_i := \left(\frac{1}{2}\right)^i$):

$$\begin{aligned} \texttt{divis}_0 &= 2^0 = 1 \ ; \\ \texttt{divis}_i &= \texttt{divis}_{i-1} + \texttt{divis}_{i-1} \ . \end{aligned}$$

We give a part of the main function:

```
  Eingabe(&h,&s,&m);
/* base 2^h; dim s; number of points 2^(hm) */
  Init(h,s,m,&basis,&N);
/* initializing all variables */
  Galois(h);
/* arrays with add. and mult. rules for GF(2^h) - */
/* - calculated with ``MATHEMATICA'' */
```

```
  Matrizen(m,s,basis);
/* calculating the elements C[j][r][i] */

  integral = 0.0;
  for(n=0; n<N; n++)          /* N=2^(hm) = number of points */
  {
    net(x,s,m,basis,h);       /* gives the n-th point of the net */
    f(x,s);                   /* exponents of (0.5) at this point */
  };
  divis = 1.0;
  for(i=0;i<=100;i++)
  {
    if (anzahl[i] != 0.0)
    {
      teilintegral = anzahl[i] / divis;
      /* anzahl[i] * (0.5)^i */
      integral+= teilintegral;
    };
    divis+= divis;            /* for the recursion  */
  };
  integral/= (double)N;       /* result of the integration */
```

The first calculations (see Table 1) were carried out on a Silicon Graphics Iris INDIGO with a MIPS R3000A RISC-Processor. If the dimension s is not too large already a relatively small number of net-points guarantees small integration errors.

For example for $s = 3$ (base $b = 4$) and $N = 65536$ we have a (sequential) run-time of 8.5 sec. Here (with $\alpha' = 3$) the integration error is $\approx 10^{-17}$. To get nearly the same error in dimension $s = 7$ we need ≈ 134 billion nodes (see Table 2) and a CPU time of more than 18 hours (Indigo).

Table 1: sequential calculations

$\alpha' = 2$, base $b = 4$, dimension $s = 3$		
N	numerical integration	integration error
16	0.1173629611730576	4.3175251347147203 E-01
64	-0.3178653717332622	3.4758194348477600 E-03
256	-0.3145006634294988	1.1111113108430715 E-04
1024	-0.3143905664620092	1.0141635947702454 E-06
4096	-0.3143896212968685	6.8998454061386383 E-08
16384	-0.3143895538144221	1.5160076527465094 E-09
65536	-0.3143895523228047	2.4390267583385139 E-11
262144	-0.3143895522990524	6.3787863879838369 E-13
$\alpha' = 3$, base $b = 4$, dimension $s = 3$		
N	numerical integration	integration error
16	0.0527399778075051	4.0115837422590150 E-01
64	-0.3495940715074610	1.1756750890645762 E-03
256	-0.3484276232265984	9.2268082019231201 E-06
1024	-0.3484184268146535	3.0396257066911403 E-08
4096	-0.3484183967716397	3.5324321245866486 E-10
16384	-0.3484183964203300	1.9335089085359414 E-12
65536	-0.3484183964184048	8.3266726846886741 E-15
262144	-0.3484183964183965	5.5511151231257827 E-17
$\alpha' = 2$, base $b = 8$, dimension $s = 7$		
N	numerical integration	integration error
64	0.0020300596961533	4.8213576923171553 E-01
512	-0.4612248946569917	1.8880814878570518 E-02
4096	-0.4808358634761971	7.3015394063491446 E-04
32768	-0.4801073128868772	1.6033513149937839 E-06
262144	-0.4801057174805712	7.9450089840449323 E-09
$\alpha' = 3$, base $b = 8$, dimension $s = 7$		
N	numerical integration	integration error
64	0.0001804840649129	4.8790634527826676 E-01
512	-0.4661444031680542	2.1581458045299706 E-02
4096	-0.4881680796822815	4.4221846892761318 E-04
32768	-0.4877259237250645	6.2511710585688718 E-08
262144	-0.4877258614504577	2.3710378105334939 E-10

6 Parallel calculations

In order to get small integration errors for even higher dimensions the sequential method is far too slow. We have to use parallel methods. But what kind of parallel methods?

There is one loop in the algorithm where "all" the time is spent:

```
for(n=0; n<N; n++)
{
  net(x,s,m,basis,h);
  f(x,s);
};
```

In the procedure `net(...)` the s coordinates of the n-th point of the net is computed, in `f(...)` our testfunction is evaluated at the just computed point. To get the result of the numerical integration we basicly have to sum up all the values of the testing function. For a fixed n these computations do not depend on the other net-points; this is the reason why it is "easy" to parallelize the program. We can divide the whole set of net-points in arbitrary small parts, compute the result of every part and then sum up the results.

We have implemented this concept in the programming language C, using the parallel programming environment PVM version 3.1. PVM (= Parallel Virtual Machine) is a software system that allows a network of heterogeneous Unix computers to be used as a single large parallel computer. Under PVM, a user-defined collection of serial, parallel, and vector computers appears as one large distributed-memory computer. One main processor partitions the problem, sends the initializing data to and receives the results from the "working" processors doing the calculations. For more information see PVM 3.0 User's Guide and Reference Manual [2]; the versions 3.0 and 3.1 don't differ for the user.

As our algorithm needs only few communication time PVM provides a very efficient solution. With increasing number of Workstations the running-time is nearly linearly falling. We used 9 DECstations 5000/33 with a R3000A RISC-Processor for getting the results of Table 2.

Table 2: parallel calculations

$\alpha' = 3$, base $b = 8$, dimension $s = 7$		
N	numerical integration	integration error
2097152	-0.4877258612135547	2.0078383400345956 E-13
16777216	-0.4877258612133638	9.9364960703951510 E-15
134217728	-0.4877258612133538	5.5511151231257827 E-17
$\alpha' = 3$, base $b = 8$, dimension $s = 8$		
N	numerical integration	integration error
2097152	-0.4934537912548443	5.1528231681530201 E-10
16777216	-0.4934537917700262	1.0041967257734541 E-13
134217728	-0.4934537917701263	3.3306690738754696 E-16
$\alpha' = 3$, base $b = 8$, dimension $s = 9$		
N	numerical integration	integration error
16777216	-0.0024416211936037	4.9895031008285906 E-01
134217728	-0.4965089451612083	2.5627195299193772 E-07
1073741824	-0.4965086889518218	6.2566452019297003 E-11
$\alpha' = 3$, base $b = \mathbf{16}$, dimension $s = 10$		
N	numerical integration	integration error
1048576	-0.4974980356357615	6.3993176841559141 E-04
16777216	-0.4974981565978168	6.3981080636027476 E-04
268435456	-0.4974986350031226	6.3933240105451539 E-04

7 Conclusion

The constant $c'(s, \alpha, c \cdot 2^{hs\alpha})$, which does not depend on N and therefore is not of a concerning theoretical interest, depends on the dimension and on the base. In order to get a small integration error it turns out that we need about $N = b^s$ points to compensate c'. As we have mentioned above for dimensions $s \geq 10$ we have to take base $b = 16$ and so we need at least $16^{10} \sim 10^{12}$ net-points. It's easy to see that even parallel methods cannot solve this in a reasonable amount of time.

This method for the numerical integration of Walsh series, with best theoretical results for all dimensions, is only practicable for dimensions up to $s = 9$, but in these cases with excellent results.

At time we are testing two other methods for getting suitable (t, m, s)-nets to a base 2 with small t ($t > 0$ for $s > 3$).

The first method (given in [10]), where irreducible polynomials over $\mathbb{Z}_2[x]$ are used for constructing the elements $c_{jr}^{(i)}$, provides $t = O(s \log s)$ for base 2, t independently of m. From [10], Table 2, we get the following values of t_s for a (t_s, m, s)-net to base 2:

s	2	3	4	5	6	7	8	9	10	11	12	13	14	15
t_s	0	0	1	3	5	8	11	14	18	22	26	30	34	38

The second method is in analogy to the classical theory of good lattice point sets and optimal coefficients (see [4] and [5]). Here, for every s and m, we are searching for a so-called optimal polynomial $\underline{g} \in (\mathbb{Z}_2[x])^s$ and use this polynomial for the construction of a $(t(s,m), m, s)$-net to base 2 (see [3]). In [3], there is given a table of optimal polynomials for $s = 4$ and $2 \leq m \leq 20$, and for $5 \leq s \leq 12$ and $2 \leq m \leq 10$. Of course, for most practical purposes this table is still too small. At the present, using a slightly different method, we are establishing a table of optimal polynomials for $3 \leq s \leq 15$ and $2 \leq m \leq 21$.

First tests up to a dimension $s = 15$ have shown that, also for higher dimensions, both methods provide small integration errors.

References

[1] N.S. Bakhvalov. Approximate computation of multiple integrals. *Vestnik Moskov. Univ. Ser. Mat. Mekh. Astr. Fiz. Khim.*, **4**:3–18, 1959. (Russian).

[2] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, and V. Sunderam. *PVM 3.0 User's Guide and Reference Manual.* Tennessee, 1993.

[3] T. Hansen, G.L. Mullen, and H. Niederreiter. Good parameters for a class of node sets in quasi-monte carlo integration. *Math. Comp.*, **61**:225–234, 1993.

[4] E. Hlawka. Zur angenäherten Berechnung mehrfacher Integrale. *Monatsh. Math.*, **66**:140–151, 1962.

[5] N.M. Korobov. *Number-Theoretic Methods in Approximate Analysis.* Fizmatgiz, Moscow, 1963. (Russian).

[6] G. Larcher, W.Ch. Schmid, and R. Wolf. Representation of functions as walsh series to different bases and an application to the numerical integration of high-dimensional walsh series. To appear in: Math. Comp.

[7] G. Larcher and C. Traunfellner. On the numerical integration of walsh-series by number-theoretic methods. To appear in: Math. Comp.

[8] K. Niederdrenk. *Die endliche Fourier- und Walsh-Transformation mit einer Einführung in die Bildverarbeitung.* Vieweg, 1988.

[9] H. Niederreiter. Point sets and sequences with small discrepancy. *Monatsh. Math.*, **104**:273–337, 1987.

[10] H. Niederreiter. Low-discrepancy and low-dispersion sequences. *J. Number Theory*, **30**:51–70, 1988.

[11] H. Niederreiter. Existence theorems for efficient lattice rules. In *Numerical Integration (Bergen, 1991)*, pages 71–80. Kluwer Acad. Publ., Dordrecht, 1992.

[12] W.Ch. Schmid. Zur numerischen Integration von Walshreihen, 1993. Diplomarbeit, Universität Salzburg.

[13] P. Zinterhof and P.jun. Zinterhof. Hyperbolic filtering of walsh series; a new method for data compression. Preprint.