

Help

```
/*
  Author: Syoiti Ninomiya
  Tokyo Institute of Technology
  Implementation of generalized Sobol sequences
  It gives very uniformly distribution even over several thousand dimensions
  */

#include <stdio.h>
#include <stdlib.h>
#include <
href../../common/math/cdo/cdo_math_h_src.pdfmath.h>
#include "
href../../common/optype_h_src.pdfoptype.h"

static int is_init = 1;
static int count;
static int DIG[1], DIM[1], SKIP[1];
static unsigned short P[1];
static int **g_m_b, *t_g_m_b, *ix_b;
static int btptn[8 * sizeof(int)];
static double two32;

extern char premia_data_dir[MAX_PATH_LEN];
extern char *path_sep;

void b2_g_sobol_seq(char *file_b, int d, double *x)
{
    FILE *in_fp;

    if (is_init)
    {
        size_t d_sz, read_sz;
        char data[MAX_PATH_LEN];
        sprintf(data, "%s%s%s", premia_data_dir, path_sep, file_b);

        in_fp = fopen(data, "rb");
        if (in_fp == NULL)
        {
            fprintf(stderr, "error in b2_g_sobol_seq(): cannot open file %s\ n",

```

```

        file_b);
    exit(1);
}
if (sizeof(DIG[0]) != sizeof(DIG[0])*fread(&(DIG[0]),
    sizeof(DIG[0]), 1, in_fp))
    goto error_1;
if (sizeof(DIM[0]) != sizeof(DIM[0])*fread(&(DIM[0]),
    sizeof(DIM[0]), 1, in_fp))
    goto error_1;
if (sizeof(P[0]) != sizeof(P[0])*fread(&(P[0]),
    sizeof(P[0]), 1, in_fp))
    goto error_1;
if (sizeof(SKIP[0]) != sizeof(SKIP[0])*fread(&(SKIP[0]),
    sizeof(SKIP[0]), 1, in_fp))
    goto error_1;
g_m_b = (int **)calloc(DIM[0], sizeof(int *));
t_g_m_b = (int **)calloc(DIM[0], sizeof(int *));
ix_b = (int *)calloc(DIM[0], sizeof(int));
{
    int i, j;

    d_sz = DIM[0] * DIG[0] * sizeof(int);
    for (read_sz = 0, i = 0; i < DIM[0]; i++)
    {
        g_m_b[i] = (int *)calloc(DIG[0], sizeof(int));
        for (j = 0; j < DIG[0]; j++)
            read_sz += fread(&(g_m_b[i][j]), sizeof(int), 1, in_fp);
    }
    if (read_sz * sizeof(int) != d_sz) goto error_1;
    d_sz = DIM[0] * sizeof(int);
    for (read_sz = 0, i = 0; i < DIM[0]; i++)
        read_sz += fread(&(ix_b[i]), sizeof(int), 1, in_fp);
    if (read_sz * sizeof(int) != d_sz) goto error_1;
}
fclose(in_fp);
{
    int t;
    size_t i;
    for (t = 0x1, i = 0; i < 8 * sizeof(int); i++, t <= 1) btptn[i] = t;
}
{

```

```

    /** t_g_m_b is the transpose of g_m_b */
    int i, j, k;
    for (i = 0; i < DIM[0]; i++)
    {
        int bidx;

        t_g_m_b[i] = (int *)calloc(DIG[0], sizeof(int));
        for (k = 0, t_g_m_b[i][k] = 0x0; k < DIG[0]; k++)
            for (bidx = 0x1, j = 0; j < DIG[0]; j++, bidx <= 1)
                t_g_m_b[i][k] |= (btptn[k] & g_m_b[i][j]) ? bidx : 0;
    } /** for (i) */
}
{
    int i;
    for (i = 0; i < DIM[0]; i++) free(g_m_b[i]);
    free(g_m_b);
}
count = SKIP[0];
two32 = pow(2.0, 32.0);
is_init = 0;
} /** if (is_init) */
{
    int i, k;
    unsigned long tp;
    unsigned long digit;

    tp = P[0];
    if (d > DIM[0])
    {
        fprintf(stderr, "error in b2_g_sobol_seq():");
        fprintf(stderr, "\ nspecified dimension is greater than tables's\
dimension.\ n");
        exit(1);
    }
    digit = count;
    k = 0;
    while (digit % tp == tp - 1)
    {
        k++;
        digit /= tp;
    }
}

```

```

    for (i = 0; i < d; i++)
    {
        int j, bidx;

        ix_b[i] ^= t_g_m_b[i][k];
        for (x[i] = 0.0, bidx = ix_b[i], j = 0; j < DIG[0]; j++, bidx >>= 1)
            x[i] = 2.0 * x[i] + (bidx & 0x1);
        x[i] /= two32;
    } /** for (i) **/
    count++;
}

return;
error_1:
    fprintf(stderr, "error in b2_g_sobol_seq(): failed in fread().\ n");
    exit(1);
}

void b2_g_sobol_free()
{
    int i;
    for (i = 0; i < DIM[0]; i++) free(t_g_m_b[i]);
    free(t_g_m_b);
    free(ix_b);

    /* the generator is reset after this call */
    is_init = 1;
}

```