Свободные библиотеки в вычислительных задачах

Демин П.,Алексеев Е.Р.
Киров, Вятский Государственный университет, кафедра прикладной математики и информатики

evgeniyalekseev.wordpress.com, er.alekseev@yandex.ru

Проблемы вычислительных задач

- Разработка и реализация высокоэффективных алгоритмов вычислительной математики;
- достижение достаточной точности вычислений;
- кросс-платформная реализация графического вывода результатов.

Научная библиотека GSL

http://www.gnu.org/software/gsl/

http://chpc.ru/cat/образовательный-семинар

- Некоторые возможности линейной алгебры (арифм. операции, поэлементное произв., LU- и QR-разложение, нахождение собственных чисел и векторов);
- задачи локальной оптимизации (одномерная и многомерная оптимизация);
- задачи статистики (генерация и работа со случайными и квази-случайными последовательностями, нормальное распределение, распределение Пуассона и др.);

Научная библиотека GSL

- Задачи численного интегрирования (в основе метод Гаусса-Кронрода с различным количеством точек, алгоритм исключения особых точек, вычисление интеграла Фурье и др.);
- задачи с комплексными переменными (арифметические операции);
- сортировки.

GSL: BLAS

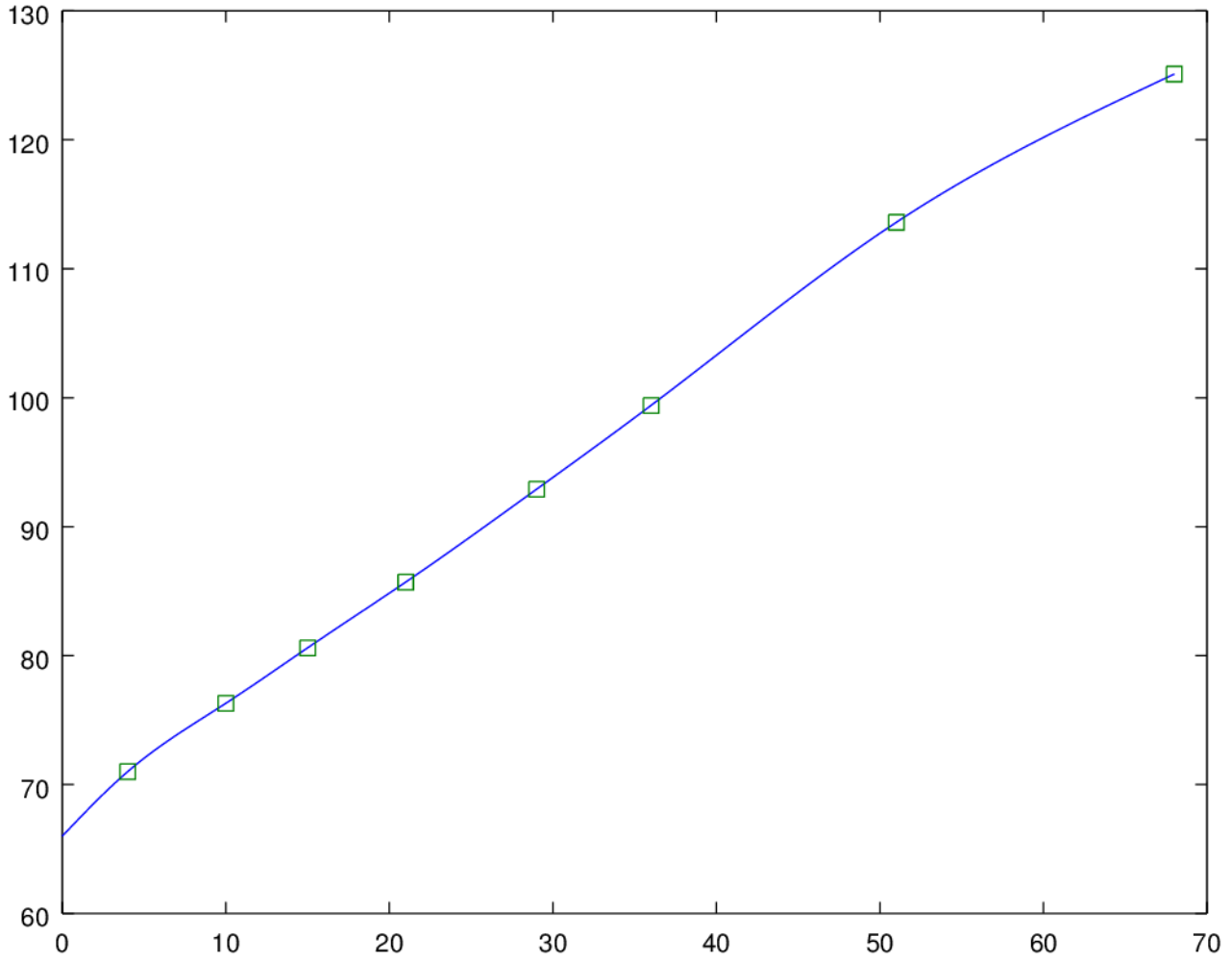
BLAS — Basic Linear Algebra Subprograms

- Level_1(скалярное произведение, норма, присваивание, сумма векторов).
- Level_2 (умножение матрицы на вектор).
- Level_3 (умножение матрицы на матрицу).

Пример: интерполяция в GSL

```
 //особый вид итератора для интерполяции, отслеживающий текущее состояние поиска
    gsl_interp_accel *acc = gsl_interp_accel_alloc ();
   //выбор типа сплайна - кубический
    gsl_spline *spline = gsl_spline_alloc (gsl_interp_cspline, 9);
//инициализация сплайнов, копирование данных
    gsl_spline_init (spline, x, y, 9);
    for (xi = x[0]; xi < x[8]; xi += 0.01)
      {
        yi = gsl_spline_eval (spline, xi, acc);
        fprintf(myfile,"%g %g\n", xi, yi);
      }
    gsl_spline_free (spline);
    gsl_interp_accel_free (acc);
```

Результат



Умножение матриц

Размерность	gsl	Классический алгоритм	Классика с (с транспонированием b)	Алгоритм блочного умножения	Алгоритм Штрассена
64	0,000186	0,000130	0,000207	0,000254	0,000278
128	0,001366	0,001200	0.001936	0,002287	0,002103
256	0,011349	0,014221	0.014949	0,018372	0,015308
512	0,084804	0,113403	0.113165	0,138497	0,109818
1024	1,071467	2.805108	1.066807	1.111738	0.756480
2048	12,90892	43.11276	8.62358	8.83695	5.39590

Описание алгоритма Штрассена

$$A=\begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} B=\begin{bmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & B_{2,2} \end{bmatrix} C=\begin{bmatrix} C_{1,1} & C_{1,2} \\ C_{2,1} & C_{2,2} \end{bmatrix}$$

$$P_1=(A_{1,1}+A_{2,2})(B_{1,1}+B_{2,2})$$
$$P_2=(A_{2,1}+A_{2,2})B_{1,1}$$
$$P_3=A_{1,1}(B_{1,2}-B_{2,2})$$
$$P_4=A_{2,2}(B_{1,2}-B_{2,2})$$
$$P_5=(A_{1,1}+A_{1,2})B_{2,2}$$
$$P_6=(A_{2,1}-A_{1,1})(B_{1,1}+B_{1,2})$$
$$P_7=(A_{1,2}-A_{2,2})(B_{2,1}+B_{2,2})$$

$$C_{1,1}=P_1+P_4-P_5+P_7$$
$$C_{1,2}=P_3+P_5$$
$$C_{2,1}=P_2+P_4$$
$$C_{2,2}=P_1-P_2+P_3+P_6$$

Что в GSL отсутствует

- Методы обращения матриц.
- Методы вычисления определителя.
- Графический вывод результатов.
- Возможность увеличить точность вычислений.

Точность вычислений

- Библиотека большой точности MPFR языков C/C++ (http://www.mpfr.org/).
- Библиотека интервального анализа XSC языка C++ (http://www2.math.uni-wuppertal.de/~xsc/xsc/cxsc_new.html).

Функция Румпа

$$f = 333.75\,b^6 + a^2\left(11\,a^2 b^2 - b^6 - 121\,b^4 - 2\right) + 5.5\,b^8 + a/\left(2\,b\right)$$

где a=77617, b=33096.

Ожидаемое значение:

f = - 0.827396059946821368141165095479816…

- Возведение в степень (b^8 - число 36 порядка).
- Арифметические операции с числами разных порядков.

Тестирование

Ожидаемое значение:

f = - 0.827396059946821368141165095479816…

C/C++ double x = -1.18059e+21

C/C++ long double x = 5.76461e+17

MPFR: x = -0.827396059946821368141165095479816291999060331234781169259215 (77 знаков или 256 бит)

C-XSC interval x = [-8.264142E+021,5.902959E+021]

C-XSC l_interval 32 знака

X = [- 0.827396059947204776108264923095703125000000000, -0.827396059939928818494081497192382812500000000]

C-XSC l_real x = -0.827396059946821305075559394026640802621841431

Пример использования MPFR

```
….
mpfr_pow(t1,y,k6,MPFR_RNDN);
//t1=b^6
mpfr_mul(t2,k1,t1,MPFR_RNDN);
//t2=333.75b^6
mpfr_set_d(k6,8.0,MPFR_RNDN);
//k6=8
mpfr_pow(t1,y,k6,MPFR_RNDN);
//t1=b^8
mpfr_mul(t3,k5,t1,MPFR_RNDN);
//t3=5.5b^8
mpfr_add(t1,t2,t3,MPFR_RNDN);
//t1=333.75b^6+5.5b^8
….
```

MPFR C++

http://www.holoborodko.com/pavel/mpfr/

….

```
mpreal::set_default_prec(256);
mpreal k5;
k5=5.5
sum+=k1*pow(b,k6);
K6=8.0;
sum+=k5*pow(b,k6);
```

….

Скалярное умножение векторов

$$x=\left(10^{20},1223,10^{18},10^{15},3,-10^{12}\right),\ y=\left(10^{20},2,-10^{22},10^{13},2111,10^{16}\right)$$

Ожидаемое значение: 8779

- Вычисление 10^{40}.
- Арифметические операции с числами разных порядков.

Тестирование

C/C++ double SUM = -2.3438e+23

C/C++ long double SUM = -3.24155e+19

MPFR: x = 8779 (256 бит/77 знаков)

C-XSC использование rvector и real

SUM = 0.000000

C-XSC использование l_vector + l_real

SUM = 8779.000 (32 знака)

Матрица Бутройда-Деккера

$$a_{ij} = \binom{n+i-1}{i-1}\binom{n-1}{n-j}\frac{n}{i+j-1}\, i, j = 1,\ldots,n$$

Определитель

C/C++ double: 1.00000;

C/C++ long double: 1.00000;

C-XSC

interval: [-3.204219E+002,1.877296E+002]

l_interval: [0.9999999999,   1.0000000001] 32 знака

MPFR C++: 1.00000. 38 знаков или 128 бит

Решение СЛАУ Ax=b 10x10

При правой части b=(1,2,3,….,10) ожидается решение x=(0,1,-2,….,9)

GSL	MPFR C++(128 бит/38 знаков)
9.84104e-09 1 -2 3 -3.99999 4.99998 -5.99996 6.9999 -7.9998 8.9996	1.4868114007e-30 1 -2 3 -4 5 -6 7 -8 9

Матрица Гильберта

$$a_{ij} = \frac{1}{i + j - 1}$$

$$A = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{bmatrix}$$

Решение СЛАУ Ax=b 12x12

Элемент b_i вектора b равен сумме элементов i-й строки матрицы A.

Ожидаемое : (1,1,1,1,1,1,1,1,1,1,1,1)

GSL	MPFR C++ (128 бит/38 знаков)	MPFR C++ (64 бит/19 знаков)
1	1	0.99999999995155
1	1	1.0000000062518
0.999862	1	0.999999800577111
1.00188	1	1.00000274853804
0.986242	1	0.999979658988887
1.06038	1	1.00009008596554
0.831939	1	0.999747301888991
1.30397	1	1.00046006454399
0.643862	1	0.99945792941662
1.26068	1	1.00039873868678
0.891667	1	0.999833577857578
1.01951	1	1.00003008737121

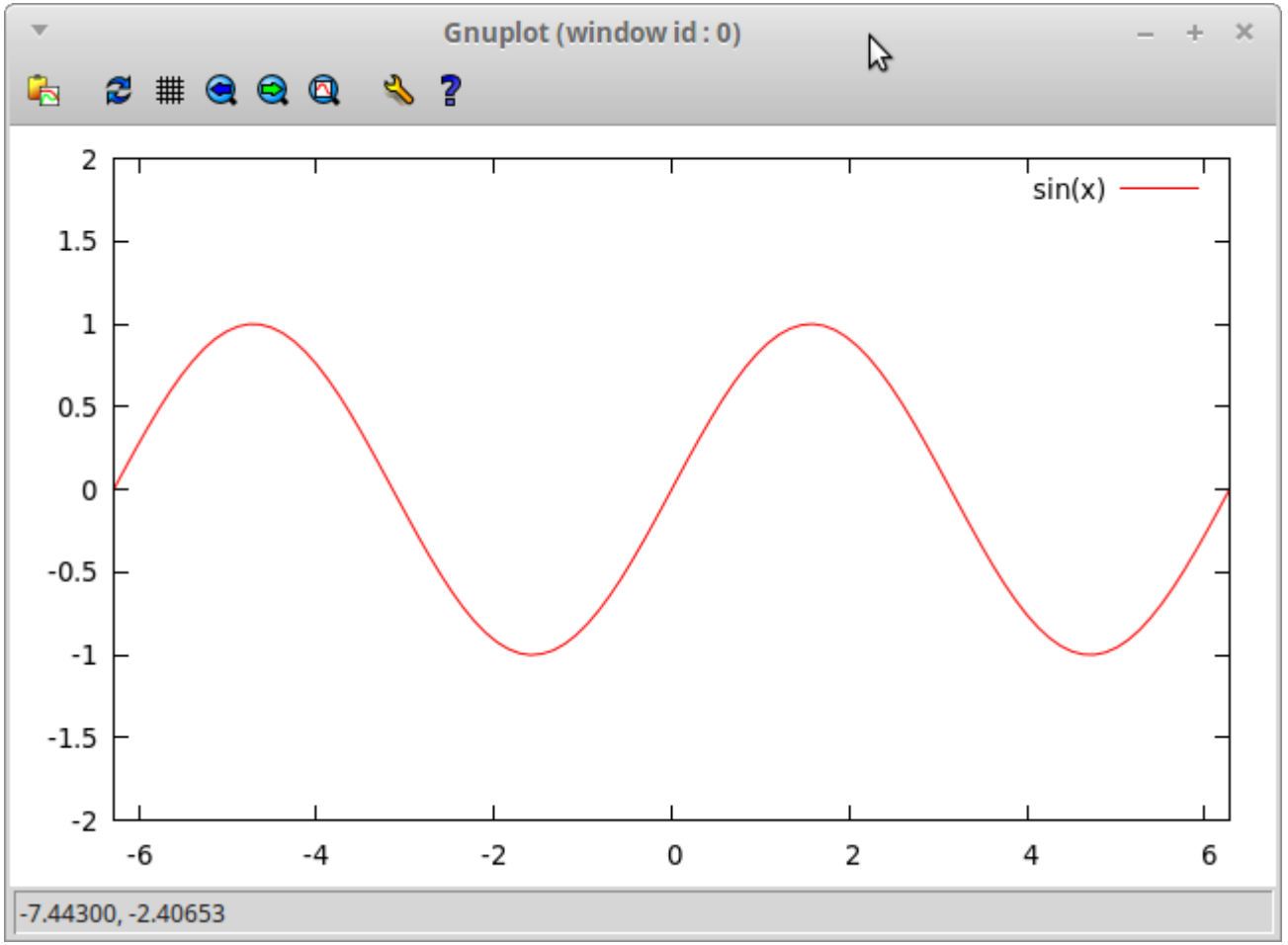
Вывод графиков

- gnuplot-cpp (https://code.google.com/p/gnuplot-cpp);
- gnufor2 (http://www.math.yorku.ca/~akuznets/gnufor2/);
- Встроенные средства языка.

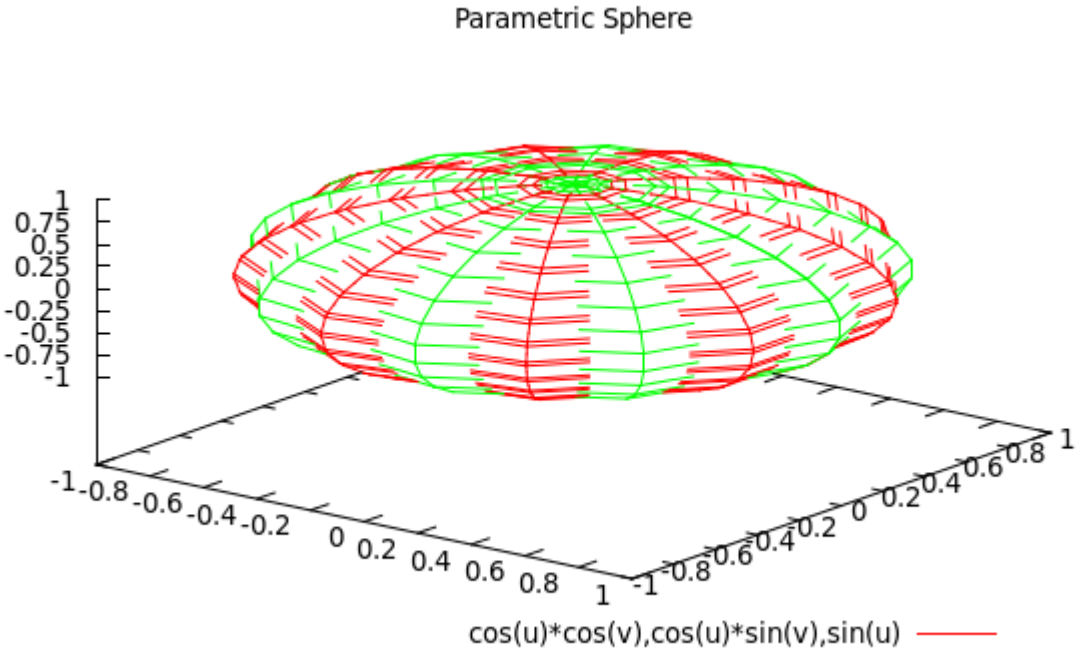
Использование встроенных возможностей языка

```
//создание процесса
FILE *gpipe = popen(GNUPLOT_NAME, "w");
//посылаем команды открытому потоку
fprintf(gpipe,"set xrange [-20*pi:20*pi]\n");
fprintf(gpipe,"set yrange [-2:2]\n");
fprintf(gpipe,"plot sin(x)\n");
//выходим из GNUPlot и закрываем поток
fprintf(gpipe, "exit\n");
pclose(gpipe);
```

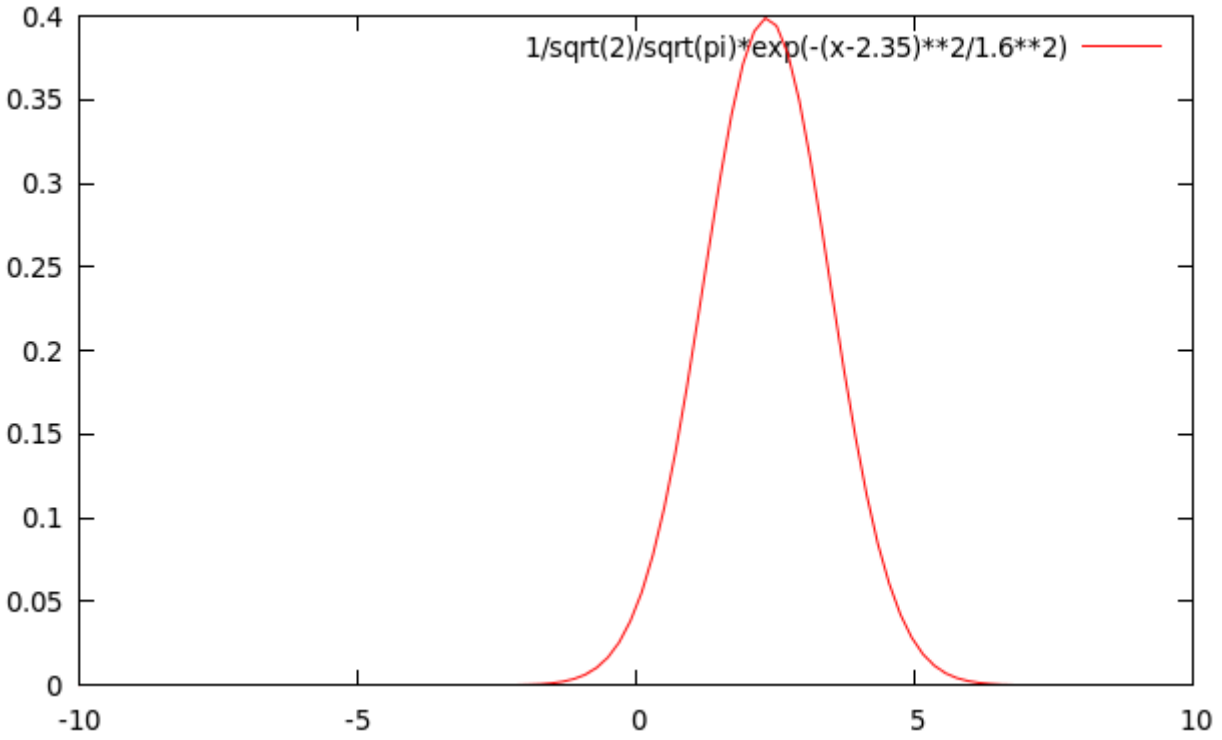
График sin(x)


Gnuplot (window id : 0)
sin(x)
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-6
-4
-2
0
2
4
6
-7.44300, -2.40653

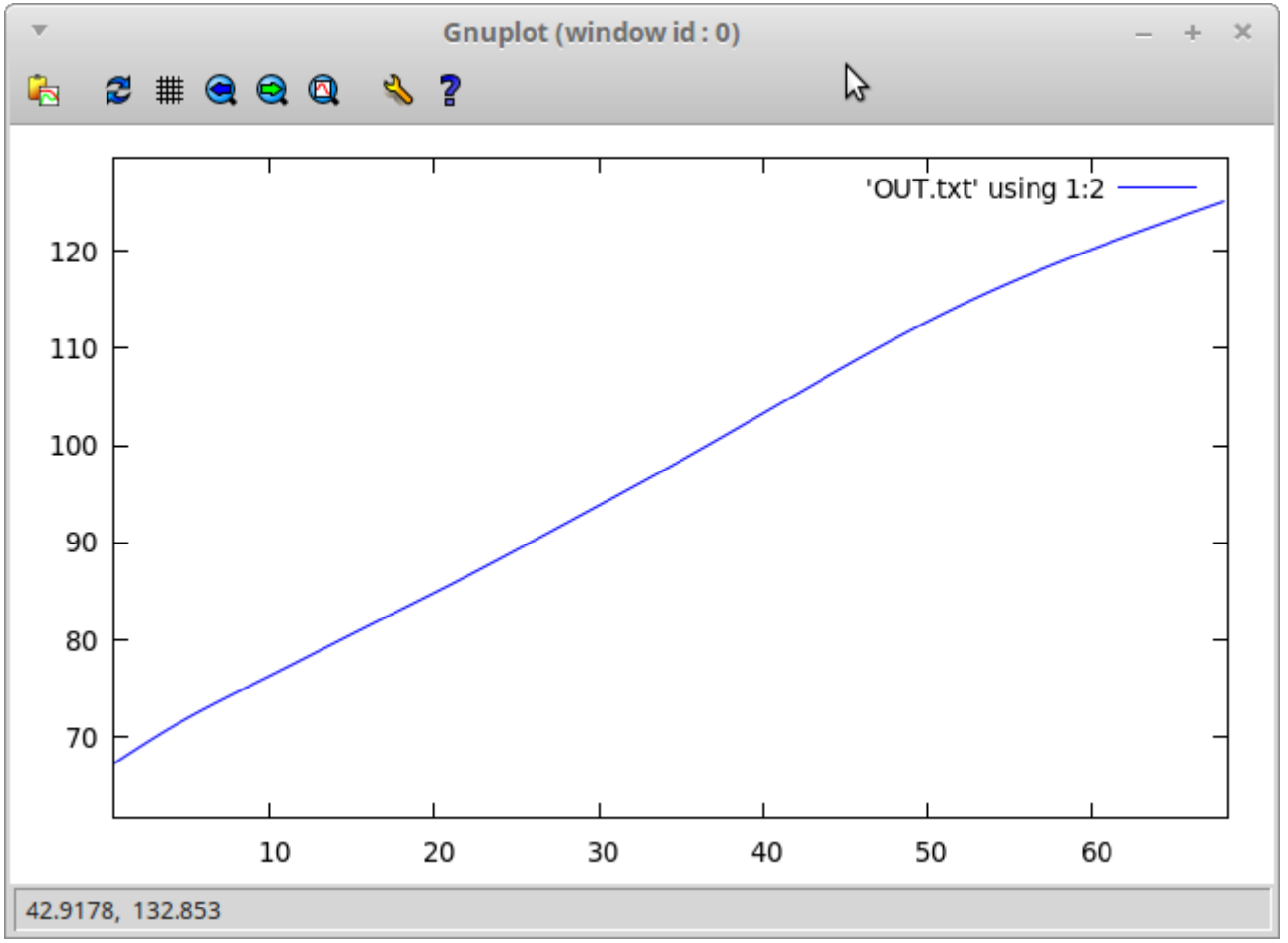
Сфера


Parametric Sphere
1
0.75
0.5
0.25
0
-0.25
-0.5
-0.75
-1
-1 -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 1
-1 -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 1
cos(u)*cos(v),cos(u)*sin(v),sin(u)

Нормальное распределение



Интерполяция



Спасибо за внимание

Готов ответить на Ваши вопросы.