ОБ ОДНОЙ МОДЕЛИ ГОСУДАРСТВЕННОГО УПРАВЛЕНИЯ ФИНАНСОВЫМ РЫНКОМ

А.Г. Исламов, Г.Г. Исламов

Удмуртский госуниверситет

Финансовый рынок служит важной, но не определяющей, составляющей экономики государства. Заложенные исторически правила его функционирования не являются совершенными, что показывает затянувшийся мировой финансовый кризис. Обладая ограниченным в объёме универсальным ресурсом, выражающим всеобщий эквивалент труда, финансовый рынок искусственно создаёт предпосылки для безграничного печатания денег как необходимого условия своего существования. Это приводит к хронической инфляции, последствия которой оказываются катастрофическими как для населения государства, так и для отдельных отраслей экономики. Сказанное не относится к исламской экономической технологии ведения банковского дела и методов финансирования [1].

1. ***Исламская экономическая модель и современность / Р.И. Беккин. – М.:ИД Марджани, 2009. – 344 с.***

Рассмотренная авторами в работе [2]

2. ***Исламов А.Г., Исламов Г.Г., Лукин О Л. Об одной модели финансового рынка. Режим доступа: http://www.iton.mfua.ru/tesis/islamov2.doc.***

математическая модель финансового рынка показывает, что финансовый рынок может находиться в одном из трёх состояний: *деградация, кризис и стабильность.* Это следствие иерархической структуры финансового рынка и закона его эволюции, которые задаются матрицей P процентов взаимных платежей участников финансового рынка и столбцом k недельных гонораров.

Имеется n участников финансового рынка, образующих иерархическую структуру с выделенным корнем. Единственный путь, ведущий из этого корня к любой другой вершине, определяет полное имя участника этого рынка. Каждый из участников финансового рынка обладает уникальной способностью увеличивать за указанный промежуток времени (неделя) имеющийся у него капитал на определенное число процентов, не зависящее от объема капитала и порядкового номера недели, но своё для каждого участника. В конце любой недели каждый родитель перечисляет на счет своего дочернего участника определенный процент (для каждого дочернего участника свой) от той суммы, которой родитель располагал в начале недели. В это же время все участники финансового рынка (включая и корневого) перечисляют на счёт корневого участника сумму, которая была у них на начало недели, и весь доход, который был получен в течение недели от использования этой суммы. В конце недели каждый из участников получает определенный (свой) процент от той общей суммы, которой

располагали все n участников в начале недели, в качестве «недельного» гонорара. Обозначим через $x(t) = colon(x_1(t),...,x_n(t))$ вектор-столбец, описывающий распределение капитала по участникам финансового рынка в начале недели с порядковым номером $t = 0,1,...,54$. Тогда на начало недели с порядковым номером $t+1$ (конец недели с порядковым номером t) имеем следующее распределение капитала: $x(t+1) = (P-K)x(t)$. Здесь P - матрица, содержащая проценты перевода денежного капитала среди участников финансового рынка, $K = colon(k_1,...,k_n)*(1,...,1)$ - одноранговая матрица, содержащая проценты недельного «гонорара» этих участников.

Ниже приводятся результаты численного эксперимента в пакете Mathematica для случая трёх участников, образующих линейную структуру (финансовое звено). Из приводимого ниже фрагмента программы видно, что корневой участник (под номером один) обладает способностью увеличивать имеющийся денежный капитал на 18%, средний участник (под номером два) – на 12%, третий – всего на 0.1%. Проценты недельных гонораров этих участников составляют, соответственно, 1%, 1% и 0.1%. Обозначим $S = P - K$.

Фрагмент программы 1:

```
a = {1.1,1.02,1.001},  b = {0.08, 0, 0},  c = {0,0.1,0},
S = {a,b,c} - {{0.01},{0.01},{0.001}}.{{1,1,1}},  Q = MatrixPower[S,54],
Q.{1,1,1},  {1,1,1}.%,  u = 1.17^54,  v = 1.11^54,  w = 1.^52,  u + v + w,
```

Результаты работы программы для единичного стартового капитала :

```
{6270.93, 376.60, 26.83},  6674.36,
4808.8,  280.182,  1.,  5089.98.
```

В последней строке находятся суммы, которые могут получить участники, если они будут действовать поодиночке. В предпоследней строке находятся выигрыши, полученные в результате кооперации в течение 54 недель. Как видим, совместное использование денежного капитала дает выигрыш.

В силу иерархической структуры возможна такая нумерация участников финансового рынка, при которой номер дочернего участника больше номера родителя. Мы предполагаем, что такая нумерация уже имеет место. В таком случае матрица P, содержащая проценты перевода денежного капитала среди участников рынка представима в виде суммы нижнетреугольной матрицы с нулями на основной диагонали и одноранговой матрицы. Можно показать, что каждое ненулевое собственное значение матрицы P имеет геометрическую кратность, равную единице. В силу теоремы двойственности работы [3]

3. ***Исламов Г.Г. Об управлении спектром динамической системы // Дифференциальные уравнения. – 1987. – Т.23, № 8. – С. 1299-1302.***

за счёт выбора одноранговой матрицы K всегда можно получить наперёд заданный спектр возмущённой матрицы $P-K$ без учёта нуля, и, тем самым, обеспечить заданный темп роста (убывания) капитала каждого из участников финансового рынка. Таким образом, мы видим, что рассмотренная модель финансового рынка приводит к задаче модального управления, в которой требуется построить вектор обратной связи $k = colon(k_1,...,k_n)$, содержащий проценты недельного гонорара участников рынка и обеспечивающий заданный темп роста (убывания) капитала. Из доказательства теоремы двойственности работы [3] видно, что существует эффективный алгоритм построения такого вектора обратной связи k. При этом построении используется информация о спектрах матриц P и $P-K$, фробениусова (первая естественная) форма матрицы P и соответствующая ей преобразующая матрица T. В последующей работе [4]

4. ***Исламов Г.Г. Свойства одноранговых возмущений // Известия вузов. Математика. – 1989. - № 4. – С. 29-35.***

приведена схема построения одноранговых возмущений ab^T для $n \times n$-матрицы P с простым спектром $\lambda_1,...,\lambda_n$, при которых возмущённая матрица $Q = P - ab^T$ имеет наперёд заданный спектр $\mu_1,...,\mu_n$. Если $P = \sum_{i=1}^{n} \lambda_i u_i v_i^T$ есть спектральное разложение матрицы P взаимных платежей, то вектора a и b таких возмущений можно взять в виде $a = \sum_{i=1}^{n} \alpha_i u_i$ и $b = \sum_{i=1}^{n} \bar{\beta}_i v_i^T$. Здесь черта сверху обозначает комплексное сопряжение, а все величины α_i и β_i можно брать любыми скалярами, отличными от нуля и связанными условием $\alpha_i \beta_i = \prod_{j=1}^{n} (\lambda_i - \mu_j) / \prod_{\substack{j=1 \\ j \neq i}}^{n} (\lambda_i - \lambda_j)$.

В качестве второго примера рассмотрим систему из 8 банков, образующих иерархию, задаваемую строкой $(2,6,8;\ 3,5;\ 4;\ ;\ ;7;\ ;\ ;)$. Поясним, что $2,6,8$ - это номера дочерних банков корневого банка под номером 1. Далее, после точки с запятой в строке идут номера $3,5$ дочерних банков родительского банка под номером 2 и так далее. Из указанной строки видно, что концевые вершины дерева иерархии образуют банки с номерами $4,5,7,8$. Соответствующие этой иерархии проценты отчислений родительских банков дочерним банкам имеют вид: $(\alpha_1, \alpha_2, \alpha_3;\ \alpha_4, \alpha_5;\ \alpha_6;\ ;\ ;\alpha_7;\ ;\ ;)$. Отчисления участников финансового рынка корневому банку заданы строкой $\beta = (\beta_1, \beta_2, ..., \beta_8)$. Матрица P рассматриваемого примера финансового рынка из восьми банков имеет вид:

$$P=\begin{pmatrix} \beta_1 & \beta_2 & \beta_3 & \beta_4 & \beta_5 & \beta_6 & \beta_7 & \beta_8 \\ \alpha_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \alpha_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \alpha_6 & 0 & 0 & 0 & 0 & 0 \\ 0 & \alpha_5 & 0 & 0 & 0 & 0 & 0 & 0 \\ \alpha_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \alpha_7 & 0 & 0 \\ \alpha_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Пусть первая строка этой матрицы, содержащая проценты отчислений корневому банку, равна $(1.11, 1.12, 1.13, 1.14, 1.12, 1.13, 1.11, 1.2)$, а проценты отчислений дочерним банкам задаются набором $(0.01, 0.02, 0.03, 0.02, 0.03, 0.02, 0.01)$. В случае отсутствия обратной связи (вектор k - нулевой), имеем следующие результаты работы приводимого ниже фрагмента программы для единичного стартового капитала и 54 недель:

$$(36299, 310.19, 5.30, 0.09, 7.95, 620.38, 5.30, 930.57).$$

Если же разрешить каждому участнику 10% долю потребления в конце недели от общей суммы всех участников в начале недели, то при тех же стартовых условиях через 9 недель будем иметь нулевое распределение капитала среди участников, то есть банковская система обанкротилась. С другой стороны, если каждый участник не потребляет, а получает от «спонсоров извне» всего лишь 0.1% от указанной выше суммы, то через 54 недели для единичного стартового капитала будем иметь следующий результат:

$$(51821.70,\ 486.41,\ 54.79,\ 47.46,\ 58.92,\ 926.29,\ 54.39,\ 1366.18).$$

Фрагмент программы 2:

```
k = {{k1}, {k2}, {k3}, {k4}, {k5}, {k6}, {k7}, {k8}} =
   {-{0.001}, {-0.001}, {-0.001}, {-0.001}, {-0.001}, {-0.001},
    {-0.001}, {-0.001}};
e = {{1, 1, 1, 1, 1, 1, 1, 1}};
A = {α1, α2, α3, α4, α5, α6, α7} =
   {0.01, 0.02, 0.03, 0.02, 0.03, 0.02, 0.01};
P1 = {β1, β2, β3, β4, β5, β6, β7, β8} =
   {1.11, 1.12, 1.13, 1.14, 1.12, 1.13, 1.11, 1.2};
P2 = {α1, 0, 0, 0, 0, 0, 0, 0};
P3 = {0, α4, 0, 0, 0, 0, 0, 0};
P4 = {0, 0, α6, 0, 0, 0, 0, 0};
P5 = {0, α5, 0, 0, 0, 0, 0, 0};
P6 = {α2, 0, 0, 0, 0, 0, 0, 0};
P7 = {0, 0, 0, 0, 0, α7, 0, 0};
P8 = {α3, 0, 0, 0, 0, 0, 0, 0};
P = {P1, P2, P3, P4, P5, P6, P7, P8};
S = P - k.e;
MatrixForm[%]
Eigenvalues[S] // N;
Q = MatrixPower[S, 54];
Q.{1, 1, 1, 1, 1, 1, 1, 1}
```

Принятая в модели иерархия финансового рынка обеспечивает его мобильность и управляемость. Однако эти положительные качества могут быть использованы в разрушительных целях, придавая финансовому рынку уязвимость при неблагоприятных внешних воздействиях: интервенция, вывоз капитала, и т. д.

Известно, что финансовый рынок обязан стимулировать инновации в сфере производства материальных благ и воспроизводства интеллектуальных ресурсов, финансировать планомерное развитие потребностей людей и бесперебойное потребление материальных и информационных ресурсов. Это значит, что требуется эффективное управление финансовым рынком со стороны государства. Рассмотренное нами уточнение описанной выше модели содержит внешнее управление финансовым рынком со стороны государства и описывается дискретным управляемым процессом $x[t+1]=(P-K)x[t]+u[t],\ t=0,1,2,...,$ где управление со стороны государства формируется по правилу $u[t]=p\cdot qx[t]$. Здесь $p=colon(p_1,...,p_n)$ есть столбец гонораров и штрафов, установленных государством. Положительная компонента p_i выражает величину гонорара в процентах $i-$го участника финансового рынка, тогда как абсолютное значение отрицательной компоненты p_i определяет величину налога, которым облагается участник i в конце планового периода k. Строка $q=(q_1,...,q_n)$ содержит

государственные приоритеты, а столбец $x[t] = colon(x_1[t],...,x_n[t])$ - капиталы участников рынка.

Предлагаемая здесь математическая модель государственного управления финансовым рынком показывает, что финансовый рынок также может находиться в одном из указанных выше трёх состояний: *деградация, кризис и стабильность. Первое* состояние свидетельствует о крайне неэффективном управлении и отвечает процессу разорения финансового рынка. *Второе* состояние связано с необходимостью включения печатного станка для обеспечения экспоненциального роста денежной массы в зависимости от номера временного периода. *Третье* состояние соответствует степенному росту денежной массы. Мы показываем, что за счёт выбора управляющих векторов p и q можно обеспечить любое из указанных выше трёх состояний финансового рынка.

Для матриц с простым спектром предлагается высокоэффективный алгоритм построения одноранговых возмущений, которые переводят их в матрицы с наперёд заданным спектром. На практике построение указанных векторов p и q мы производим следующим образом. Пусть $A = P - K$, $S = (A^{n-1}f, A^{n-2}f,..., Af, f)$ - матрица базиса Крылова с циклическим вектором f длины n, $g = A^n f$. Тогда столбец $h = -S^{-1}g$ даёт нам коэффициенты характеристического многочлена матрицы A. Добавим, справа, к противоположному столбцу (- h) единичные столбцы так, чтобы получилась матрица Фробениуса F характеристического многочлена матрицы A. Непосредственно проверяется справедливость факторизации $A = SFS^{-1}$ матрицы динамической системы через эту матрицу Фробениуса. По заданному многочлену $\prod_{j=1}^{n}(\lambda - \mu_j)$ возмущённой матрицы $B = A + p \cdot q$ построим матрицу Фробениуса $\tilde{F}$ такой же структуры, что и матрица F. Из очевидного разложения $A = S\tilde{F}S^{-1} + S(F - \tilde{F})S^{-1}$ находим, что спектр первого слагаемого состоит из набора $\{\mu_1,...,\mu_n\}$, а разность $F - \tilde{F}$ имеет структуру произведения вектора-столбца на вектор-строку $u \cdot (1,0,...,0)$. Здесь u обозначает вектор-столбец длины n, составленный из разностей коэффициентов характеристических многочленов матриц B и A. Теперь понятно как строить управляющие вектора: в качестве p можно взять Su, а q положить равным первой строке обратной матрицы S^{-1}.

Пусть $\|\cdot\|$ обозначает норму матрицы, согласованную с нормой вектора в координатном пространстве векторов-столбцов. Для того, чтобы обеспечить хорошую обусловленность матрицы базиса Крылова S ($\gamma(S) = \|S\| \cdot \|S^{-1}\| \le 1 + \Delta$), поиск циклического вектора f произ-

водим методом случайного поиска до выполнения условия $\|I - S\| \le \Delta/(\Delta + 2)$. Здесь I - единичная матрица порядка n, а в качестве значения параметра обусловленности Δ выбирается достаточно малое положительное число из условия компромисса между точностью вычислений и временем счёта. Разработана параллельная программа на языке С построения таких возмущений. Ввиду большого объёма вычислений все расчёты производятся на вычислительном устройстве с быстродействующим многоядерным центральным процессором, усиленным системой графических процессоров, поддерживающих технологию многопоточного программирования CUDA [5].

5. NVIDIA CUDA Programming Guide Version 3.0 / http://www.nvidia.com

Поиск циклического вектора f и построение хорошо обусловленной матрицы базиса Крылова S выполняется на центральном процессоре. Однако вычисление столбца h коэффициентов характеристического многочлена матрицы динамической системы A, обратной матрицы S^{-1} к матрице базиса Крылова, детерминанта $\det S$ и управляющих векторов p и q осуществляется на системе графических процессоров одновременно в ходе одного и того же алгоритма, основанного на применении высокоэффективной универсальной матричной операции [6],

***6.* Исламов Г.Г. Универсальная операция над матричными структурами // Современные проблемы вычислительной математики и математической физики : Междунар. науч. конф., посвящ. памяти акад. А. А. Самарского в связи с 90-летием со дня рожд., М., 16-18 июня 2009 г. - М., 2009.**

применённой к расширенной матрице (g, S) с разными стратегиями выбора разрешающих элементов на позиции матрицы Крылова (см. [7]).

***7.* Исламов Г.Г., Коган Ю.В. Об одном алгоритме поиска базисного минора матрицы // Вестник Удмуртского университета. - 2006. - № 1, Математика. - С. 63-70.**

Приводимый в конце доклада фрагмент 3 программы на языке С для CUDA описывает универсальную операцию над матричными структурами для графического устройства, поддерживающего технологию многопоточного программирования:

Ниже описывается начальная информационная структура высокопроизводительного алгоритма построения управляющих векторов. Все вычисления организованы в виде двумерной таблицы, первая из которых имеет вид

$$
\begin{array}{cccccc}
 & & -s_1 & +s_2 & \dots & (-1)^n s_m \\
t_1 & b_1 & d_{11} & d_{12} & \dots & d_{1m} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
t_m & b_m & d_{m1} & d_{m2} & \dots & d_{mm}
\end{array}
$$

В первой строке находятся s-переменные, которые обозначают коэффициенты характеристического многочлена матрицы A. В первом столбце находятся вспомогательные t-переменные, которые будут обмениваться с s-переменными на каждом шаге применения универсальной операции над матричными структурами. Стратегия применения этой операции, правило пересчёта таблицы и математическое обоснование необходимых для наших целей свойств можно найти в работе [7]. Следующие столбцы исходной таблицы составлены из координат соответственно векторов $A^m f, A^{m-1} f, ..., Af, f$, где f есть порождающий вектор базиса Крылова. Они, без учёта первого вектора, образуют *матрицу перехода* S от единичного базиса к базису Крылова. После m-кратного применения универсальной операции получим последнюю таблицу, в первом столбце которой окажутся все s-переменные, во втором столбце будут находиться коэффициенты характеристического многочлена. В последних столбцах находится информация для получения *преобразующей матрицы*, которая вместе с матрицей перехода необходима для построения управляющих векторов p и q. Сама двумерная таблица формируется на центральном процессоре. Обмен информацией между центральным процессором и графическим устройством происходит лишь на этапе передачи матрицы S с центрального устройства на графическое устройство и выдачи управляющих векторов p и q на центральный процессор.

Выбор графического устройства сильно влияет на точность и быстродействие вычислений, производимых с помощью процедуры Pivot. Наиболее перспективными для наших целей должны оказаться графические устройства на базе процессоров Fermi компании NVIDIA [5]. Этот процессор с 960 ядрами имеет унифицированную 64-разрядную схему адресации памяти. Такая адресация позволяет процессору Fermi работать с адресным пространством объемом до 1 Тб.

В заключение отметим, что приведенная выше модель финансового рынка является достаточно простой и не учитывает многих реальных факторов. Однако она допускает обобщения и уточнения в рамках матричной модели с динамически изменяющимися параметрами. Эти более сложные модели также поддаются исследованию на персональных компьютерах, так как размер матрицы P невелик ($n \approx 2^{12}$).

Фрагмент программы 3.

```
//Kernel definition
__global__ void Pivot(float * Ad, float * Bd, int p, int q, int M)
{
float a,b,c,d;
int i = threadIdx.x;
int j = threadIdx.y;
a = Ad[i+M*j];  b = Ad[i+M*q];
```

```
c = Ad[p+M*j];  d = Ad[p+M*q];
if ((i == p) && (j == q)) Bd[i+M*j] = 1 / d;
else if  (i == p)  Bd[i+M*j] = -a / d;
    else if (j == q) Bd[i+M*j] = a / d;
        else Bd[i+M*j] = a - b * c /d;
__syncthreads();
}
#include <stdio.h>
#include <math.h>
int main()
{
//Initial_Matrix
//Generation
#define M 2 // 3 // 8
#define N 3 // 4 // 9
  float * A;
  float * B;
  A = (float *) malloc(M * N * sizeof(float));
  B = (float *) malloc(M * N * sizeof(float));
  float D[M][M] = {{2,3},{1,5}};
            //{{1,1,0},{0,1,1},{1,1,1}};
            /**{{ 1.11, 1.12, 1.13, 1.14, 1.12, 1.13, 1.11, 1.2},
              { 0.01, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.0},
              { 0.00, 0.02, 0.00, 0.00, 0.00, 0.00, 0.00, 0.0},
              { 0.00, 0.00, 0.02, 0.00, 0.00, 0.00, 0.00, 0.0},
              { 0.00, 0.03, 0.00, 0.00, 0.00, 0.00, 0.00, 0.0},
              { 0.02, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.0},
              { 0.00, 0.00, 0.00, 0.00, 0.00, 0.01, 0.00, 0.0},
              { 0.03, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.0}};**/
   float sum;
   float h[M] = {0,1}; // {-1,1,1}; // {0,4,0,-6,0,4,0,1};
   float p[M];
//Krylov Basis
  //initial vector
  for ( int i = 0; i < M; i++)
      if (i == 0) A[i+M*(N-1)] = 1; // sin(3.3*rand());
      else A[i+M*(N-1)] = 0; // sin(7.2*i);
  for ( int j = N-2; j >= 0; j--)
     for ( int i = 0; i < M; i++)
        {sum = 0.;
         for ( int k = 0; k < M; k++)
            sum += D[i][k] *A[k+M*(j+1)];
            A[i+M*j] = sum;
         }
//Print initial matrix
   printf("\n\n");
   for ( int i = 0; i < M; i++)
       {
        for ( int j = 0; j < N; j++)
```

```
            printf("%f    ", A[i+M*j]);
        printf("\n");
        }
    printf("\n");
//Define a memory for Ad, Bd and load matrix A to device in Ad
  float * Ad;
  float * Bd;
  float * Cd;
  int size = M * N *sizeof(float);
  cudaMalloc((void**)&Ad,size);
  cudaMalloc((void**)&Bd,size);
  cudaMemcpy(Ad, A, size, cudaMemcpyHostToDevice);
//Strategy
  //Thread block size
  #define BLOCK_SIZE_X 2 // 3 // 8
  #define BLOCK_SIZE_Y 3 // 4 // 9
    dim3 dimBlock(BLOCK_SIZE_X,BLOCK_SIZE_Y);
//Universal matrix operation
  for ( int k = 0; k < M; k++)
  {
   printf("%d",k);
   Pivot<<<1,dimBlock>>>(Ad,Bd,k,k+1,M);
//Read result from the device
   cudaMemcpy(B, Bd, size, cudaMemcpyDeviceToHost);
   Cd = Ad;
   Ad = Bd;
   Bd = Cd;
//Print result of Pivot
    printf("\n\n");
    for ( int i = 0; i < M; i++)
      {for ( int j = 0; j < N; j++)
          printf("%f    ", B[i+M*j]);
          printf("\n");}
   }//end loop of k;
//Free device memory
  cudaFree(Ad);
  cudaFree(Bd);
  cudaFree(Cd);
//calculation of fee and tax vector and print of it
      printf("\n");
  for ( int i = 0; i < M ; i++)
      {sum = 0.;
       for ( int j = 1; j < N; j++)
           sum += A[i+M*j]*h[j-1];
       p[i] = sum - A[i];
      printf("%f  ", p[i]);
      }
  printf("\n");
//calculation of priority vector and print of it
```

```
    printf("\n");
    for ( int j = 1; j < N; j++)
    printf("%f  ", B[M*j]);
    printf("\n");
}
```

//Results of calculation

```
/**
C:\Users\Islamov\Desktop\NEW_PROJECT>nvcc -run Pivot_2.cu
Pivot_2.cu
tmpxft_000006d8_00000000-3_Pivot_2.cudafe1.gpu
tmpxft_000006d8_00000000-8_Pivot_2.cudafe2.gpu
tmpxft_000006d8_00000000-3_Pivot_2.cudafe1.cpp
tmpxft_000006d8_00000000-12_Pivot_2.ii
7.000000    2.000000    1.000000
7.000000    1.000000    0.000000
0

-3.500000    0.500000    -0.500000
3.500000    0.500000    -0.500000
1

-7.000000    0.000000    1.000000
7.000000    1.000000    -2.000000

-6.000000  -7.000000

0.000000  1.000000
```