01

# Реализация формального решения задачи переноса излучения в атмосфере нейтронной звезды на графических процессорах

© А. Даниленко

Физико-технический институт им. А.Ф. Иоффе РАН,
194021 Санкт-Петербург, Россия
e-mail: danila@astro.ioffe.ru

(Поступило в Редакцию 3 июля 2017 г.)

Представлена реализация на платформе CUDA поиска формального решения задачи переноса излучения в сильнозамагниченной плазме атмосферы нейтронной звезды на графических процессорах с архитектурой CUDA. Для решения используется метод Галеркина с конечными элементами. Реализация на графических процессорах позволяет значительно ускорить вычисления при построении моделей атмосфер нейтронных звезд.

DOI: 10.21883/JTF.2018.04.45715.2410

## Введение

Нейтронные звезды рождаются горячими в результате коллапса массивных звезд на последних стадиях их эволюции. В первые несколько секунд после рождения изолированные нейтронные звезды становятся прозрачными для нейтрино, которые образуются во внутренних слоях звезды за счет разнообразных ядерных превращений. Вследствие этого в последующие 100 тысяч лет нейтронные звезды теряют энергию за счет нейтринного излучения и постепенно охлаждаются. Скорость такого охлаждения зависит от состава, свойств и уравнения состояния материи внутри нейтронных звезд (см., например, [1]). Другой способ получения сведений о сверхплотном веществе нейтронных звезд состоит в принципиальной возможности определения радиусов и масс. Решение уравнений гидростатического равновесия нейтронной звезды для заданного уравнения состояния вещества в ее внутренних областях дает зависимость массы от радиуса. Анализируя тепловое излучение нейтронной звезды, в принципе возможно определить массу, радиус и температуру, и таким образом ограничить состав и состояние сверхплотного вещества (см., например, [2], для обзора методов исследования сверхплотного вещества посредством наблюдений).

Как правило, тепловое излучение нейтронных звезд наблюдается в мягком рентгеновском диапазоне. Тепловое излучение формируется в тонком (порядка сантиметра) слое на поверхности. Соответственно спектр теплового излучения определяется следующими свойствами поверхности: фазовым состоянием, величиной и направлением магнитного поля, химическим составом, температурой. Для того чтобы правильно интерпретировать наблюдаемое тепловое излучение нейтронной звезды, важно учитывать структуру магнитного поля. Поскольку перенос излучения в магнитном поле анизотропен, наблюдаемый спектр зависит также от направления на наблюдателя, а так как нейтронная звезда вращается, то это направление постоянно меняется относительно конфигурации магнитного поля, и спектр излучения изменяется вместе с фазой вращения даже для однородного распределения температуры по поверхности, иначе говоря, излучение пульсирует. Кроме того, теплопроводность во внешних слоях замагниченных нейтронных звезд анизотропна (теплопроводность выше вдоль силовых линий поля), поэтому температура также распределена по поверхности неоднородно (см., например, [3]).

Изначально модель замагниченной водородной атмосферы была построена в работах [4–8]. В этих работах показано, как рассчитать удельную интенсивность излучения небольшого элемента поверхности нейтронной звезды для заданных значений эффективной температуры, магнитного поля и угла между направлением магнитного поля и нормалью к поверхности. Для того чтобы рассчитать поток излучения со всей поверхности нейтронной звезды, которое зарегистрирует удаленный наблюдатель, можно разбить поверхность на небольшие элементы, так что температуру и магнитное поле в пределах элемента можно считать постоянными, посчитать интенсивность для каждого элемента и просуммировать вклады от каждого элемента с учетом искривления света в гравитационном поле звезды.

Для правильной интерпретации наблюдений необходима многопараметрическая модель теплового излучения поверхности нейтронной звезды, с помощью которой можно будет описывать наблюдаемый спектр в зависимости от фазы вращения, форму кривой блеска в зависимости от энергии фотона (или динамический спектр), а также поляризацию теплового излучения. Предполагаемая модель должна зависеть от следующих параметров, которые можно будет варьировать так, чтобы наилучшим образом описать данные наблюдений: величина магнитного поля, температура, масса, радиус, угол между магнитным моментом и осью вращения, угол между осью вращения и направлением на наблюдателя, ускорение свободного падения на поверхности. Практически можно сначала рассчитать интенсивность излучения элемента поверхности на сетке различных

значений энергии фотона, эффективной температуры, магнитного поля и угла между магнитным полем и нормалью к поверхности. Можно представить насколько большой должна быть такая сетка. Например, в пакете анализа данных Xspec модель nsa [6], которая по-существу описывает излучение одного элемента поверхности с фиксированным магнитным полем, задается таблицей с интенсивностями для 14 значений температуры и 1000 значений энергии фотона. Всего — 14000 значений. Теперь если в сетке будет еще хотя бы 10 значений магнитного поля и 10 значений угла, получим, что нам необходимо провести расчет излучения для одного элемента поверхности порядка миллиона раз. Одним из главных практических препятствий в реализации этого плана является то, что расчет теплового излучения для одного элемента поверхности весьма затратен с точки зрения вычислений.

Для решения этой проблемы разработана реализация формального решения уравнений переноса в сильно замагниченной плазме на графических картах. Формальное решение является основным и наиболее ресурсоемким шагом в расчете теплового излучения элемента поверхности. В настоящей работе перенос излучения рассматривается в диффузионном приближении [9], так как с диффузионного приближения начинается любой практический расчет [4]. Перенос излучения в диффузионном приближении кратко описан в разд. 1. Для решения соответствующих уравнений используется метод Галеркина с конечными элементами, который кратко описан в разд. 2. Далее, в разд. 3 описывается реализация метода Галеркина на графических картах. Реализация проверятся на задаче переноса в изотермической атмосфере, которая имеет простое аналитическое решение. В разд. 4 кратко обсуждаются дальнейшие перспективы построения модели излучения нейтронной звезды.

## 1. Перенос излучения в сильно замагниченной плазме

Рассмотрим систему уравнений переноса излучения для двух нормальных мод, обыкновенной и необыкновенной, в диффузионном приближении [9,10]

$$\frac{d}{d\tau} D_j \frac{dJ_j}{d\tau} - s(J_j - J_{1-j}) = k_j(J_j - B_\upsilon/2), \quad j = 0.1. \tag{1}$$

Здесь $d\tau = n_e\sigma_T dz$ — это томпсоновская оптическая глубина, $(4\pi/c)J_j$ — спектральная плотность энергии излучения для моды $j$, $B_\upsilon = B_\upsilon(T)$ — функция Планка. $D_j$ и $k_j$ — это коэффициенты диффузии и поглощения, зависящие от частоты и выраженные в томпсоновских единицах $(n_e\sigma_T)^{-1}$ и $n_e\sigma_T$. Коэффициент диффузии также зависит от угла $\Theta_B$ между магнитным полем и нормалью к поверхности:

$$D_j = D_j^{\parallel}\cos^2\Theta_B + D_j^{\perp}\sin^2\Theta_B,$$

$$D_j^{\parallel} = \int_0^1 \frac{\mu^2 d\mu}{K_j(\mu)}, \quad D_j^{\perp} = \int_0^1 \frac{(1-\mu^2)d\mu}{K_j(\mu)}. \tag{2}$$

Здесь $\mu$ — это косинус угла между волновым вектором $\mathbf{n}$ и направлением магнитного поля, а $K_j$ — это сумма коэффициентов поглощения и рассеяния,

$$K_j(\mathbf{n}) = K_j^a(\mathbf{n}) + K_j^s(\mathbf{n}). \tag{3}$$

Интегральный коэффициент излучения $K_j^s$ выражается следующим образом:

$$K_j^s(\mathbf{n}) = \sum_{i=0}^{1} \int d\mathbf{n}' k_{ji}^s(\mathbf{n}, \mathbf{n}'). \tag{4}$$

Усредненные по углам коэффициенты рассеяния и излучения выражаются следующим образом:

$$k_j = \frac{1}{4\pi}\int d\mathbf{n} K_j^a(\mathbf{n}),$$

$$s = \frac{1}{4\pi}\int d\mathbf{n} d\mathbf{n} K_{01}^s(\mathbf{n}, \mathbf{n}'). \tag{5}$$

Систему уравнений необходимо решать при следующих граничных условиях [4]:

$$2D_j\frac{dJ_j}{d\tau} = J_j\,(\tau = 0),$$

$$J_j \to \frac{1}{2}B_\upsilon(T)\,(\tau \to \infty). \tag{6}$$

Также систему (1) необходимо дополнить условиями сохранения потока и локального термодинамического равновесия. Система уравнений (1) решается методом последовательных приближений [4]. Сперва необходимо задать профиль температуры. Обычно выбирается профиль для серой атмосферы, или эдингтоновский профиль [11],

$$T(\tau_R) = T_{\text{eff}}\left[\frac{3}{4}\left(\tau_R + \frac{2}{3}\right)\right]^{1/4}, \tag{7}$$

где $T_{\text{eff}}$ — эффективная температура, $d\tau_R = K_R d\tau$ — росселандова оптическая глубина, а $K_R$ — росселандово среднее коэффициента поглощения (см. [11]),

$$K_R^{-1} = \frac{3\pi}{4\sigma T^3}\int d\upsilon\,\frac{D_0 + D_1}{2}\,\frac{\partial B_\upsilon}{\partial T}. \tag{8}$$

Для заданного профиля температуры коэффициенты $k_j$ и $s$, а также $B_\upsilon(T)$ в уравнении (1) заданы. В такой постановке — это задача о нахождении формального решения уравнений переноса. Решая эту задачу, мы находим $J_j(\tau)$ и вычисляем поправку к профилю температуры:

$$\Delta T = -\frac{f}{df/dT}; \quad f = \int d\upsilon \sum_{j=0}^{1} k_j\left(J_j - \frac{1}{2}B_\upsilon\right), \tag{9}$$

получаем новый профиль температуры $T(\tau)$, опять решаем систему уравнений (1), пока значение относительной погрешности $\Delta T/T$ не станет меньше заданного. Наиболее затратный с точки зрения объема вычислений этап — это получение формального решения.

Далее мы рассмотрим построение формального решения методом Галеркина с конечными элементами.

## 2. Метод Галеркина с конечными элементами

Сперва рассмотрим частный случай системы (1), для которого $k_0 = k_1 = k$, $s = 0$ и $D_0 = D_1 \equiv D$. Соответственно в этом случае $J_0 = J_1 \equiv J$ и уравнение для $J$ принимает вид

$$\frac{d}{d\tau} D \frac{dJ}{d\tau} = k \left( J - \frac{B_v}{2} \right). \tag{10}$$

В методе Галеркина с конечными элементами решение уравнения (10) на отрезке $[\tau_0, \tau_N]$ ищется в виде (см., например, [12])

$$J(\tau) = \sum_{i=0}^{N} \psi_i(\tau) \tilde{J}_i, \tag{11}$$

где $\tilde{J}_i = J(\tau_i)$ — это значения искомой функции в узловых точках сетки, $N$ — число элементов, а $\psi_i(\tau)$ — кусочно-гладкие полиномы. Для простоты сначала рассмотрим линейные полиномы. Все рассуждения легко обобщаются для полиномов более высокой степени. На элементе $[\tau_i, \tau_{i+1}]$ линейные полиномы имеют вид в локальных координатах элемента $\xi$ (рис. 1):

$$\psi_i(\xi) = \frac{1}{2}(1 - \xi), \quad \psi_{i+1}(\xi) = \frac{1}{2}(1 + \xi),$$

$$\xi = 2 \frac{\tau - 0.5(\tau_i + \tau_{i+1})}{\tau_{i+1} - \tau_i}, \quad \xi \in [-1, 1]. \tag{12}$$

Далее обозначим оператор уравнения (10) как $L(J)$ и запишем уравнение в слабой форме:

$$\int_{\tau_0}^{\tau_N} \psi_j(\tau) L\big(J(\tau)\big) d\tau = 0. \tag{13}$$

Подставляя в (13) разложение (11), интегрируя член со второй производной по частям и меняя местами суммирование и интегрирование, получим

$$-\psi_j D \frac{dJ}{d\tau}\bigg|_{\tau_0}^{\tau_N} + \sum_{i=0}^{N} \tilde{J}_i \left[ \int_{\tau_0}^{\tau_N} D \frac{d\psi_i}{d\tau} \frac{d\psi_j}{d\tau} d\tau + \int_{\tau_0}^{\tau_N} k \psi_i \psi_j d\tau \right]$$

$$= \int_{\tau_0}^{\tau_N} k \frac{B_v}{2} \psi_i d\tau. \tag{14}$$

Коэффициенты $k, D$ и $B_v$ также удобно представить в виде (11). Подставляя разложения для коэффициентов и производя интегрирование, получим систему алгебраических уравнений для нахождения $\tilde{J}_i$:

$$\mathbf{M}\tilde{\mathbf{J}} = \mathbf{g} \equiv \mathbf{G}\mathbf{f}, \tag{15}$$

где $\mathbf{f}$ — это вектор с компонентами $1/2\tilde{k}_i\tilde{B}_i$, а $\tilde{k}_i$ и $\tilde{B}_i$ — значения коэффициента $k$ и функции Планка в узловых точках сетки. Забудем на время про граничные условия и про внеинтегральный член в (14). Тогда для сетки из четырех элементов матрица $\mathbf{M}$ имеет такой вид:

$$\begin{bmatrix} M_{00}^0 & M_{01}^0 & 0 & 0 & 0 \\ M_{10}^0 & M_1^0 + M_{11}^1 M_{12}^1 & 0 & 0 & \\ 0 & M_{21}^1 & M_{22}^1 + M_{22}^2 & M_{23}^2 & 0 \\ 0 & 0 & M_{32}^2 & M_{33}^2 + M_{33}^3 & M_{34}^3 \\ 0 & 0 & 0 & M_{43}^3 & M_{44}^3 \end{bmatrix}. \tag{16}$$

Матрица $\mathbf{G}$ строится аналогично. Верхние индексы обозначают номер элемента сетки, на котором вычисляется данный элемент матрицы, причем матрицы $\mathbf{M}^e$ и $\mathbf{G}^e$ для элемента $e$ имеют следующий вид:

$$\mathbf{M}^e = \frac{2}{\Delta\tau_e} \left[ \tilde{D}_e \mathbf{g}^0 + \tilde{D}_{e+1} \mathbf{g}^1 \right] + \frac{\Delta\tau_e}{2} \left[ \tilde{k}_e \mathbf{n}^0 + \tilde{k}_{e+1} \mathbf{n}^1 \right],$$

$$\mathbf{G}^e = \frac{\Delta\tau_e}{2} \mathbf{n},$$

$$\Delta\tau_e = \tau_{e+1} - \tau_e, \tag{17}$$

где $\tilde{D}_e$ — значение функции $D$ в узловой точке $e$, а элементы матриц $\mathbf{g}^{0,1}$ и $\mathbf{n}^{0,1}$ и $\mathbf{n}$ выражаются через интегралы

$$\int_{-1}^{1} \psi_i' \psi_j' \psi_k d\xi, \quad \int_{-1}^{1} \psi_i \psi_j \psi_k d\xi, \quad \int_{-1}^{1} \psi_i \psi_j d\xi. \tag{18}$$

Интегралы элементарно вычисляются и матрицы имеют следующий вид:

$$\mathbf{g}^0 = \frac{1}{6} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \quad \mathbf{g}^1 = \frac{1}{3} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix},$$

$$\mathbf{n}^0 = \frac{1}{2} \begin{bmatrix} 1 & 1/3 \\ 1/3 & 1/3 \end{bmatrix}, \quad \mathbf{n}^1 = \frac{1}{2} \begin{bmatrix} 1/3 & -1/3 \\ 1/3 & 1 \end{bmatrix},$$

$$\mathbf{n} = \frac{1}{3} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \tag{19}$$

Приведенные выше выражения получены без учета внеинтегрального члена в (14) и соответствуют однородным условиям Неймана в точках $\tau_0$ и $\tau_N$. Теперь рассмотрим граничные условия (6). В соответсвии с условием Дирихле на правой границе значение $\tilde{J}_N$ фиксировано. Соответственно мы можем исключить последнее уравнение, а также значение внеинтегрального члена выражения (14) в точке $\tau_N$ из рассмотрения. С учетом условия на левой

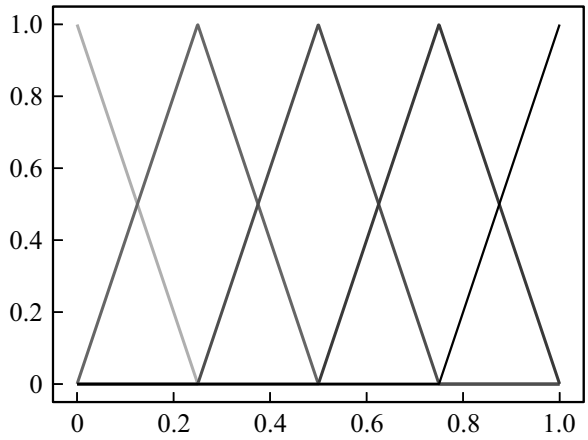


**Рис. 1.** Линейные функции $\psi_i(\tau)$ $(i = 0{-}5)$ для сетки из четырех элементов на отрезке $\tau \in [0, 1]$.

границе значение внеинтегрального члена в точке $\tau_0$ представим следующим образом:

$$\psi_0 D \left.\frac{dJ}{d\tau}\right|_{\tau_0} = D \left.\frac{dJ}{d\tau}\right|_{\tau_0} = \left.\frac{J}{2}\right|_{\tau_0} = \frac{\tilde{J}_0}{2}. \qquad (20)$$

Соотвественно с учетом граничных условий матрица $\mathbf{M}$ и вектор $\mathbf{g}$ принимают вид (для сетки из четырех элементов):

$$\mathbf{M} \to \begin{bmatrix} M^0_{00} + 1/2 & M^0_{01} & 0 & 0 \\ M^0_{10} & M^0_1 + M^1_{11} & M^1_{12} & 0 \\ 0 & M^1_{21} & M^1_{22} + M^2_{22} & M^2_{23} \\ 0 & 0 & M^2_{32} & M^2_{33} + M^3_{33} \end{bmatrix},$$

$$\mathbf{g} \to \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 - M^3_{34} \frac{\tilde{B}_4}{2} \end{bmatrix}. \qquad (21)$$

Важно отметить, что $\mathbf{M}$ — симметричная и положительно определенная матрица, т. е. $\mathbf{M}^\top = \mathbf{M}$ и $\mathbf{x}^\top \mathbf{M}\mathbf{x} > 0$ для $\forall x \neq 0$.

Теперь легко представить, как будет выглядеть алгебраическая система для уравнений (1):

$$\begin{bmatrix} \mathbf{L}_0 & \mathbf{S} \\ \mathbf{S} & \mathbf{L}_1 \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{J}}_0 \\ \tilde{\mathbf{J}}_1 \end{bmatrix} = \begin{bmatrix} \mathbf{G}\mathbf{f}_0 \\ \mathbf{G}\mathbf{f}_1 \end{bmatrix}, \qquad (22)$$

где матрицы $\mathbf{L}_j$ и $\mathbf{S}$ соответствуют следующим операторам:

$$L_j(J) = -\frac{d}{d\tau} D_j \frac{dJ}{d\tau} + (k_j + s)J, \quad S(J) = -sJ. \qquad (23)$$

Рассмотрим, как можно реализовать построение и решение системы (22) на графических картах.

## 3. Реализация на платформе CUDA

Рассмотрим реализацию решения формальной задачи на платформе CUDA. Для этого использовался язык CUDA C. Для знакомства с платформой CUDA и языком CUDA C можно порекомендовать книгу Sanders, Kandrot [13]. Для разработки и использования приложений для графических карт с технологией CUDA необходимо установить пакет CUDA toolkit, в который входит компилятор nvcc, а также некоторые утилиты для отладки программ и полезные библиотеки.

Система уравнений (22) решалась методом сопряженного градиента. Метод сопряженного градиента можно сформулировать как метод последовательных приближений для решения системы $\mathbf{A}\mathbf{x} = \mathbf{b}$, где $\mathbf{A}$ — симметричная положительно определенная матрица. В Приложении A приведен соответствующий псевдокод.

Из приведенного алгоритма видно, что на каждой итерации нам нужно умножать матрицу $\mathbf{A}$ на вектор, а также вычислять прямые произведения и суммы векторов. С точки зрения вычислений самая затратная операция — это умножение матрицы на вектор.

Сперва рассмотрим, как умножать матрицу типа (21) так, чтобы использовать преимущества многоядерных устройств типа графических карт. Сперва мы можем умножить каждую из двумерных матриц $\mathbf{M}^e$ на соответствующий двумерный вектор. Эти операции можно проводить независимо. Далее нам нужно составить из двумерных векторов результирующий вектор. Для примера реализация данной процедуры на языке CUDA C, для матрицы, возникающей при использовании линейных элементов, приведена в Приложении B. Для операций скалярного произведения и сложения векторов использовалась библиотека cuBLAS, которая является частью пакета CUDA toolkit.

Для вычисления коэффициентов линейной системы (22) необходимо предварительно вычислить векторы значений коэффициентов $k_j$, $D_j$, $s$ и $B_v$ системы уравнений (1) в точках сетки $[\tau_0, \tau_N]$. В простейшем случае, когда коэффициенты задаются простыми формулами, можно легко реализовать расчет коэффициентов непосредственно на CUDA. В качестве примера рассмотрим

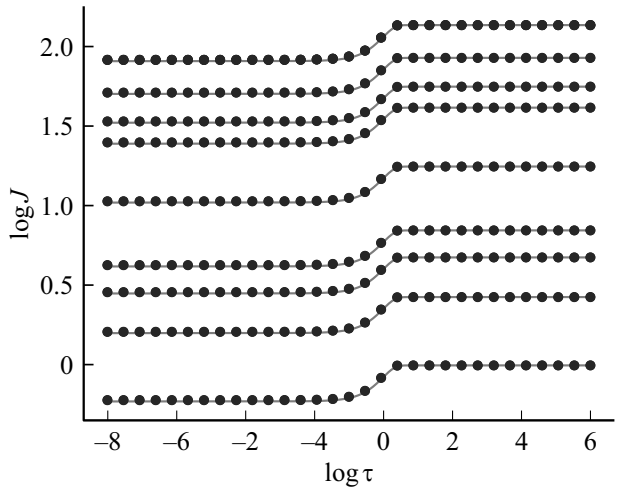


**Рис. 2.** Результаты расчета для изотермической атмосферы в сравнении с аналитическим решением (показано сплошными линиями).

изотермическую атмосферу. Распространение излучения в такой атмосфере описывается уравнением (10) с коэффициентами $D = \text{Const}$ и $k = \kappa\tau$. Тогда аналитическое решение уравнения (10) может быть выражено через функцию Эйри $\phi(\tau)$ [14,15]. Будем считать $\kappa = 1$ и $D = 1$, тогда, учитывая граничные условия (6):

$$J(\tau) = \frac{B_v}{2}\left(1 - C_\phi(\tau)\right),$$
$$C = \phi(0) - 2\phi'(0). \qquad (24)$$

На рис. 2 аналитическое решение (24) сравнивается с расчетом для сетки из 30 точек по $\tau = 10^{-8}-10^{6}$ и 9 точек по энергии $E = 10^{-2}-1$. Число точек было выбрано для удобства представления. В реальных расчетах лучше использовать большее число точек. В общем случае удобно использовать таблицы с предварительно посчитанными значениями на некоторой сетке по температуре и энергии фотона и далее находить необходимые значения посредством интерполяции. Интерполяция также эффективно реализована на CUDA.

## Заключение

Реализация нахождения формального решения для уравнений переноса излучения в сильнозамагниченной плазме на платформе CUDA позволяет ускорить расчет излучения небольшого элемента поверхности нейтронной звезды с заданными значениями температуры, магнитного поля и угла между направлением магнитного поля и нормалью к поверхности. Как обсуждается во Введении, на практике необходимо провести порядка миллиона таких расчетов. Коротко обсудим, что нужно делать после того, как такие расчеты проведены. Можно задать модель нейтронной звезды в виде карты распределения температуры и магнитного поля по поверхности. Тогда для сравнении с данными наблюдений необходимо определять те элементы поверхности, излучение от которых зарегистрирует удаленный наблюдатель для заданных направлений: на наблюдателя, оси вращения звезды, магнитного момента (если поле дипольное) и фазы вращения. Далее эти направления можно будет варьировать так, чтобы наилучшим образом описать наблюдения. Вычислять, какие элементы поверхности видны удаленному наблюдателю, можно, например, методом трассировки лучей (ray tracing), который также можно эффективно реализовать на графических картах.

Автор выражает признательность П. Алексееву за ценные замечания, которые помогли улучшить настоящую работу. Работа выполнена при поддержке Российского фонда фундаментальных исследований, грант мол_а_дк 16-3260129.

## A. Алгоритм построения формального решения

Ниже приведены алгоритмы для решения системы уравнений (1) методом Галеркина с конечными элементами. Также приведен алгоритм для метода сопряженного градиента, с помощью которого решается система (22).

**Algorithm 1.** Решение системы уравнений (1) на графических картах

1: Вычисление $k_j, D_j, s$ и $B_v$ в точках сетки посредством интерполяции. Расчет значений в каждой точке выполняется параллельно.
2: Вычисление $\mathbf{L}_j$, $\mathbf{S}$, $\mathbf{G}$. Расчет выполняется независимо для каждого элемента сетки.
3: Решение системы (22) методом сопряженного градиента.

**Algorithm 2.** Метод сопряженного градиента для решения системы $\mathbf{Ax} = \mathbf{b}$

1: $n \leftarrow 0$,
2: $\mathbf{x}_0 \leftarrow 0$,
3: $\mathbf{r}_0 \leftarrow \mathbf{b}$,
4: $\mathbf{p}_0 \leftarrow \mathbf{b}$,
5: $\delta_0 \leftarrow r_0^\top r_0$,
6: while $\delta_n > \epsilon^2\delta_0 do$,
7: $\alpha_n \leftarrow \delta_{n-1}/(\mathbf{p}_{n-1}^\top \mathbf{A}\mathbf{p}_{n-1})$,
8: $x_n \leftarrow x_{n-1} + \alpha_n \mathbf{p}_{n-1}$,
9: $r_n \leftarrow r_{n-1} - \alpha \mathbf{A}\mathbf{p}_{n-1}$,
10: $\delta_n \leftarrow \mathbf{r}_n^\top \mathbf{r}_n$,
11: $\beta_n \leftarrow \delta_n/\delta_{n-1}$,
12: $\mathbf{p}_n \leftarrow \mathbf{r}_n + \beta_n \mathbf{p}_{n-1}$,
13: $n \leftarrow n + 1$,
14: end while.

## B. Реализация на языке CUDA C умножения матрицы, возникающей в методе линейных элементов, на вектор

Приведен код на языке CUDA C для умножения матрицы типа (21) на вектор.

```
__global__ void multMbyX(float *M, float *X, float *Y) {
__shared__ float cache [N_elem] [2];
int tid=threadIdx.x+blockIdx.x*blockDim.x;
int cache Index=threadIdx.x;
if (tid<N_elem) {
cache[id][0]=M[tid][0]*X[tid]+M[tid][1]*X[tid+1];
cache[id][1] = M[tid][1]*X[tid]+M[tid][2]*X[tid+ 1];
  }
__syncthreads () ;
int i = N_elem − 1;
while (i > 0) {
Y[i] = cache[i][0] + cache[i − 1][1];
i = i − 1;
  }
}
```

## Список литературы


[1] *Yakovlev D.G., Pethick C.J.* // ARA&A. 2004. Vol. 42. P. 169–210.
[2] *Miller M.C.* Astrophysical Constraints on Dense Matter in Neutron Stars // ArXiv e-prints. 2013.
[3] *Greenstein G., Hartke G.* // Astrophysical J. 1983. Vol. 271. P. 283–293.
[4] *Shibanov I.A., Zavlin V.E., Pavlov G.G., Ventura J.* // Astronomy & Astrophysics. 1992. Vol. 266. P. 313–320.
[5] *Pavlov G.G., Shibanov Y.A., Ventura J., Zavlin V.E.* // Astronomy & Astrophysics. 1994. Vol. 289. P. 837–845.
[6] *Pavlov G.G., Shibanov Y.A., Zavlin V.E., Meyer R.D.* // NATO Advanced Science Institutes (ASI) Series C / Ed. by M.A. Alpar, U. Liziloglu, J. van Paradijs. Vol. 450 of NATO Advanced Science Institutes (ASI) Series C. 1995. P. 71.
[7] *Zavlin V.E., Pavlov G.G., Shibanov Y.A., Ventura J.* // Astronomy & Astrophysics. 1995. Vol. 297. P. 441.
[8] *Shibanov Y.A., Zavlin V.E.* // Astronom. Lett. 1995. Vol. 21. P. 3–9.
[9] *Kaminker A.D., Pavlov G.G., Silantev N.A., Shibanov I.A.* // Astrofizika. 1982. Vol. 18. P. 283–299.
[10] *Nagel W.* // Astrophys. J. 1980. Vol. 236. P. 904–910.
[11] *Mihalas D.* Stellar atmospheres. 2nd edition. 1978.
[12] *Fletcher C.A.J.* Computational Galerkin Methods. Berlin, Heidelberg: Springer Berlin Heidelberg, 1984.
[13] *Sanders J.* CUDA by Example: An Introduction to General-Purpose GPU Programming / Jason Danders, Edward Kandrot. 1st edition. Addisin-Wesley Professional, 2010.
[14] *Zel'dovich Y.B., Shakura N.I.* // Astronomicheskii Zhurnal. 1969. Vol. 46. P. 225.
[15] *Zhelezniakov V.V.* // Astrophys. Space Sci. 1981. Vol. 77. P. 279–297.