Малоранговый метод Монте-Карло для моделирования процессов агрегации в открытых систем с несколькими источниками

Р.Р. Дьяченко
Научный руководитель: С.А.Матвеев

Февраль 2023

1 Введение

Не так давно появились новые методы быстрого решения обыкновенных дифференциальных уравнений Смолуховского (ОДУ). Они основаны на малоранговой аппроксимации ядра коагуляции C, которая позволяет уменьшить сложность каждого временного шага с $O\left(N^2\right)$ до $O(NR\log N)$, где R - ранг матрицы $C \in R^{N\times N}$, а N - это число уравнений. Значение N соответствует максимальному размеру кластеров в системе.

Однако не все уравнения могут быть решены таким образом. Хотя все популярные ядра (баллистические, броуновское и т.д.) имеют малый ранг, проблемы возникают, когда происходит гелеобразование, после чего решение ОДУ не существует. Более того, условие сохранения массы на значительном отрезке времени может потребовать очень больших значений N, чтобы иметь конечную систему, которая аппроксимирует бесконечную систему. Наконец, с течением времени обычно требуется соответствующим образом менять временной шаг, поскольку эволюция системы сильно влияет на скорость изменения концентраций.

В то время как вышеупомянутые проблемы могут быть решены с помощью специальных подходов, существует другой простой способ их обойти. А именно, вместо решения ОДУ можно выполнить моделирование методом Монте-Карло с некоторым очень большим конечным числом частиц; в этом случае матрица C определяет вероятности столкновения для частиц соответствующего размера. Тогда условие сохранения массы всегда выполняется, что позволяет изучать гелеобразование, а временной шаг автоматически определяется временем между столкновениями. К сожалению, известные методы Монте-Карло (см. раздел 1.7) требуют слишком много времени для выбора сталкивающихся частиц.

Здесь мы предлагаем модификацию метода Монте-Карло, которая позволяет использовать свойство малого ранга ядра коагуляции для уменьшения требуемого количества операций до значения $O(R\log N)$. Более того, этот подход не требует, чтобы само ядро было малоранговым или имело хорошую малоранговую аппроксимацию, вместо этого достаточно ограничить его сверху некоторым малоранговым ядром $\tilde{C}$.

В этой работе мы адаптируем этот алгоритм для окрытой системы с несколькими источниками, демонстрируем его сходимость на прмиере известных теоретических решений и демонстрируем схожесть полученных решений с резльтатами полученными методами решения ОДУ.

2 Постановка задачи

Мы рассматриваем модель агрегации хорошо перемешанных частиц с двумя источниками: мономерами ($k = 1$) и частицами массой k ($k = I_s > 1$) и полным поглощением для частиц, превышающих максимальную массу ($k = N_{lim}$). Таким образом, уравнение Смолуховского приобретает вид:

$$\frac{dn_k}{dt} = \frac{1}{2}\sum_{j=1}^{k-1} C_{j,k-j}n_j n_{k-j} - n_k\sum_{j=1}^{\infty} C_{k,j}n_j + P_k - R_k \tag{1}$$

где $C_{k,j}$ - ядро агрегации (слияния) двух частиц, содержащих k и j мономеров в себе. Первый член в (1) описывает скорость образования частиц с массой k, второй член описывает скорость

истощения наночастиц из-за столкновений. Слагаемое $P_k \geq 0$ представляет внешний источник (образование частиц), а член R_k представляет удаление слишком крупных наночастиц из системы.

$$P_k = \begin{cases} p_1, & k = 1 \\ p_k, & k = I_s \\ 0, & k \neq 1, I_s \end{cases}$$

$$R_k = \begin{cases} 0, & k \leq N_{lim} \\ \\ \infty, & k > N_{lim} \end{cases} \tag{2}$$

где p_1 - интенсивность внешнего источника для мономеров, а p_k - интенсивность внешнего источника для частиц массой k. Начальными условиями для заполнения кластера являются: $n_k(t = 0) = 0$. Важно отметить, что в модели (1) агрегация кластеров моделируется с детальным разрешением по мономеру. Коэффициенты скорости (или ядра) $C_{i,j}$ являются симметричными и однородными функциями i, j и в большинстве приложений могут быть представлены как $C_{i,k} = K(T)C_{i,j}$, где $C(T)$ зависит только от температуры, а $C_{i,j}$ является двоичной функцией от i, j. В этой статье мы рассматриваем два распространенных случая столкновения ядер для диффузионного и баллистического подходов и устанавливаем единичный безразмерный температурный член $C(T) = 1$. Эти выражения часто используются при моделировании в астрофизике, динамике аэрозолей, флокуляции, грануляции:

$$\begin{aligned} &\textit{Diffusion:} \quad C_{i,j} \;=\; i^a \; j^{-a} + i^{-a} \; j^a \\ &\textit{Ballistic:} \quad C_{i,j} \;=\; (i^{1/3} + j^{1/3})^2 \cdot \sqrt{\tfrac{1}{i} + \tfrac{1}{j}}. \end{aligned} \tag{3}$$

3 Структура методов Монте-Карло

Здесь и далее будем обозначать через $N_i, i = \overline{1, N}$ число частиц размера i. Пусть $N_p = \sum_{i=1}^{N} N_i -$ общее число частиц. Элементы C_{ij} обозначают частоту агрегации частиц размером i и j. Тогда, если произошла агрегация, то вероятность того, что это были частицы i и j, равна

$$P_{ij} = \frac{C_{ij} N_i N_j}{\sum_{i,j} C_{ij} N_i N_j}.$$

Метод Монте-Карло выполняется следующим образом:

1. Инициализируется система с N_i частиц размера i. Устанавливается $t = 0$.
2. Находится интервал Δt до следующего столкновения. Время обновляется на $t := t + \Delta t$.
3. Выбираются размеры в соответствии с вероятностями P_{ij}.
4. Выполняется агрегация:

$$N_i := N_i - 1, \quad N_j := N_j - 1, \quad N_{i+j} := N_{i+j} + 1.$$

5. Пока не достигнут конечный момент времени, возвращаемся к шагу 2.

Простейший способ выбора вероятностей связан с перебором всех N^2 возможных значений P_{ij}, что весьма неэффективно. Представленный далее алгоритм довольно эффективно решает эту проблему в предположении малости ядра агрегации.

4 Малоранговый Монте-Карло метод

Для начала напомним понятие ранга:

Определение 1. Пусть $A \in R^{M \times N}$ - матрица $M \times N$ с вещественными элементами A_{ij}. Минимальное значение R, для которого A может быть выражена как сумма внешних произведений

$$A = \sum_{k=1}^{R} \vec{u}^k \cdot \vec{v}^k \tag{4}$$

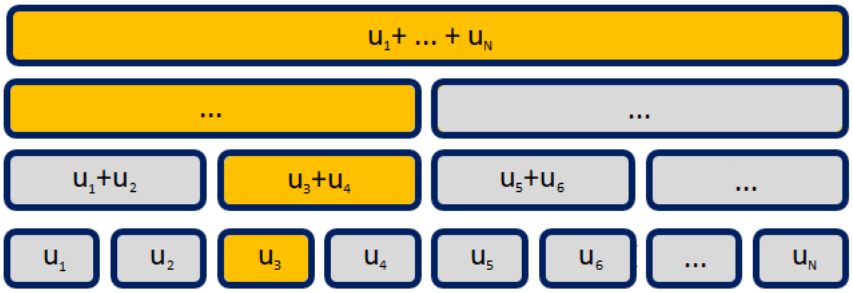

Рис. 1: Дерево отрезков для вектора $\vec{u}$

двух наборов векторов

$$\vec{u}^k \in R^M, \quad k = \overline{1, R},$$
$$\vec{v}^k \in R^N, \quad k = \overline{1, R},$$

называется рангом матрицы A. Когда $R < \min(M, N)$, выражение (4) называется малоранговым разложением матрицы A

Пример:
$A_{ij} = C_{ij} N_i N_j$. Тогда для линейного ядра $C_{ij} = i + j$ матрица A имеет ранг не больше $R = 2$ для любого размера, поскольку она может быть выражена как

$$A = \vec{u}^1 \cdot \vec{v}^1 + \vec{u}^2 \cdot \vec{v}^2$$

или, поэлементно,

$$A_{ij} = iN_i \cdot N_j + N_i \cdot jN_j = u_i^1 v_j^1 + u_i^2 v_j^2$$

с

$$u_i^1 = iN_i, \quad v_j^1 = N_j$$

и

$$u_i^2 = N_i, \quad v_j^2 = jN_j.$$

Причина использовать малоранговое разложение заключается в том, что оно обеспечивает более простой способ выбора размеров частиц. Например, пусть $R = 1$, так что $C_{ij} N_i N_j = u_i \cdot v_j$. Тогда вероятность того, что первая частица имеет размер i, может быть вычислена как

$$p_i = \frac{\frac{\Delta t}{V} \sum_{j=1}^N C_{ij} N_i N_j}{\frac{\Delta t}{V} \sum_{i,j=1}^N C_{ij} N_i N_j} = \frac{\sum_{j=1}^N u_i v_j}{\sum_{i,j=1}^N u_i v_j} = \frac{u_i \sum_{j=1}^N v_j}{\sum_{i=1}^N u_i \sum_{j=1}^N v_j} = \frac{u_i}{\sum_{i=1}^N u_i} \tag{5}$$

и зависит только от компонент вектора $\vec{u}$. Аналогично,

$$p_j = \frac{v_j}{\sum_{j=1}^N v_j} \tag{6}$$

зависит только от компонент вектора $\vec{v}$.

Если ранг больше 1, мы можем представить $A_{ij} = C_{ij} N_i N_j$ как сумму ядер ранга (4) и рассматривать их отдельно, как будто мы рассматриваем разные механизмы агрегации. Чтобы определить, какое ядро использовать, мы можем сравнить общие скорости агрегации и выбрать k-й член вида $\vec{u}^k \cdot \vec{v}^k$ с вероятностью:

$$p_k = \frac{\sum_{i,j=1}^N u_i^k v_j^k}{\sum_{k=1}^R \sum_{i,j=1}^N u_i^k v_j^k} = \frac{\sum_{i=1}^N u_i^k \sum_{j=1}^N v_j^k}{\sum_{k=1}^R \left(\sum_{i=1}^N u_i^k \sum_{j=1}^N v_j^k \right)}.$$

На данный момент еще не видно преимуществ в скорости малорангового подхода. Теперь рассмотрим этот подход более подробно.

Задача состоит в том, чтобы ускорить выбор размеров частиц в соответствии с вероятностями p_i и p_j (5), (6). Это можно сделать, используя дерево отрезков (см. Рис. 1) Дерево

отрезков - это двоичное дерево, которое содержит частичные суммы элементов массива. Всякий раз, когда мы выбираем случайную величину

$$r := \text{rand}(0, 1] \cdot \sum_{i=1}^{N} u_i$$

чтобы определить размер первой частицы, мы можем определить размер за $O(\log N)$ шагов, вычитая частичные суммы u_i из r. Кроме того, дерево отрезков может быть обновлено за $O(\log N)$ операций путем обновления частичных сумм от измененного листа дерева к его корню. Поскольку в общем случае требуется обновить R деревьев, общая сложность составляет $O(R \log N)$. Поскольку R подбирается так, чтобы оно не росло с увеличением максимального размера частиц N, асимптотическая сложность ниже, чем $O\left(N^{\alpha}\right)$ для любого $\alpha > 0$.

Ниже представлен псевдокод алгоритма:

Algorithm 1 LowRank Monte-Carlo

procedure LowRank Monte-Carlo(N_i, C_{ij}, V, maxtime)
  $N_{p0} := \sum_{i=1}^{N} N_i$ // Задание начального числа частиц .N_{p0}
  $N_p := N_{p0}$ // Задание текущего числа частиц N_p
  Создание деревьев отрезков u_t^k и v_t^k на векторах $\vec{u}^k$ и $\vec{v}^k$
  curtime := 0
  while curtime < maxtime **do**
    repeat
      // Первые элементы деревьев отрезков содержат общую сумму частот столкновений, которую мы можем использовать для вычисления общей скорости агрегации
      total_rate := $\sum_{k=1}^{R} \left(u_t^k\right)_1 \left(v_t^k\right)_1$
      // Обновление времени
      $r := total\_rate \cdot rand(0, 1]$
      curtime := curtime + $\frac{2V}{r}$
      // Выбор компоненты ранга 1
      for $k := 1$ **to** R **do**
        // Первые элементы деревьев отрезков содержат общую сумму, которую мы вычитаем из r
        $r := r - \left(u_t^k\right)_1 \left(v_t^k\right)_1$
        if $r \leq 0$ **then**
          break
        end if
      end for
      Выбор размеров i и j с помощью двоичного поиска в деревьях отрезков u_t^r и v_t^r
      // Отклонение столкновения с собой
      if $(i = j)$ **and** $(N_i \cdot \text{rand}(0, 1] \leq 1)$ **then**
        continue
      end if
      until i и j не выбраны
      // Агрегация
      if $i + j > N$ **then**
        $N := 2N$
      end if
      $N_i := N_i - 1$
      $N_j := N_j - 1$
      $N_{i+j} := N_{i+j} + 1$
      $N_p := N_p - 1$
      if $N_p \leq N_{p0}/2$ **then**
        $N_p := 2N_p$
        $V := 2V$
      end if
      Обновление деревьев отрезков
  end while

5 Адаптация алгоритма для открытой системы

Работа источников происходит независимо от взаимодействия частиц в системе, следовательно можно получить уравнение количества добавленных частиц ко времени t:

$$N_k = p_k \cdot V \cdot t$$

Таким образом, будем вести контроль числа частиц из источника, после каждого шага по времени. Для корректной работы алгоритма необходимо после каждого добавления частиц делать обновление деревьев отрезков. При достижении стацианарного состояния на фиксированном объёме происходит процедура удвоения системы необходимая для улучшения точности найденных концентраций. Более детальная последовательность действий представлена в псевдокоде ниже:

Algorithm 2 LowRank Monte-Carlo in open system

procedure OPEN_MONTE-CARLO($C_{ij}, V_{max}, p_k, Lim$, maxtime)
  $N_i = 0$ // В момент $t = 0$ система пуста
  $S_i = 0$ // Количество частиц размера i, попавших в систему к моменту t
  $S_w = 1$ // $w = \underset{k}{\operatorname{argmax}}\, p_k$
  $N_w = 1$
  $N := 1$ // Суммарное число частиц в системе
  $V = 1$
  curtime := $\frac{1}{max(p_k)\cdot V}$
  Создание деревьев отрезков u_t^k и v_t^k на векторах $\vec{u}^k$ и $\vec{v}^k$
  while curtime < maxtime **do**
    repeat
     total_rate := $\sum_{k=1}^{R} \left(u_t^k\right)_1 \left(v_t^k\right)_1$
     $r := total\_rate \cdot rand(0, 1]$
     curtime := curtime + $\frac{2V}{r}$
    // Добавление внешних частиц размера k в систему (аналогично для других источников)
    if curtime $\cdot\, p_k \cdot V - S_k \geq 1$ **then**
      extra = $\lfloor curtime \cdot p_k \cdot V - S_k \rfloor$
      $N_k = N_k$+ extra
      $S_k = S_k$+ extra
      Обновление всех деревьев
    end if
     // Выбор компоненты ранга 1
    for $k := 1$ **to** R **do**
      // Первые элементы деревьев отрезков содержат общую сумму, которую мы вычитаем из r
      $r := r - \left(u_t^k\right)_1 \left(v_t^k\right)_1$
      if $r \leq 0$ **then**
        break
       end if
    end for
    Выбор размеров i и j с помощью двоичного поиска в деревьях отрезков u_t^r и v_t^r
    // Отклонение столкновения с собой
    if $(i = j)$ **and** $(N_i \cdot \text{rand}(0, 1] \leq 1)$ **then**
      continue
    end if
    until i и j не выбраны
    // Агрегация
    if $i + j > Lim$ **then**
      $N_i := N_i - 1$
      $N_j := N_j - 1$
    else
      $N_i := N_i - 1$
      $N_j := N_j - 1$
      $N_{i+j} := N_{i+j} + 1$
      $N := N - 1$
    end if
    if Масса системы стабилизироваласть **and** $V < V_{max}$ **then**
      $N := 2N$
      $V := 2V$
    end if
    Обновление деревьев отрезков
  end while

6 Численные эксперименты

Использую разработанный малоранговый метод Монте-Карло, я повторяю результаты экспериментов статьи Mathematical and Computer Modelling of Dynamical Systems S. A Matveev ,

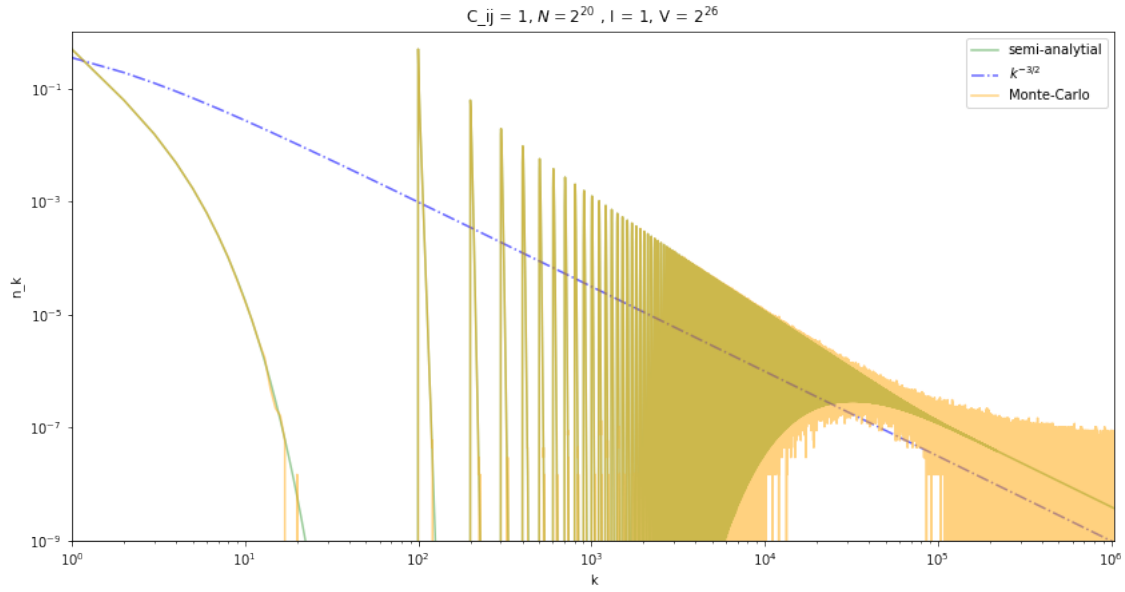

Рис. 2: Сравнение численного стационарного решения уравнений. (1) с полуаналитической оценкой из уравнений (7), (8), для исходных параметров начальных кластеров $I_s = 100$, $p_{I_s} = 1$ и постоянного ядра $C_{i,j} = 1$ с $N_{lim} = 2^{20}$. Асимптотическое правило $n_k \propto k^{-3/2}$ может быть достигнуто только для очень больших размеров кластеров. Сравнение стационарных численных решений уравнений.

A. A Sorokin , A. P Smirnov E.E. Tyrtyshnikov.

6.1 Единичное ядро

В статье было получено полуаналитическое решение для системы с двумя источниками и константным ядром $C_{ij} = 1$ для концентраций частиц размера $\geq k$:

$$n_k = \frac{\sum_{i=1}^{k-1} n_i n_{k-i}}{2N_{tot}} = \frac{\sum_{i=1}^{k-1} n_i n_{k-i}}{2\sqrt{2(p_k+1)}}. \tag{7}$$

и аналитическое рещение для частиц $< k$:

$$n_k = c_k \cdot 2^{-k+\frac{1}{2}} \cdot (1+p_k)^{-k+\frac{1}{2}} = \frac{1}{\sqrt{2\pi}} \frac{\Gamma(k-\frac{1}{2})}{\Gamma(k+1)} \cdot (1+p_k)^{-k+\frac{1}{2}} \approx$$
$$\frac{1}{\sqrt{2\pi}} k^{-3/2} (1+p_k)^{-k+\frac{1}{2}}. \tag{8}$$

В любом случае, для $k \gg p_s$ асимптотическое решение должно сходиться к начальному степенному закону

$$n_k \propto k^{-3/2},$$

На Рис. 2. представлен результат работы алгоритма после 10^9 коагуляций.Кроме того, наблюдаемая сходимость подтверждается ошибкой в Таблице 1. Однакое для получения лучшей точности необходимо увеличение объёма системы до $V \approx 10^9 = 2^{30}$, что требует большего количества вычислений до сходимости.

Таблица 1: Измерения точности метода Монте-Карло для задачи с постоянным ядром, $N = 2^{20}$ уравнений в сравнении с аналитическими выражениями для формально бесконечной системы.

k	I	Relative error of n_k
1	1	$2.3 \cdot 10^{-5}$
2	1	$1.1 \cdot 10^{-4}$
7	1	$2.6 \cdot 10^{-3}$
1	0.1	$3.2 \cdot 10^{-5}$
2	0.1	$4.1 \cdot 10^{-4}$
7	0.1	$7.2 \cdot 10^{-3}$

6.2 Диффузионное ядро

Для системы с двумя источниками и диффузионным ядром не известны теоретические решения, однако разработанный метод Монте-Карло повторил численный результат, полученный методом Андерсона. Моделирвоание было проведено до 10^9 коагуляций и изображены на Рис. 3. и Рис. 4.

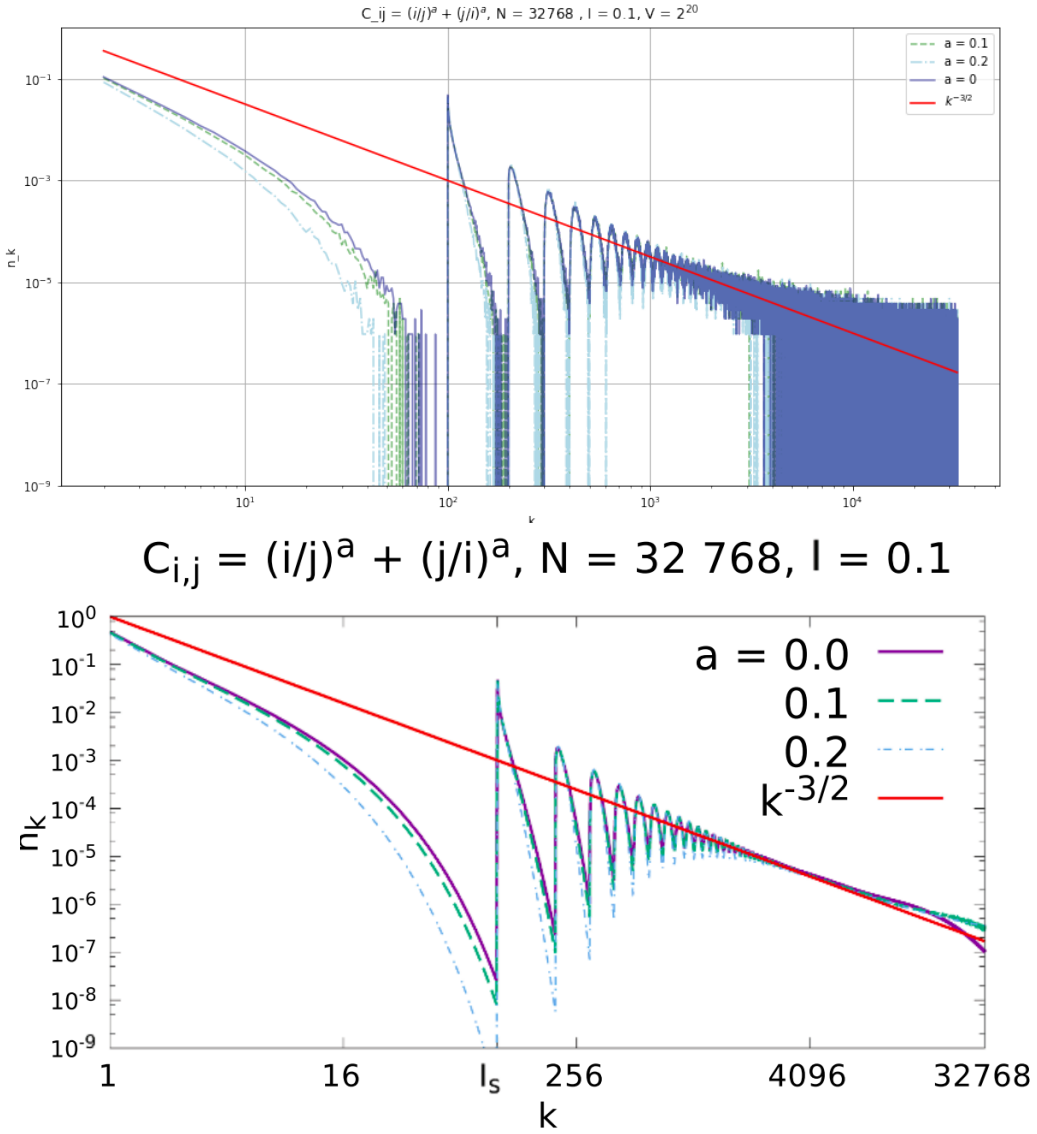

Рис. 3: Численные решение для системы размера $N_{lim} = 32768$ с диффузионным ядром $a = 0.1$ и источниками частиц с $I_s = 100$, $p_{I_s} = 0.1$.

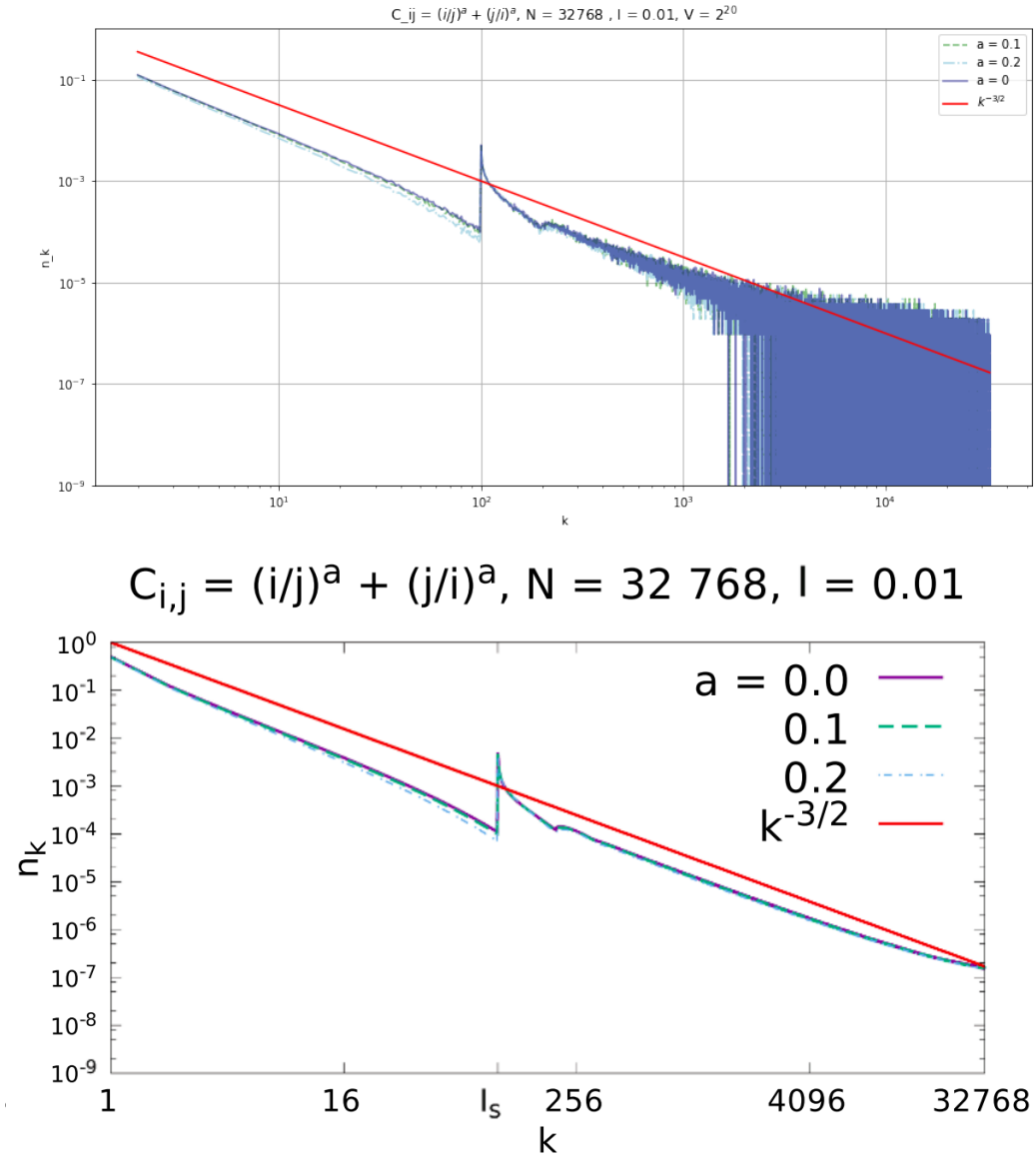

Рис. 4: Численные решение для системы размера $N_{lim} = 32768$ с диффузионным ядром $a = 0.01$ и источниками частиц с $I_s = 100$, $p_{I_s} = 0.1$.

7 Вывод

Разработанный метод показал хорошее качество на константном и диффузионном ядре. Однако для получения лучшей точности алгоритма ещё предстоит придумать ускорения. Например, реализовать идею фиктивных частиц, которая позволяет описывать работу систем с большим количеством частиц. Кроме того, предстоит разработать более формальные требования для остановки системы и масштабирования её в течение работы. Код разработанного алгоритма можно найти на моём Github