УДК 519.688; 532.5; 551.465

# УСКОРЕННЫЙ АЛГОРИТМ ИЗМЕНЕНИЯ ТОПОЛОГИИ ДЛЯ МЕТОДА АДВЕКЦИИ КОНТУРОВ

**А. А. Баранов**[1], **М. С. Пермяков**[2]

Метод адвекции контуров представляет собой один из лагранжевых подходов для моделирования процессов переноса скалярной примеси в квазидвумерных потоках идеальной несжимаемой жидкости, которые встречаются во многих приложениях геофизической гидродинамики. Для сохранения вычислительной эффективности при усложнении структуры переносимых полей этот метод включает в себя процедуру редактирования контуров, а также формы и топологии ограничиваемых ими областей. В статье анализируются основные шаги указанной процедуры и предлагается ускоренный вариант ее алгоритма, позволяющий существенно сократить временны́е затраты. Работа выполнена при финансовой поддержке РФФИ (проект 12–05–31011–мол_а) и ДВО РАН (проекты 12–I–П–19–02 и 12–III–А–07–051).

**Ключевые слова:** геофизическая гидродинамика, вычислительная гидродинамика, контурная динамика, адвекция контуров, редактирование контуров, изменение топологии.

**1. Введение.** Одной из актуальных задач вычислительной геофизической гидродинамики является разработка численных методов для моделирования процессов переноса в квазидвумерных потоках жидкости, характеризующихся высокими значениями чисел Рейнольдса. Характерной чертой таких потоков является наличие тонких (мелкомасштабных) структур переносимых субстанций, которые должны быть сохранены при переходе к численной модели. Наиболее эффективными методами для описания такого рода переноса являются лагранжевы методы, рассматривающие перенос физических величин (пассивный скаляр или потенциальная завихренность) отдельными жидкими частицами или элементами. Среди них можно выделить метод адвекции контуров (МАК) [1, 2]. Этот метод является обобщением (модификацией) известного метода контурной динамики (МКД) [3], который нашел широкое применение в решении задач геофизической гидродинамики [4–7].

Одной из особенностей таких методов является то, что в качестве жидких элементов они используют контуры (изолинии) исходного поля, представленные конечным, но регулируемым набором опорных точек (узлов). Число опорных точек изменяется с течением времени и может быть увеличено (или уменьшено) в зависимости от сложности формы контура. Это позволяет сохранять и достаточно точно описывать более мелкие детали переносимого поля при приемлемых вычислительных ресурсах.

С целью сохранения вычислительной эффективности при усложнении структуры переносимых полей алгоритмы МАК и МКД включают в себя процедуру редактирования контуров (вместе с формой и топологией ограничиваемых ими областей), называемую также “хирургией” [8, 9]. Суть данной процедуры заключается в удалении отдельных контуров либо деталей контура, размеры которых меньше установленного масштаба. Это позволяет значительно замедлить рост общего числа узлов на контурах, что приводит к уменьшению затрат памяти, необходимой для хранения массива узлов контуров, и машинного времени, требующегося на их обработку. Применение такой процедуры является необходимым при проведении долгосрочных расчетов в случае, когда имеет место резкий рост числа мелких структур переносимого поля. Однако сама по себе хирургия также требует достаточно много времени на поиск близких участков контуров. В этом случае возникает необходимость использования ускоренного варианта такого поиска.

В настоящей статье обсуждаются некоторые алгоритмические аспекты процедуры хирургии, рассматриваются известные варианты ускоренного поиска близких участков контуров и предлагается эффективный подход для решения данной задачи. Предлагается новая схема алгоритма хирургии, не зависящая от выбора процедуры поиска, и обсуждаются особенности ее программной реализации. На ряде тестовых задач проводится сравнение времени выполнения указанного алгоритма для каждого из рассмотренных

[1] Дальневосточный федеральный университет, ул. Суханова, 8, 690950, г. Владивосток; аспирант, e-mail: armath123@gmail.com

[2] Дальневосточный федеральный университет, ул. Суханова, 8, 690950, г. Владивосток; Тихоокеанский океанологический институт им. В. И. Ильичёва ДВО РАН, ул. Балтийская, 43, 690041, г. Владивосток; зав. лабораторией, e-mail: permyakov@poi.dvo.ru

вариантов поиска, а также приводятся оценки точности метода адвекции контуров с включением процедуры хирургии.

**2. Адвекция контуров.** Уравнение двумерного переноса скалярного поля $Q(x, y, t)$ в известном поле скорости $\big[u(x, y, t), v(x, y, t)\big]$ может быть записано в виде

$$\frac{DQ}{Dt} \equiv \frac{\partial Q}{\partial t} + u\,\frac{\partial Q}{\partial x} + v\,\frac{\partial Q}{\partial y} = 0.$$

Здесь значения скалярного поля $Q$ сохраняются для каждой подвижной частицы жидкости в любой момент времени, что является основой лагранжевых методов.

Метод адвекции контуров представляет собой один из подходов для лагранжевого описания процесса переноса путем добавления в исходный поток набора трассеров, движение которых отслеживается в отдельные моменты времени. В качестве таких трассеров этот метод использует контуры (изолинии) переносимого скалярного поля $C(z,t) = \big\{(x, y) : Q(x, y, t) = z\big\}$, где $z$ — определяет уровень скалярного поля $Q$. В дискретном случае выбирается конечный набор значений уровней $Z_l$, $l = \overline{1, M}$. Отметим, что множество контуров $C_l$, соответствующих некоторому значению уровня $Z_l$, может состоять из нескольких не связанных между собой частей. Среди них могут встречаться как замкнутые контуры (целиком лежащие внутри расчетной области), так и незамкнутые (концы которых лежат на границе области). Еще одним допущением является кусочно-постоянное распределение исходного поля. В этом случае контуры будут представлять собой границы, разделяющие области, в которых значение переносимого скаляра остается постоянным, при этом на всех контурах необходимо установить одинаковое направление обхода.

Каждый контур, в непрерывном случае состоящий из бесконечного набора точек, связанных с выделенными частицами жидкости, в дискретном виде представляется конечным, но регулируемым числом опорных точек, называемых также узлами контура [5, 7]: $\boldsymbol{X}_i = (x_i, y_i)$, $i = \overline{1, N}$. Таким образом, задача адвекции контура сводится к переносу частиц жидкости, связанных с его опорными точками:

$$\frac{dx}{dt} = u(x, y, t), \quad \frac{dy}{dt} = v(x, y, t). \tag{1}$$

При решении задачи переноса потенциальной завихренности указанный подход должен быть дополнен процедурой инверсии, выполняющей расчет поля скорости по известному значению завихренности [2, 3, 5–7]. Однако здесь мы будем рассматривать лишь адвективную составляющую МАК и МКД, при этом поле скорости полагается известным в каждой точке области.

**2.1. Аппроксимация контура.** В некоторых случаях алгоритмы МАК и МКД используют глобальные или локальные методы для восстановления формы контуров по заданному набору узлов [3, 5, 8, 9]. Для аппроксимации контура будем использовать формулы, приведенные в работе [8]. На отрезке между двумя узлами $\boldsymbol{X}_i$ и $\boldsymbol{X}_{i+1}$ контур будет представлен в виде

$$\boldsymbol{X}(p) = \boldsymbol{X}_i + p\,\boldsymbol{t}_i + H(p)\,\boldsymbol{n}_i, \tag{2}$$

где $\boldsymbol{t}_i = \boldsymbol{X}_{i+1} - \boldsymbol{X}_i = (a_i, b_i)$, $\boldsymbol{n}_i = (-b_i, a_i)$, $p \in [0, 1]$ — безразмерный параметр, $H(p)$ — отклонение контура от прямого отрезка, соединяющего два последовательных узла. Для интерполяции $H(p)$ используется локальный кубический сплайн

$$H(p) = \alpha\,p + \beta\,p^2 + \gamma\,p^3, \tag{3}$$

коэффициенты которого рассчитываются по кривизне контура в опорных точках.

**2.2. Перераспределение узлов контура.** С течением времени (в процессе переноса) контуры скалярного поля могут деформироваться, при этом расстояние между их опорными точками будет увеличиваться либо уменьшаться. Как следствие, возникают участки, которые не могут быть достаточно точно представлены имеющимся набором узлов. Тогда возникает необходимость в обновлении опорных точек контура на каждом временно́м шаге. Их распределение контролируется функцией плотности узлов $\rho$, которая может быть определена следующим образом: $N = \oint\limits_C \rho\,dl$, где $N$ — число узлов на контуре $C$, а $dl$ — дифференциал длины дуги.

Можно выделить два варианта процедуры перераспределения узлов: с полным и частичным их обновлением. В первом случае происходит полная замена старого набора узлов. Во втором случае — отдельные опорные точки фиксируются, а полное обновление выполняется только для узлов, лежащих между ними. Например, в работе [8] в качестве фиксированных точек выступают “угловые” точки (точка помечается угловой, если угол, образуемый двумя прилежащими к ней сегментами, является острым). Такой подход

позволяет сохранять острые углы и соответствующие им тонкие структуры, которые могут быть сглажены при интерполяции кубическими сплайнами.

Пусть $n$ — число узлов между двумя фиксированными узлами. Вначале вычисляется требующееся число промежуточных узлов: $q = \sum_{i=1}^{n} \rho_i\, d_i$, где $\rho_i$ — заранее вычисленная средняя плотность распределения узлов на участке $[i, i+1]$, а $d_i$ — расстояние между точками $\boldsymbol{X}_i$ и $\boldsymbol{X}_{i+1}$. Далее определяется новое число узлов на текущем участке контура: $\widetilde{n} = [q] + 2$. Положение каждого нового узла с индексом $j$ на контуре однозначно определяется индексом сегмента $i$ и безразмерным параметром $p \in [0, 1]$ [8], которые легко находятся из соотношения $j - 1 = p\,\sigma_i + \sum_{k=1}^{i-1} \sigma_k$, где $\sigma_i = \rho_i d_i\, \widetilde{n}/q$ — дробное число новых узлов на участке $[i, i+1]$. После этого их координаты могут быть вычислены по интерполяционным формулам (2) и (3).

Для вычисления средней плотности распределения узлов на каждом интервале могут использоваться различные подходы. Так, в работах [2, 8] предлагается использовать ее степенную зависимость от значений кривизны контура. Плотность распределения узлов контура, пропорциональная кривизне, позволяет одинаково хорошо описывать детали любого масштаба. Далее в нашей работе будем использовать формулы для средней плотности $\rho_i$, приведенные в [2]. Они устанавливают минимально допустимый шаг между двумя последовательными узлами контура равным половине параметра хирургии $\delta$, смысл которого разъясняется в следующем разделе. Для обеспечения адекватного распределения на участках с малым значением кривизны задается также максимально допустимый шаг, равный $\mu L$, где $L$ — характерный размер крупномасштабных структур потока, $\mu \in (0, 1)$ — безразмерный параметр перераспределения.

**3. Хирургия контуров.** При численном моделировании в переносимом поле могут возникать разного рода тонкие структуры, масштабы которых с течением времени будут уменьшаться, а для их представления в численной модели будет требоваться все большее число опорных точек на контурах. По этой причине через определенный интервал времени применяется процедура редактирования контуров, а также формы и топологии ограничиваемых ими областей — хирургия [8, 9]. Эта процедура включает в себя две операции. Первая отвечает за объединение двух близких контуров (либо разбиение одного контура на части). Вторая выполняет устранение чрезмерно тонких нитей (филаментов). Последняя операция также отвечает за удаление отдельных контуров, если число узлов этих контуров меньше заданного порогового значения. Параметры хирургии устанавливают минимальный масштаб мелких структур потока, тем самым ограничивая рост числа опорных точек на контурах. Кроме того, данная процедура способна также предотвращать самопересечения контуров, которые могут возникать при расчете их переноса [1, 10].

В работе [8] хирургия проводится, когда расстояние между узлом и прямым сегментом, соединяющим пару последовательных узлов контура, становится меньше некоторого хирургического масштаба. Кривизной контура здесь, для простоты, можно пренебречь. Однако это накладывает определенные требования на минимально допустимое расстояние между узлом и противопоставленным ему сегментом контура: оно не должно быть меньше, чем максимум нормального отклонения контура от прямого отрезка, соединяющего два последовательных узла. Для процедуры перераспределения узлов контуров и интерполяционных формул, приведенных в работах [2, 8], имеет место критерий, согласно которому хирургия проводится, когда указанное расстояние становится меньше, чем масштаб отсечения $\delta$, который выбирается из следующего соотношения: $\delta > \mu^2 L/8$ (в работе [2] было использовано $\delta = \mu^2 L/4$), где $\mu L$ — соответствует максимальному расстоянию между соседними узлами контура (см. раздел 2).

Несколько иной подход для идентификации близких участков контуров предлагается в работе [9]. Вначале каждый контур окружается некоторой $\delta/2$-окрестностью, которая может быть представлена двумя вспомогательными контурами (лежащими слева и справа от основного контура). После этого для обнаружения близких участков рассматриваются пересечения вспомогательных отрезков, соответствующих сегментам исходных контуров.

Приведем краткое описание основных операций хирургии, которые будут использоваться нами в настоящей статье.

**3.1. Слияние и разбиение контуров.** Когда расстояние между сегментами контуров становится меньше масштаба хирургии $\delta$, контуры или части одного контура (в зависимости от обстоятельств) должны быть объединены [8]. Таким образом, либо один контур разбивается на части, либо отдельные контуры сливаются вместе. Отметим, что операции слияния и разбиения алгоритмически неразличимы и могут быть объединены в одну операцию "связывания" контуров. Данная операция допустима только между контурами, соответствующими одинаковым уровням скалярного поля.

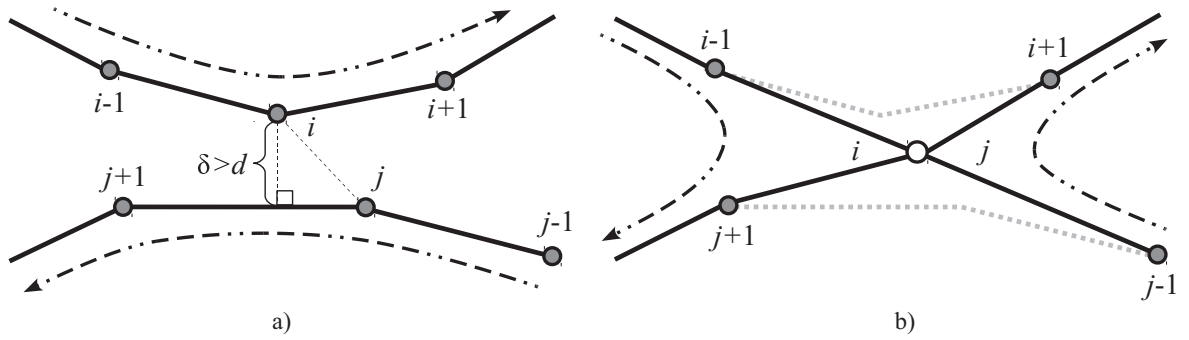


Рис. 1. Операция слияния/разбиения контуров. Расположение узлов контура до применения хирургии (a) и после хирургии (b). Стрелками показано направление обхода

Вначале для тройки узлов ($\boldsymbol{X}_i$, $\boldsymbol{X}_j$, $\boldsymbol{X}_{j+1}$) вычисляется расстояние $d$ между узлом $\boldsymbol{X}_i$ и сегментом $[\boldsymbol{X}_j, \boldsymbol{X}_{j+1}]$. Здесь следует исключить сегменты, прилежащие к узлу $\boldsymbol{X}_i$, т.е. $i \neq j$ и $i \neq j+1$. Если $d < \delta$, то две части контура должны быть объединены следующим образом. Из двух точек $\boldsymbol{X}_j$ и $\boldsymbol{X}_{j+1}$ выбирается наиболее близкая к узлу $\boldsymbol{X}_i$ (обозначим индекс такой точки через $j'$). После этого объединение двух частей контуров может быть осуществлено путем изменения порядка следования узлов: точка $\boldsymbol{X}_{j'+1}$ объявляется следующей для $\boldsymbol{X}_i$, а $\boldsymbol{X}_{i+1}$ — следующей для $\boldsymbol{X}_{j'}$. При этом узлы $\boldsymbol{X}_i$ и $\boldsymbol{X}_{j'}$ смещаются в новые позиции с общими координатами [2]:

$$\boldsymbol{X}' = \frac{1}{2}\,(\boldsymbol{X}_{j'} + \boldsymbol{X}_i). \tag{4}$$

Отметим, что описанная операция сохраняет направление обхода на каждом контуре. Результат ее работы иллюстрирует рис. 1.

**3.2. Удаление тонких нитей (филаментов).** Для предыдущей схемы рассмотрим случай, когда узел $\boldsymbol{X}_i$ соединяется прямым сегментом с одним из узлов $[\boldsymbol{X}_j, \boldsymbol{X}_{j+1}]$, т.е. $i = j-1$ или $i = j+2$. Возьмем в качестве примера $i = j-1$. При проверке критерия хирургии здесь возможны две ситуации. Если угол, образуемый двумя сегментами, прилежащими к узлу $\boldsymbol{X}_j$, является тупым, то расстояние $d$ будет соответствовать длине сегмента $[\boldsymbol{X}_{j-1}, \boldsymbol{X}_j]$, которая обусловлена выбранной плотностью распределения узлов контура и может оказаться меньше $\delta$ (см. раздел 2). Тогда выполнение процедуры связывания узлов не имеет смысла. Аналогичная ситуация имеет место, когда $i = j+2$.

Когда угол между сегментами, прилежащими к точке $\boldsymbol{X}_j$, является острым, такая точка помечается как угловая [8]. Если точка $\boldsymbol{X}_j$ является угловой, то рассматривается тройка узлов ($\boldsymbol{X}_{j-1}$, $\boldsymbol{X}_j$, $\boldsymbol{X}_{j+1}$) и проверяется расстояние между одним из крайних узлов этой тройки и противолежащим ему сегментом контура: $\dfrac{|\boldsymbol{t}_{j-1}\,\boldsymbol{n}_j|}{\max\{|\boldsymbol{t}_{j-1}|, |\boldsymbol{t}_j|\}} < \delta$, где $\boldsymbol{t}_j = \boldsymbol{X}_{j+1} - \boldsymbol{X}_j$ и $\boldsymbol{n}_j$ — вектор, ортогональный по отношению к $\boldsymbol{t}_j$.

Выполнение указанного критерия означает наличие на контуре тонкой нити, которая должна быть устранена по аналогии с описанной выше схемой. При этом отсоединенная часть контура, состоящая из одного или двух узлов, будет удалена (рис. 2).

Указанная процедура может применяться несколько раз подряд до тех пор, пока такая нить не будет удалена полностью. Оставшиеся по завершении такой процедуры контуры удаляются, если они состоят из узлов, число которых меньше заданного порогового значения $N_{\min} > 4$ [8, 9].

Подобные нити могут проявиться как результат проведения предыдущей операции в виде тонких “швов”, возникающих в местах слияния/разбиения частей контуров. Такие “швы” в дальнейшем должны быть удалены.

Обе рассмотренные операции могут быть применены как к замкнутым, так и к незамкнутым контурам. Однако при их использовании на концах незамкнутых контуров следует учитывать, что крайние узлы таких контуров всегда привязаны к границе расчетной области.

**4. Поиск близких участков контуров.** Процедура хирургии включает в себя поиск близких сегментов контуров, соединяющих пары последовательных узлов. На этом этапе для каждого сегмента контура потребуется оценить расстояние до близлежащих узлов (см. раздел 3). Каждый сегмент $[\boldsymbol{X}_i, \boldsymbol{X}_{i+1}]$ однозначно задается индексом своего начального узла $i$. Общее число таких сегментов на каждом замкнутом контуре будет равняться числу его узлов (для незамкнутого контура — на единицу меньше).

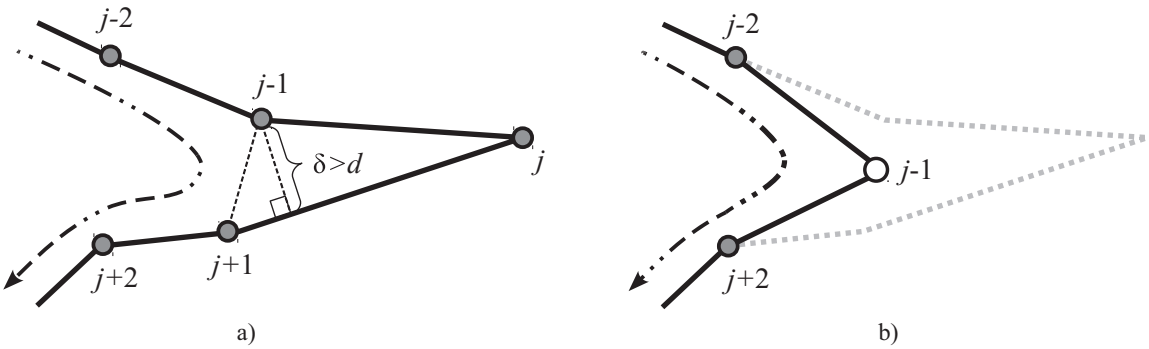


Рис. 2. Операция удаления тонких нитей. Расположение узлов контура до применения хирургии (a) и после хирургии (b). Стрелками показано направление обхода

При разбиении на части контура, состоящего из $k$ сегментов, возникает необходимость рассмотреть $k-3$ узла (за исключением узлов, принадлежащих смежным сегментам). Для случая слияния контуров потребуется рассмотреть узлы, принадлежащие другим контурам. В общем случае вычислительная сложность такого поиска может достигать $O\big(N^2\big)$, где $N$ — полное число узлов на всех контурах, что делает данную процедуру весьма дорогостоящей в плане вычислительных ресурсов. В связи с этим имеет смысл рассмотреть возможные способы для ускорения алгоритма поиска.

**4.1. Алгоритм поточечного поиска с описанными прямоугольниками.** Первый способ построения ускоренного варианта хирургии ранее был изложен в работах [2, 9]. Суть его заключается в уменьшении общего времени поиска узлов на этапе слияния путем выделения таких пар контуров, расстояние между которыми допускает существование узлов и сегментов, удовлетворяющих условию хирургии. Иначе говоря, вначале производится поиск близких пар контуров, а затем для каждой такой пары производится поточечный поиск, т.е. сравнивается расстояние от каждого сегмента первого контура до каждого узла второго контура. Преимущество такого способа заключается в исключении из поиска тех контуров, расстояние между узлами и сегментами которых однозначно больше установленного масштаба $\delta$.

Для определения пар таких контуров используется следующий алгоритм [2]. На начальном этапе хирургии выполняется предварительный расчет минимальных и максимальных координат для каждого контура: $x_{\min}$, $y_{\min}$, $x_{\max}$, $y_{\max}$. После этого вычисляются координаты центра $x_c$, $y_c$ и размеры $w_x$, $w_y$ описанного прямоугольника, который заключает в себе текущий контур:

$$x_c = \frac{x_{\max} + x_{\min}}{2}\,, \quad y_c = \frac{y_{\max} + y_{\min}}{2}\,, \quad w_x = \frac{x_{\max} - x_{\min}}{2}\,, \quad w_y = \frac{y_{\max} - y_{\min}}{2}\,.$$

Далее выполняется “расширение” таких прямоугольников на величину $\delta/2$. На этапе слияния контуров поточечный поиск ведется только между теми контурами, “расширенные” прямоугольники которых перекрываются, т.е. выполняются неравенства $|x_c - x'_c| \leqslant w_x + w'_x + \delta$, $|y_c - y'_c| \leqslant w_y + w'_y + \delta$, где штрихом обозначены величины, соответствующие второму контуру. Узлы и сегменты таких контуров, не попавшие в зону пересечения двух прямоугольников, также могут быть исключены из поиска [9], что позволит сократить число операций сравнения.

Следует отметить, что такой подход дает достаточно грубую оценку для близости контуров. В действительности пересечение прямоугольников, описывающих каждый контур, еще не означает наличие на них близких деталей (рис. 3).

**4.2. Алгоритм поточечного поиска с сеточным представлением контуров.** Более точный алгоритм для выделения близких частей контуров был предложен в [10]. Здесь каждый контур отображается на соответствующую ему регулярную сетку, которая может быть представлена как двумерный массив ячеек. Каждая ячейка принимает одно из двух значений: 1, если как минимум одна из линий контура проходит через нее, и 0, если таких линий нет. При обходе контура для каждого $i$-го сегмента поточечный поиск близких узлов ведется только на тех контурах, множество ненулевых ячеек которых попадает в его $\delta$-окрестность. Для простоты такая окрестность может быть аппроксимирована следующим прямоугольником:

$$\big\{\min\{x_i, x_{i+1}\} - \delta \leqslant x \leqslant \max\{x_i, x_{i+1}\} + \delta\big\} \times \big\{\min\{y_i, y_{i+1}\} - \delta \leqslant y \leqslant \max\{y_i, y_{i+1}\} + \delta\big\}.$$

Оба описанных алгоритма способны снизить время поиска близких узлов и сегментов только в том слу-

чае, когда они принадлежат разным контурам. Для поиска близких узлов и сегментов, принадлежащих одному контуру, данный подход не применим. В этом случае поиск следует осуществлять по стандартной поточечной схеме. Сложность такого поиска составляет $O(k^2)$, где $k$ — число узлов на текущем контуре.

**4.3. Алгоритм сеточного поиска.** Этот алгоритм направлен на уменьшение зоны поиска узлов, близких к текущему сегменту, путем разбиения расчетной области на множество непересекающихся регионов. Наиболее простым вариантом является разбиение области на регулярный массив ячеек сетки [12] (здесь и далее будем рассматривать прямоугольную сетку с постоянным шагом). Ниже будем называть такую сетку хирургической.

Вначале для каждого узла контура определяется содержащая его ячейка (для сетки с постоянным шагом такая операция может быть выполнена за константное время). После этого он добавляется в список узлов, соответствующих этой ячейке, что позволяет уменьшить зону поиска близких узлов до набора близлежащих ячеек сетки. Данный подход достаточно прост в реализации по сравнению с алгоритмами двумерного поиска, использующими иерархические структуры данных (такие как 2d деревья [12]), и требует минимум операций для корректировки списка узлов в случае их добавления, удаления или перемещения (см. раздел 6). При надлежащем выборе шага сетки рассматриваемый поиск будет происходить за $O(N)$ операций, что значительно ниже, чем для поточечного поиска.

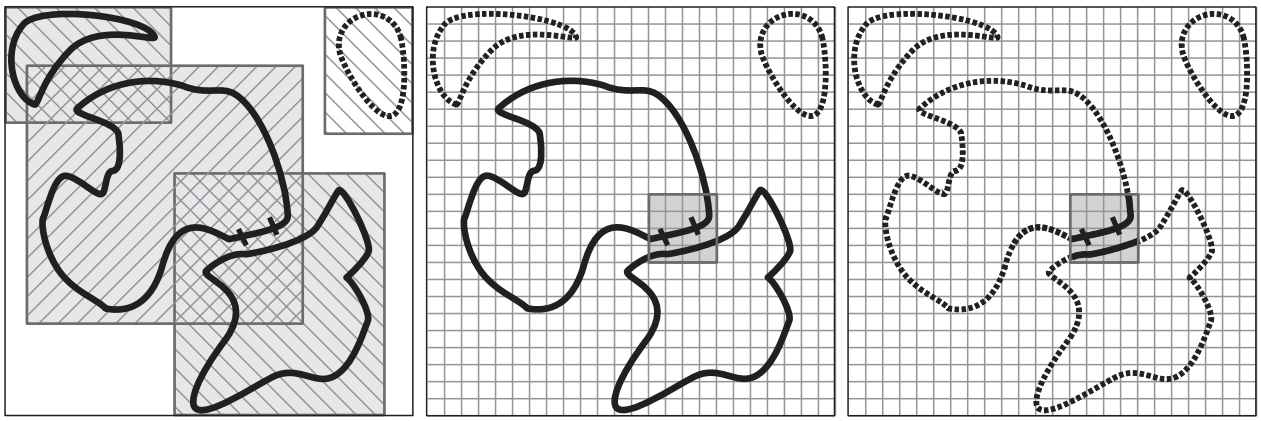


Рис. 3. Демонстрация алгоритмов поиска близких узлов для сегмента, отмеченного на рисунке двумя штрихами. Сплошными линиями выделены участки контуров, участвующие в поиске, пунктирными линиями — участки контуров, исключенные из поиска: a) поточечный поиск с описанными прямоугольниками; b) поточечный поиск с сеточным представлением контуров; c) сеточный поиск

Как упоминалось выше, хирургия проводится тогда, когда расстояние между некоторым узлом и сегментом контура становится меньше $\delta$. Как можно видеть из рис. 1, необходимое условие проведения хирургии между узлом $\boldsymbol{X}_i$ и сегментом $[\boldsymbol{X}_j, \boldsymbol{X}_{j+1}]$ может быть записано в форме

$$|\boldsymbol{X}_j - \boldsymbol{X}_i| \leqslant R, \tag{5}$$

где $R$ — "радиус хирургии", равный максимально допустимому расстоянию между текущим $\boldsymbol{X}_j$ и искомым $\boldsymbol{X}_i$ узлами: $R = \sqrt{\delta^2 + {d_{\max}}^2}$ , где $d_{\max}$ — максимальное расстояние между двумя последовательными узлами контуров. Таким образом, для каждого $j$-го сегмента достаточно рассмотреть только те узлы, которые попадают в круг радиуса $R$ с центром в точке $(x_j, y_j)$. В свою очередь, ячейки хирургической сетки, которые могут содержать такие узлы, формируют "зону поиска".

Рассмотрим способ выделения набора ячеек, попадающих в зону поиска для узла $\boldsymbol{X}_j$. Круг сам по себе является весьма неудобной фигурой, поскольку для определения попадающих в него ячеек сетки может потребоваться излишнее число операций сравнения. По этой причине имеет смысл заменить его более простой фигурой (в данном случае рассматривается прямоугольник). Определим прямоугольный шаблон ячеек, полностью покрывающих квадрат со сторонами $2R$ и центром в текущей точке. Далее для каждой точки, попавшей в выделенный диапазон, необходимо будет выполнить дополнительно проверку (5).

Отметим, что общее число ячеек, попадающих в зону поиска, зависит от положения самой точки $\boldsymbol{X}_j$ относительно границ содержащей ее ячейки. Несложно показать, что в случае, когда шаг хирургической сетки $h \geqslant 2R$, минимальное число таких ячеек будет равно 1, а максимальное их число равно 4. При увеличении $h$ их площадь будет расти и, как следствие, в зону поиска может попасть большое число

лишних узлов. Время, необходимое на обработку всех таких узлов, также увеличится. При уменьшении шага сетки $h < 2R$ число таких ячеек возрастает, однако их суммарная площадь $S$ будет уменьшаться. При выборе параметров алгоритма следует учитывать, что при достаточно малом шаге хирургической сетки количество памяти, необходимой для хранения всех ячеек сетки, значительно возрастет, при этом для любого шага $h$ будет иметь место неравенство $S \geqslant 4R^2$. Численные оценки влияния размера шага хирургической сетки на общую эффективность метода приводятся в разделе 7.

В отличие от алгоритма, описанного выше, здесь для всех контуров будет использоваться общая хирургическая сетка. Однако такого рода стратегия накладывает некоторые ограничения на порядок следования отдельных операций хирургии, которые подробно обсуждаются в следующем разделе.

Данный подход с несущественными изменениями может быть также применен для поиска близких узлов, принадлежащих контурам, соответствующим различным значениям скалярного поля. Это, в свою очередь, может использоваться в некоторых алгоритмах обнаружения и предотвращения пересечений контуров, описанных в работах [10, 11].

**5. Общая схема процедуры хирургии.** Для использования и корректного сравнения описанных выше процедур поиска нам потребуется рассмотреть некоторые особенности, возникающие при реализации процедуры хирургии.

Во-первых, обсудим условие окончания процедуры поиска близких узлов и сегментов контуров. Поиск может быть остановлен после первого найденного узла, который удовлетворяет критерию хирургии для некоторого выделенного сегмента. В этом случае скорость выполнения поиска и дальнейшее поведение хирургии контуров будут сильно зависеть от порядка обхода узлов, который может отличаться для различных алгоритмов поиска. Наиболее приемлемым вариантом является поиск узла, расстояние до которого минимально. Иначе говоря, поиск проводится до тех пор, пока не будут рассмотрены все узлы, попадающие в зону поиска. В случае поточечного алгоритма для этого потребуется просмотреть все узлы, принадлежащие как текущему контуру, так и близлежащим контурам.

Во-вторых, следует отметить, что каждая операция слияния/разбиения будет порождать новый набор контуров. Это является неприемлемым для вариантов поточечного поиска, которые рассматривались в разделе 4. После каждой такой операции потребуется обновление вспомогательных массивов, что может свести на нет преимущество их использования.

Для сохранения эффективности может быть предложена схема, при которой операции слияния и разбиения контуров разделены и выполняются последовательно. Например, в работе [9] используется следующая последовательная схема. Вначале выполняется поиск всех возможных сближений на каждом отдельно взятом контуре, а затем выполняется процедура разбиения с последующим удалением чрезмерно тонких участков. Таким образом, будут удалены разного рода тонкие нити, “затяжки” и “перемычки” (места, где противоположные участки контуров сходятся между собой ближе, чем на величину $\delta$). После этого для всех оставшихся и вновь появившихся контуров могут быть заполнены вспомогательные массивы. Затем производится сравнение между парами близких контуров. В этом случае происходит их слияние.

При таком подходе необходимо различать узлы и сегменты, принадлежащие разным контурам. Однако при использовании сеточного поиска, описанного в разделе 4, возникает проблема с идентификацией контура, которому принадлежит найденный узел. Для решения данной проблемы может быть использован дополнительный массив, хранящий соответствия узлов отдельным контурам. Естественно, что при операции слияния/разбиения потребуется провести обновление записей в массиве. В некоторых случаях это может значительно понизить вычислительную эффективность такого метода.

Ниже предлагается схема, позволяющая согласовать результаты, полученные при использовании различных техник поиска близких участков контуров, и устранить различия между операциями слияния и разбиения путем объединения их в операцию “связывания” контуров. Особенности ее программной реализации обсуждаются в следующем разделе.

Первый этап заключается в удалении тонких нитей (филаментов), что позволяет уменьшить число точек на контурах и сократить общее время процедуры поиска. Далее выполняется поиск “сближений” контуров либо частей одного контура, для чего проверяются расстояния между их узлами и сегментами. Найденный участок однозначно задается тройкой значений $(i,\ j,\ j+1)$, где $[j,\ j+1]$ — соответствует текущему сегменту, $i$ — индекс ближайшего к нему узла, при этом $i \neq j-1$, $i \neq j$, $i \neq j+1$ и $i \neq j+2$ (см. раздел 3).

После обнаружения такого “сближения” производится операция “связывания” контуров, заключающаяся в изменении порядка следования узлов (как показано на рис. 1). Их координаты при этом смещаются в соответствии с (4). Далее потребуется произвести удаление тонких нитей (швов), образовавшихся в ме-

стах слияния/разбиения двух участков контура. Удаленные при этом узлы должны быть исключены из дальнейшего поиска. Для хранения информации о новом порядке следования узлов используется вспомогательный массив. Исходный порядок узлов при этом сохраняется, что позволяет идентифицировать узлы по их принадлежности старым контурам и избежать полного обновления вспомогательных массивов, используемых в алгоритмах с поточечным поиском. Заметим, что для алгоритма с сеточным поиском в этом нет необходимости, так как по существу он рассматривает не принадлежность узлов отдельным контурам, а их относительное положение на плоскости.

Операции "связывания" и дальнейшее удаление "швов" приводят к возникновению новых сегментов контуров, которые могут также удовлетворять условию хирургии для какого либо узла. В этом случае может потребоваться повторный поиск с целью обнаружения таких узлов и связанных с ними "сближений".

**6. Программная реализация.** В большинстве существующих работ, рассматривающих процедуру хирургии, не уделяется должного внимания детальному описанию ее программной реализации, хотя реализация такой процедуры представляет собой весьма нетривиальную задачу.

Общая схема процедуры хирургии, приведенная в предыдущем разделе, может быть записана в виде набора следующих процедур.

1. Удаление тонких нитей (филаментов).

2. Поиск всех возможных "сближений" контуров (близких узлов и сегментов).

3. Процедура "связывания" контуров (меняет порядок следования узлов).

4. Процедура "сборки" контуров (собирает обособленные части контуров, возникшие в результате предыдущей операции).

5. Процедура сортировки массивов контуров.

Далее обсудим особенности программной реализации указанной схемы и каждого из алгоритмов поиска, приведенных в разделе 4. Ниже будем использовать символ $\leftarrow$ для обозначения операции присваивания.

Пусть $X$ и $Y$ — массивы, хранящие координаты узлов контуров; $I$ и $N$ — массивы контуров, такие, что $I_c$ — хранит индекс начального узла контура $c$ в массивах $X$ и $Y$, $N_c$ — число узлов на контуре $c$. Общее число не связанных между собой контуров обозначим через $nc$. Здесь и далее будем рассматривать контуры, соответствующие одному уровню скалярного поля.

Положим, что перед выполнением процедуры хирургии все узлы упорядочены в своих массивах. Индекс последнего узла контура $c$ тогда будет равен $I_c + N_c - 1$. Представим контур в виде двунаправленного списка своих узлов. Обозначим через LAST и NEXT массивы, задающие порядок следования узлов контура. Для узла с индексом $i$ обозначим $\mathrm{LAST}_i$ — индекс предыдущего узла цепочки и $\mathrm{NEXT}_i$ — индекс следующего узла, при этом для всех замкнутых контуров имеем $\mathrm{NEXT}_{i_2} = i_1$, $\mathrm{LAST}_{i_1} = i_2$, а для незамкнутых $\mathrm{NEXT}_{i_2} = 0$, $\mathrm{LAST}_{i_1} = 0$ (здесь $i_1$, $i_2$ обозначают начальный и последний узлы контура соответственно). Удаление $i$-го узла и "связывание" частей контуров (изменение порядка следования узлов) могут быть выполнены путем изменения соответствующих записей в массивах NEXT и LAST. Кроме того, в дальнейшем нам понадобится массив состояний STAT, такой, что $\mathrm{STAT}_i > 0$, если $i$-й узел лежит на контуре, и $\mathrm{STAT}_i = 0$, если $i$-й узел был удален. Перед началом процедуры хирургии все записи массива STAT должны быть заполнены единицами.

Первый этап алгоритма заключается в последовательном обходе узлов всех исходных контуров с целью обнаружения на них тонких нитей. При обнаружении такой нити она удаляется путем изменения соответствующих записей в массивах NEXT, LAST и STAT.

Следующий этап алгоритма включает в себя поиск близких узлов и сегментов контуров с целью их дальнейшего слияния/разбиения. При реализации алгоритмов поиска, рассмотренных в разделе 4, нам потребуется провести ряд предварительных расчетов, специфических для каждого из них.

Рассмотрим подготовительный этап для алгоритмов с поточечным поиском. Отличие указанных алгоритмов состоит в критерии для определения близости контуров. Для алгоритма, использующего описанные прямоугольники, потребуется завести массив $G$ размерности $4 \times nc$. Указанный массив используется для хранения размеров ($wx$, $wy$) и центров ($cx$, $cy$) прямоугольников, соответствующих каждому контуру. Для алгоритма, использующего сеточное представление контуров, понадобится массив $G$ размерности $nx \times ny \times nc$, где $nx$ и $ny$ — размеры двумерной сетки. Перед началом хирургии все значения в массиве $G$ устанавливаются нулевыми. На предварительном этапе происходит обход контуров и заполнение соответствующих записей массива $G$. При смещении какого-либо из узлов контура может потребоваться их обновление. Так, для поточечного поиска с описанными прямоугольниками придется расширить прямоугольник текущего контура, если такой узел выходит за его пределы. Для поточечного поиска с сеточным представлением контуров потребуется заполнить ячейки, соответствующие прилежащим к текущему узлу

сегментам.

Особое внимание уделим описанию реализации алгоритма с сеточным поиском. Для данного алгоритма нам потребуется двумерный массив ячеек "хирургической" сетки $G$ размерности $nx \times ny$, изначально заполненный нулевыми значениями. Далее требуется провести предварительный расчет с целью размещения узлов контура в соответствующие ячейки сетки. Каждая такая ячейка будет хранить указатель на полный список индексов всех размещенных в ней узлов. Такой список может быть организован в виде двух массивов STAT и NEXT размерности $N$ (где $N$ — общее число узлов на всех контурах), которые вначале заполнены нулями. При обходе контура для каждого узла $i$ определяем его текущую ячейку $(k, l)$. После этого помещаем его в список узлов данной ячейки: извлекаем верхний узел из списка $(j \leftarrow G_{kl})$; если $j \neq 0$, то размещаем его после текущего узла ($\mathrm{STAT}_j \leftarrow i$, $\mathrm{NEXT}_i \leftarrow j$); помещаем текущий узел в вершину списка ($G_{kl} \leftarrow i$). В дальнейшем обход списка узлов, попадающих в текущую ячейку, происходит от "головы" к "хвосту": от $j \leftarrow G_{kl}$ до тех пор, пока $j \neq 0$, двигаемся вниз по списку ($j \leftarrow \mathrm{NEXT}_j$). После того как было произведено смещение узла, может потребоваться его перемещение из старой ячейки в новую. Удаление узла из списка будет происходить путем изменения соответствующих записей в массивах STAT и NEXT.

На этапе поиска сближений для каждого исходного контура $c$ последовательно рассматриваются сегменты от $i_1 \leftarrow I_c$ до $i_2 \leftarrow I_c + N_c - 1$, для которых выполняется поиск близких узлов. При этом из поиска исключаются узлы $i$ и связанные с ними сегменты, которые были удалены ранее ($\mathrm{STAT}_i = 0$). Если сближение было найдено, то происходит операция связывания узлов с дальнейшим удалением тонких нитей (как это было описано выше).

Как уже упоминалось, для новых сегментов, возникших в результате операции связывания, может также потребоваться дальнейший поиск близких узлов. Иначе говоря, процедуры поиска и связывания узлов будут происходить за несколько итераций. На каждой $m$-й итерации будем помечать все смещенные сегменты $i$ как $\mathrm{STAT}_i \leftarrow m + 1$. Соответственно при обходе узлов контура на $m$-й итерации поиск близлежащих узлов будет проводиться только для тех сегментов $i$, для которых $\mathrm{STAT}_i = m$. Цикл завершается, когда либо не было найдено новых сближений, либо $m$ превысило максимально допустимое число итераций $M_{\max}$ (в большинстве случаев будет достаточно использовать $M_{\max} = 2$).

Дальнейшая процедура "сборки" контуров заключается в последовательном обходе массива контуров и подсчете числа их узлов. Здесь следует обратить внимание на одну важную деталь. Когда операции слияния и разбиения объединены в операцию "связывания", существует возможность потерять начальный узел нового контура. Когда $i$-й и $j$-й узлы лежат на одном контуре, при использовании операции "связывания" один из них окажется на отделенной части, при этом мы можем потерять начальный узел отделенного контура. Во избежание этого индексы $i$ и $j$ должны быть добавлены в массив $I$ как начальные узлы новых контуров $c_1$, $c_2$: $I_{c_1} \leftarrow i$, $I_{c_2} \leftarrow j$ (при обходе массива контуров на этапе поиска сближений эти новые записи игнорируются).

Рассмотрим теперь случай, когда $i$-й и $j$-й узлы изначально лежат на разных контурах. Тогда при объединении этих контуров цепочки узлов, начинающиеся от $i$-го узла и $j$-го узла, будут пересекаться. В этом случае возникает ситуация с повторным обходом уже рассмотренных узлов на этапе "сборки" контуров. Для того чтобы устранить этот недостаток, по завершении обхода текущей части контура будем помечать все ее узлы $i$ как удаленные: $\mathrm{STAT}_i \leftarrow 0$. Тогда при переходе к следующему контуру необходимо проверить условие, что его начальный узел не был удален ранее. В противном случае массив начальных узлов сокращается: $I_c \leftarrow I_{nc}$, $nc \leftarrow nc - 1$.

По завершении процедуры сборки контуры однозначно задаются индексами своих начальных узлов ($i = I_c$, $c = \overline{1, nc}$) и порядком их следования (LAST, NEXT). Их дальнейшая сортировка в новых массивах $X'$, $Y'$ выполняется последовательным обходом составляющих их узлов.

**7. Тестирование.** Здесь мы обсудим ряд численных экспериментов с целью получения оценок производительности рассмотренной схемы хирургии при использовании различных алгоритмов поиска, описанных в разделе 4. В качестве теста рассмотрим случай адвекции контуров в известном поле скорости с применением процедуры хирургии на каждом временно́м шаге. Для интегрирования уравнений движения (1) воспользуемся схемой Рунге–Кутты 4-го порядка.

Рассмотрим задачу адвекции контуров в прямоугольной области $(-\pi, \pi)\times(-\pi, \pi)$ на временно́м интервале $T \in [0, 40]$. На границе будем полагать периодические условия по обеим координатам. Поле течения задается функцией тока

$$\psi(x, y, t) = 0.05 \sin(3\,y)\left[\sin(2\,x) + 4\cos(0.6\,t)\cos(2\,x)\right].$$

Компоненты поля скорости могут быть получены из определения: $u(x, y, t) = -\dfrac{\partial\psi}{\partial y}$, $v(x, y, t) = \dfrac{\partial\psi}{\partial x}$.

Данный тест позволяет в простом аналитически заданном поле имитировать случай хаотического переноса и перемешивания поля пассивного скаляра. С течением времени в таком поле будут возникать разного рода мелкие структуры. Контуры, представляющие поле скаляра, будут деформироваться. При численной реализации под действием процедуры хирургии контуры будут либо объединяться вместе, либо разбиваться на части. Это позволит нам продемонстрировать и оценить эффективность используемых процедур поиска близких участков контуров при наличии большого числа мелких структур.

Зададим начальное распределение пассивного трассера в виде однородного круглого пятна с радиусом, равным 1 (рис. 4a). В начальный момент времени выберем на контуре 100 узлов. Процедуру хирургии с последующим перераспределением узлов (по формулам из раздела 2) будем производить на каждом временно́м шаге. Во всех экспериментах шаг по времени $\Delta t = 0.05$. В качестве параметров перераспределения узлов выберем характерный масштаб $L = 1$ и параметр измельчения $\mu$ ($\mu L$ — максимально допустимое расстояние между последовательными узлами контура). Тогда масштаб отсечения $\delta$ будет рассчитываться из соотношения $\delta = \mu^2 L/4$.

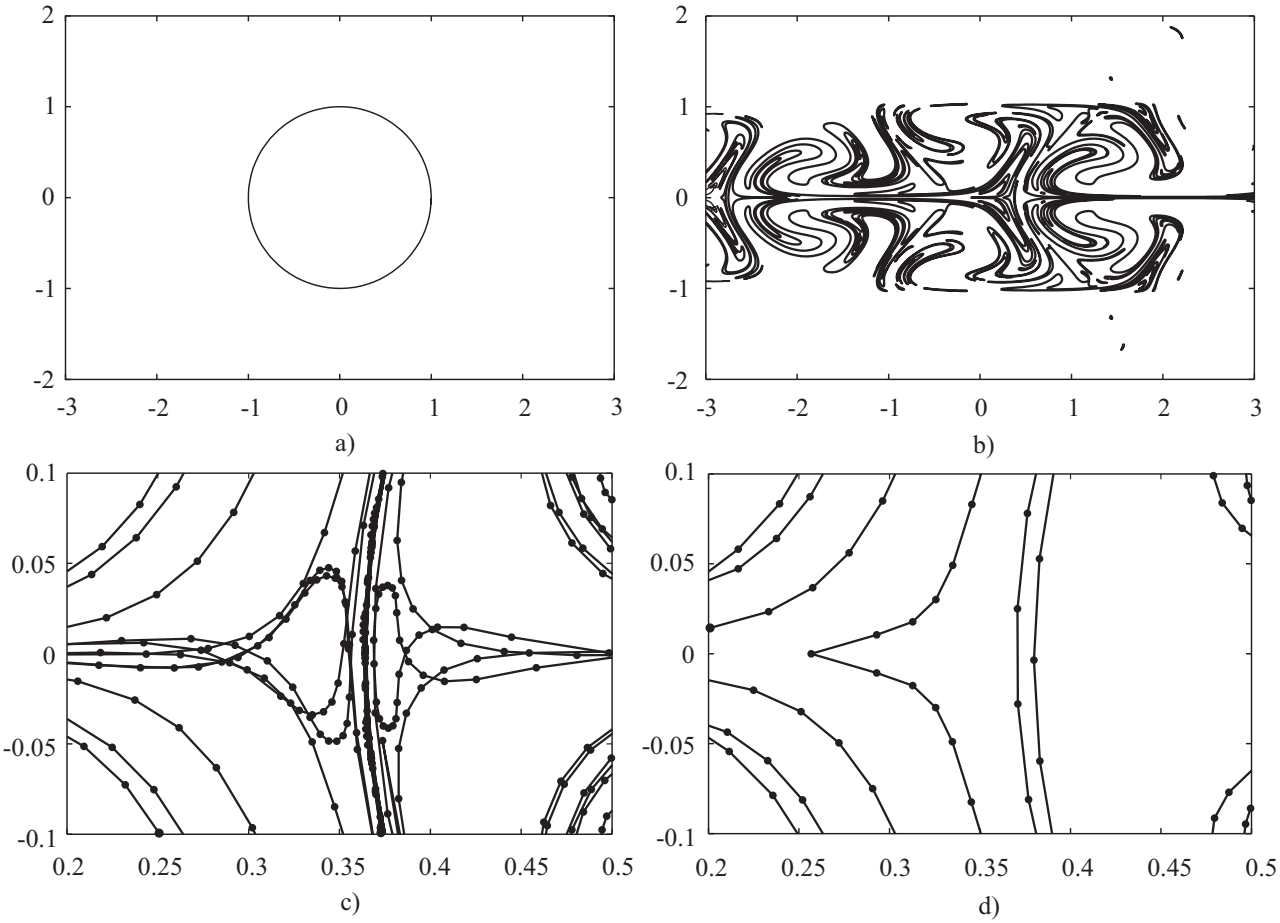


Рис. 4. На рис. a) и b) показано положение контуров в моменты времени $T = 0$ и $T = 40$ соответственно. Рис. c) и d) демонстрируют мелкие детали контуров в момент времени $T = 40$ для двух случаев: c) когда хирургия не используется, d) при включении процедуры хирургии

Результаты моделирования для $\mu = 0.1$ представлены на рис. 4b. Можно видеть наличие мелких деталей и отдельных частей контуров, которые возникли при использовании хирургии. Рис. 4c демонстрирует большое число тонких структур и пересекающихся участков контура. Одной из причин такого поведения является ограниченность пространственного разрешения контуров [1, 10], которое не позволяет адекватно описывать подобные структуры в рамках выбранной численной модели. Включение процедуры хирургии позволяет контролировать минимально допустимое разрешение на контурах путем удаления деталей, меньших установленного масштаба, что позволяет избежать самопересечений контуров (рис. 4d).

Важным свойством рассматриваемой задачи является то, что площадь однородного пятна должна сохраняться в процессе переноса [13]. Площадь, ограниченная контуром $C$, может быть рассчитана по

формуле: $s(C) = \frac{1}{2}\sum_{k=1}^{n}(x_k\, y_{k-1} - x_{k-1}\, y_k)$, где $x_0 = x_n$ и $y_0 = y_n$ ($n$ — число узлов на контуре $C$). В качестве оценки точности метода адвекции контуров с хирургией будем рассматривать относительную погрешность площади однородного пятна: $\overline{s} = \frac{s - s_0}{s_0}$, где $s_0 = \pi$, $s = \sum_{i=1}^{nc} s(C_i)$, $nc$ — число не связанных между собой контуров. Результаты при различных значениях параметра измельчения $\mu$ приведены на рис. 5.

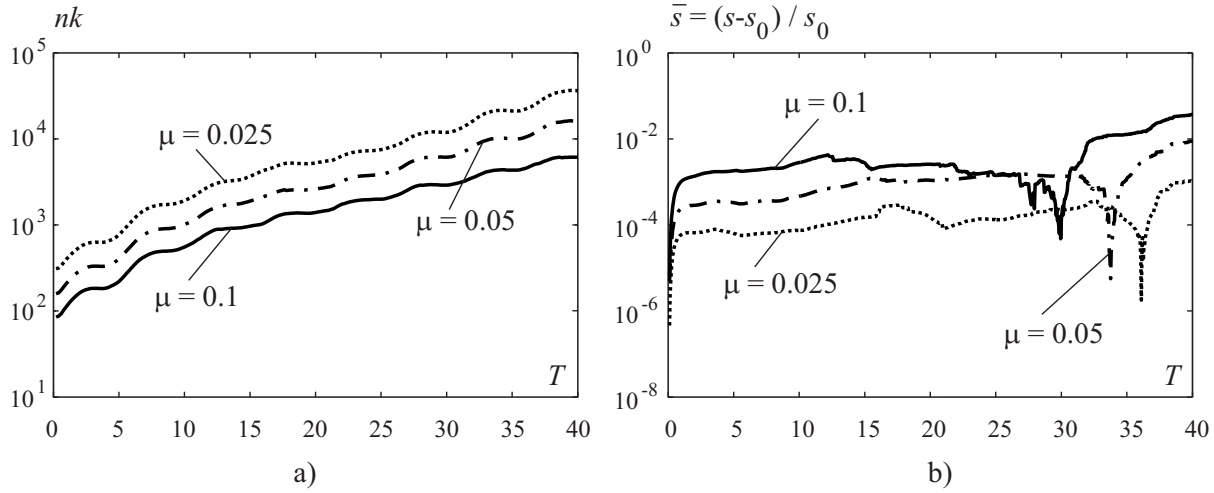


Рис. 5. Временной ход общего числа узлов $nk$ (a) и относительная погрешность площади однородного пятна $s$ (b). Показаны результаты при различных значениях параметра измельчения $\mu$

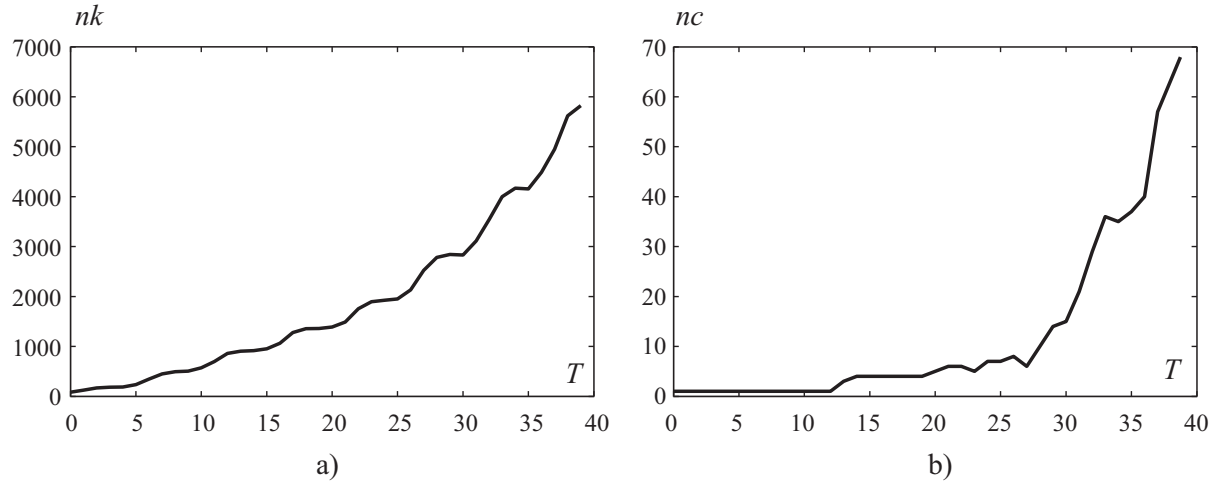


Рис. 6. Временной ход общего числа узлов $nk$ (a) и числа не связанных между собой контуров $nc$ (b). Показаны результаты для параметра измельчения $\mu = 0.1$

Во всех последующих экспериментах зафиксируем значение параметра измельчения $\mu = 0.1$. На рис. 6 приведен временной ход общего числа узлов на всех контурах и числа не связанных между собой контуров. Как можно заметить, вплоть до момента времени $T \approx 12$ число контуров ($nc$) равно одному. Затем единственный контур разбивается на части. В дальнейшем число контуров продолжает расти или убывать, что является результатом действия процедуры хирургии. Так, с момента времени $T \approx 25$ наблюдается преимущественный рост числа контуров, при этом общее число точек на всех контурах продолжает возрастать.

Проведем сравнение вычислительной эффективности рассмотренной нами схемы хирургии при использовании различных вариантов поиска близких узлов и сегментов контуров. В дальнейшем для простоты будем обозначать их следующим образом: *1* — поточечный поиск с описанными прямоугольниками;

*2* — поточечный поиск с сеточным представлением контуров; *3* — сеточный поиск. Для схем *2* и *3*, использующих пространственную сетку, зададим значения параметра $h$ (шага сетки) равным радиусу хирургии $R = \sqrt{\delta^2 + (\mu L)^2}$.

В качестве оценки эффективности будем рассматривать число пар узлов $K$, для которых проводилась проверка критерия хирургии (при использовании различных алгоритмов поиска), и отношение времени выполнения процедуры хирургии $S$ ко времени, затраченному при использовании прямого поточечного поиска (полный перебор сегментов и узлов). Результаты представлены на рис. 7.

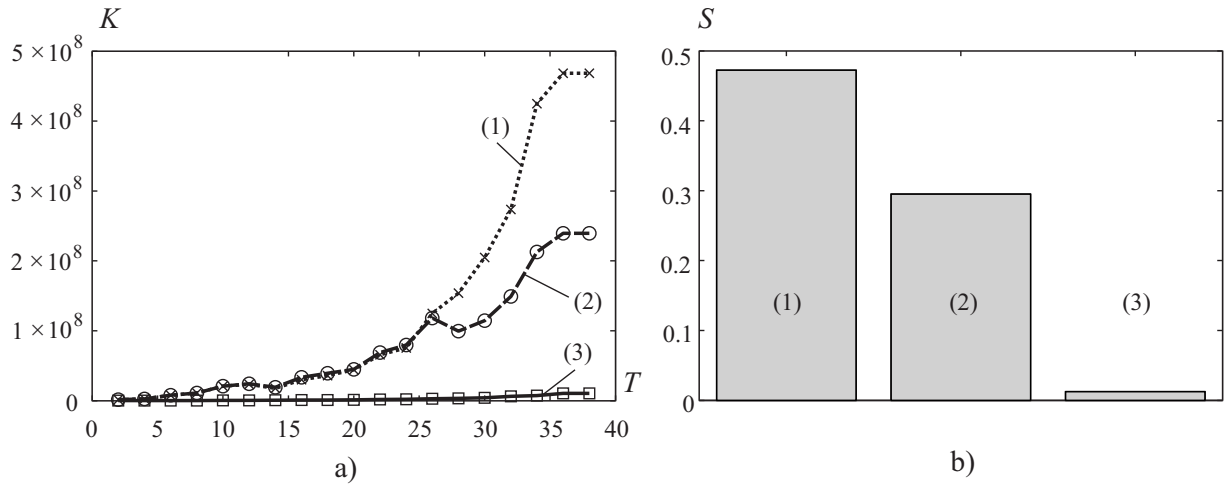


Рис. 7. Временной ход числа пар узлов $K$, для которых проводилась проверка критерия хирургии (a). Диаграмма отношения суммарного времени $S$, затраченного на выполнение процедуры хирургии, ко времени, затраченному при использовании прямого поточечного поиска (b). Показаны результаты при различных вариантах поиска для параметра измельчения $\mu = 0.1$

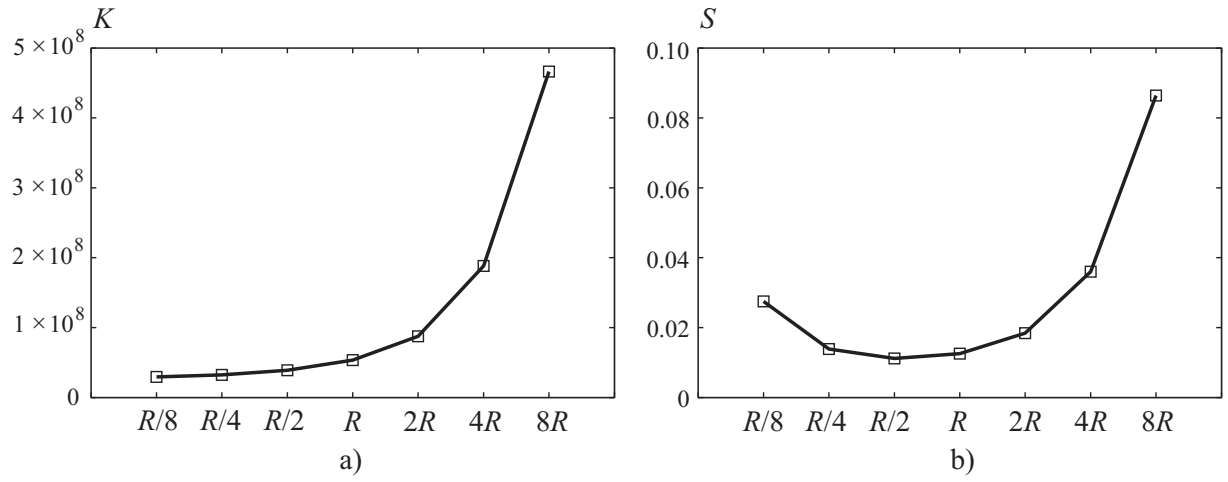


Рис. 8. Суммарное число рассмотренных пар узлов $K$, для которых проводилась проверка (a). Отношение времени $S$, затраченного на выполнение процедуры хирургии, ко времени, затраченному при использовании прямого поточечного поиска (b). Показаны результаты для алгоритма *3* при различных значениях шага хирургической сетки $h$

Как видно из рисунка, при использовании метода *3* число пар узлов $K$, для которых проводилась проверка, за все время было наименьшим, в то время как для вариантов *1* и *2*, использующих поточечный поиск, это число достаточно быстро растет в зависимости от общего количества узлов. Вследствие этого время выполнения процедуры хирургии в случае *3* значительно меньше, чем для случаев *1* и *2*.

Если проводить отдельное сравнение для методов *1* и *2*, то можно заметить, что вплоть до момента времени $T \approx 12$ число рассмотренных пар узлов у них совпадает, так как при поиске близких узлов

на одном контуре они действуют одинаково. Однако в дальнейшие моменты времени можно заметить тенденцию, согласно которой это число будет меньше для случая *2*, что обусловлено использованием более точного критерия отбора близких пар контуров. Существенное различие между ними наблюдается начиная с момента времени $T \approx 25$, когда имеет место преимущественный рост числа контуров (рис. 6).

Теперь оценим влияние шага хирургической сетки на вычислительную эффективность метода *3*. Представим шаг хирургической сетки $h$ в долях от радиуса хирургии $R$. Будем менять значение $h$ от $8R$ до $R/8$, каждый раз уменьшая его вдвое. Результаты представлены на рис. 8. Как можно видеть, уменьшение шага хирургической сетки позволяет снизить суммарное число пар узлов $K$, для которых выполняется проверка критерия хирургии. Однако при использовании $h < R/2$ выигрыш в уменьшении $K$ является незначительным, тогда как общее время выполнения процедуры хирургии $S$ начинает расти. Это, в свою очередь, обусловлено увеличением времени, затраченного на предварительную подготовку данных (заполнение ячеек хирургической сетки нулями; см. раздел 6). В качестве оптимального значения шага хирургической сетки может быть предложено приближенное равенство $h \approx R/2$. Такой выбор $h$ позволяет уменьшить общее число просмотренных узлов при минимальном числе вспомогательных операций.

**8. Заключение.** Вычислительная эффективность метода адвекции контуров в значительной степени определяется процедурой хирургии для редактирования контуров, а также формы и топологии ограничиваемых ими областей. В работе рассмотрены некоторые алгоритмические и вычислительные аспекты процедуры хирургии. Выполнен анализ отдельных ее операций и предложена новая схема ее алгоритма. Проведен обзор вариантов ускоренного поиска близких участков контуров. Предложен вариант такого поиска, ранее не использовавшийся в рамках процедуры хирургии контуров. В тестовых численных экспериментах выполнено сравнение вычислительной эффективности процедуры хирургии при различных вариантах поиска. Показано, что наилучшие результаты имеют место при использовании сеточного алгоритма поиска, для которого получены оценки оптимальных параметров алгоритма.

СПИСОК ЛИТЕРАТУРЫ


1. *Waugh D.W., Plumb R.A.* Contour advection with surgery: a technique for investigating finescale structure in tracer transport // Journal of the Atmospheric Sciences. 1994. **51**, N 4. 530–540.
2. *Dritschel D.G., Ambaum M.H.P.* A contour-advective semi-Lagrangian numerical algorithm for simulating fine-scale conservative dynamical fields // Quarterly Journal of the Royal Meteorological Society. 1997. **123**, N 540. 1097–1130.
3. *Zabusky N.J., Hughes M.H., Roberts K.V.* Contour dynamics for the Euler equations in two dimensions // Journal of Computational Physics. 1979. **30**, N 1. 96–106.
4. *Соколовский М.А., Веррон Ж.* Динамика вихревых структур в стратифицированной вращающейся жидкости. М.-Ижевск: Ижевский институт компьютерных исследований, 2011.
5. *Козлов В.Ф.* Метод контурной динамики в модельных задачах о топографическом циклогенезе в океане // Изв. АН СССР. Физика атмосферы и океана. 1983. **19**, № 8. 845–854.
6. *Козлов В.Ф.* Метод контурной динамики в океанологических исследованиях: результаты и перспективы // Морской гидрофизический журнал. 1985. № 4. 10–14.
7. *Dritschel D.G.* Contour dynamics and contour surgery: numerical algorithms for extended, high-resolution modelling of vortex dynamics in two-dimensional, inviscid, incompressible flows // Computer Physics Reports. 1989. **10**, N 3. 77–146.
8. *Dritschel D.G.* Contour Surgery: A Topological reconnection scheme for extended integrations using contour dynamics // Journal of Computational Physics. 1988. **77**, N 1. 240–266.
9. *Макаров В.Г.* Вычислительный алгоритм метода контурной динамики с изменяемой топологией исследуемых областей // Моделирование в механике. 1991. **5 (22)**, № 4. 83–95.
10. *Schaerf T.M., Macaskill C.* On contour crossings in contour-advective simulations. Part 1. Algorithm for detection and quantification // Journal of Computational Physics. 2012. **231**, N 2. 465–480.
11. *Schaerf T.M., Macaskill C.* On contour crossings in contour-advective simulations. Part 2. Analysis of crossing errors and methods for their prevention // Journal of Computational Physics. 2012. **231**, N 2. 481–504.
12. *Ласло М.* Вычислительная геометрия и компьютерная графика на C++. М.: БИНОМ, 1997.
13. *Мелешко В.В., Краснопольская Т.С.* Смешивание вязких жидкостей // Нелинейная динамика. 2005. **1**, № 1. 69–109.


Поступила в редакцию
06.06.2013