МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ (ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ) Факультет управления и прикладной математики Кафедра информатики (Специализация 010956 «Математические и информационные технологии») БАБИЧЕВ ДМИТРИЙ СЕРГЕЕВИЧ Применение оптимизационных методов для решения обратной задачи сейсморазведки МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ Научный руководитель: член-корреспондент РАН, доктор физико-математических наук, профес- сор И. Б. Петров МОСКВА – 2014 2 Содержание Стр. Список сокращений 4 ВВЕДЕНИЕ 5 0.1. Прямые задачи сейсморазведки . . . . . . . . . . . . . . . . . . . . 5 0.2. Обратные задачи сейсморазведки . . . . . . . . . . . . . . . . . . . 5 Глава 1. ОБЗОР ОСНОВНЫХ МЕТОДОВ И МОДЕЛЕЙ 8 1.1. Модель прямой задачи сейсморазведки . . . . . . . . . . . . . . . . 8 1.2. Особенности задачи . . . . . . . . . . . . . . . . . . . . . . . . . . . 8 Глава 2. МЕТОДЫ РЕШЕНИЯ ЭКСТРЕМАЛЬНЫХ ЗАДАЧ 12 2.1. Градиентные методы . . . . . . . . . . . . . . . . . . . . . . . . . . . 12 2.1.1. Метод локального градиентного спуска . . . . . . . . . . . . . . . 13 2.1.2. Метод координатного спуска . . . . . . . . . . . . . . . . . . . . . 15 2.1.3. Метод Пауэлла . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15 2.1.4. Метод сопряжённых градиентов . . . . . . . . . . . . . . . . . . . 16 2.1.5. Вероятность схождения градиентных методов к глобальному экстремуму . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17 2.2. Стохастические методы . . . . . . . . . . . . . . . . . . . . . . . . . 18 2.2.1. Метод роя частиц . . . . . . . . . . . . . . . . . . . . . . . . . . . 18 2.3. Эволюционные методы . . . . . . . . . . . . . . . . . . . . . . . . . 19 2.3.1. Метод дифференциальной эволюции . . . . . . . . . . . . . . . . 19 2.3.2. Математическая модель метода ДЭ . . . . . . . . . . . . . . . . . 22 2.3.3. Применение аппроксимации в методе дифференциальной эволю- ции . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 41 2.3.4. Нахождение прогностических точек . . . . . . . . . . . . . . . . . 44 3 2.3.5. Результаты модельных экспериментов сравнения методов ДЭ и ДЭА . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 48 Глава 3. ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ 51 3.1. Использование параллельных вычислений в методах оптимизации . 51 3.2. Сравнение практической реализации методов . . . . . . . . . . . . . 52 3.2.1. Вычислительная система, использованная для расчётов . . . . . 52 3.2.2. Метод локального градиентного спуска . . . . . . . . . . . . . . . 53 3.2.3. Градиентные методы . . . . . . . . . . . . . . . . . . . . . . . . . 54 3.2.4. Метод роя частиц . . . . . . . . . . . . . . . . . . . . . . . . . . . 54 3.3. Решение обратной задачи сейсморазведки . . . . . . . . . . . . . . . 56 3.3.1. Решение переборными методами . . . . . . . . . . . . . . . . . . . 56 3.3.2. Решение градиентными методами . . . . . . . . . . . . . . . . . . 58 3.3.3. Решение методом дифференциальной эволюции . . . . . . . . . . 58 ВЫВОДЫ 60 3.4. Заключение . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 60 СПИСОК ИСПОЛЬЗОВАННЫХ МАТЕРИАЛОВ 61 4 Список сокращений ДЭ — дифференциальная эволюция АДЭ — аппроксимированная дифференциальная эволюция ДПФ — дискретное преобразование Фурье БПФ — быстрое преобразование Фурье МРЧ — метод роя частиц СВФ — сложно вычислимая функция ЦФ — целевая функция 5 ВВЕДЕНИЕ 0.1. Прямые задачи сейсморазведки Для решения прямой задачи была использована программа tritask, раз- работанная на кафедре информатики МФТИ, использующая результаты, из- ложенные в [4], [9], [10], [13] 0.2. Обратные задачи сейсморазведки Обратные задачи сейсморазведки можно разделить на кинематические и динамические. В кинематических задачах исходными данными являются времена распространения волн и скорости среды. В динамических задачах к этим данным могут добавляться ещё тензор напряжённостей Коши или дру- гие характеристики волн, в зависимости от рассматриваемой модели. Боль- шинство таких задач являются некорректными задачами [11]. Для решения задачи используется модель двухслойной среды с наклон- ной линией раздела. Её можно решать методом отражённых волн или ме- тодом преломлённых волн. Задача в такой постановке является корректной, и почти любой примитивный метод даёт хорошие результаты. Также можно рассматривать задачу с многослойной средой, и решать её с помощью метода рефрагированных волн: после прохождения каждого слоя волна поворачива- ет на некоторый угол, и в конце концов разворачивается и выходит обратно. В такой постановке задача корректной уже не будет. Другая распространённая модель – вертикально-неоднородные среды, при этом подразумевается, что в каждом отдельном горизонтальном слое скоро- сти постоянны. В такой постановке задача была решена в [1] методом полного обращения сейсмических полей и в [2] с помощью акустического приближе- ния. В [8], [11] рассматриваются общие случаи некорректных и обратных за- дач, однако в данной работе рассматривается альтернативный способ. 6 Пусть математическая модель явления характеризуется вектором ~m, при- надлежащим некоторому пространству моделирования M. Пусть ~d означает некоторые атрибуты явления, ~d ∈ D, где D - пространство данных. [7] Име- ется некоторый оператор A, для которого ~d = A(~m), который связывает параметры модели с данными. Математическая сложность решения обратных задач заключается в том, что обратный оператор A−1 может быть либо разрывным, либо вообще не существовать. Одним из способов решения обратной задачи является переход от поиска обратного оператора к некоторой задаче минимизации: ~d = A(~m) ⇔ min ~m∈M kd − A(~m)k, где в качестве нормы может быть взята например норма L1, L2 или L∞. Таким образом решение обратной задачи сводится к многократному решению прямой задачи. Среду распространения - геологическую породу будем принимать за сплошную, линейно-упругую. Для таких сред будем использовать замкнутую система уравнений [10] относительно скоростей и напряжений:    ρv˙ = ∇ · σ σ˙ = λ(∇ · ~v)I + µ ∇ ⊗ ~v + (∇ ⊗ v) T , где ρ - плотность материала, ~v - скорость движения среды в данной точке, ∇ - градиент по пространственным координатам, σ - тензор напряжений Ко- ши, λ и µ - параметр Ляме, определяющие свойства упругого материала, I - единичный тензор Для решения прямой задачи для сред с наличием трещин используется 7 аналитическое решение задачи распространения волновых полей в породах с трещинами, основанных на уравнениях теории упругости. Особенностью ре- шения данной задачи является то, что, в отличие от классических методов, удаётся обойтись без введения эмпирических параметров. Используется усло- вие линейного проскальзывания при падении волнового фронта на трещину, что эффективно делает задачу одномерной. Следующая система уравнений:    ρ · v˙i = ∇j · σi,j , σ˙i,j = qijkl · εkl + Fij . , где σ˙ij , εkl - компоненты тензоров напряжения и деформаций, ∇i - кова- риантная производная по j-ой координате, Fij - добавочная правая часть, qijkl = λδijδkl + µ(δikδjl + δilδjk), определяет уравнения движения и реологи- ческие соотношения. 8 Глава 1 ОБЗОР ОСНОВНЫХ МЕТОДОВ И МОДЕЛЕЙ 1.1. Модель прямой задачи сейсморазведки Мы будем рассматривать модель среды с трещинами, а именно – вектор характеристик модели ~m будет содержать 5 параметров: (рисунок 1.1) h – глу- бина залегания трещин, α – угол наклона, d – расстояние между трещинами, n – количество трещин, r – длина трещин. Таким образом, ~m = (h, α, d, n, r). В качестве вектора наблюдаемых атрибутов явления будем рассматривать вертикальные или горизонтальные скорости на X приёмниках, которые рас- ставлены на расстоянии dh и показания с которых сняты через промежутки времени dt T раз: Vy(~m, xi , tj ) или Vx(~m, xi , tj ). Функционал невязки, или, другими словами, целевая функция, была вы- брана, как P = k ~d − A(~m)kL2 – сумма квадратов. Другими словами, пусть V˜ x(xi , tj ) - набор скоростей, полученных на приём- никах. Целевая функция может быть выбрана следующим образом: Px(~m) = X X i=1 X T j=1 V˜ x(xi , tj ) − Vx(~m, xi , tj ) 2 или Py(~m) = X X i=1 X T j=1 V˜ y(xi , tj ) − Vy(~m, xi , tj ) 2 1.2. Особенности задачи Решаемая задача имеет ряд особенностей: • большая область, на которой определена целевая функция; • недифференцируемость целевой функции (или по крайней мере, невоз- можность доказать её дифференцируемость); 9 r h d α n Рис. 1.1. Модель среды с трещинами с 5 параметрами • большая вероятность наличия локальных экстремумов; • целевая функция вычисляется приближёнными методами, что приво- дит к зашумлённости функции (наложению на значение целевой функ- ции стохастической функции с большим диапазоном собственных ча- стот и малой амплитудой); • значительное время вычисления целевой функции в одной точке, и соот- ветственно ещё большее время определения приближённого численного значения частных и полной производной в точке. Для определения гладкости целевой функции по узлам сетки параметров m были построены графики значений целевой функции для двух постановок задач: • Зафиксированы значения параметров r = 200, n = 7, h = 1000. Два параметра изменяются в следующих границах: 80 6 α 6 90 и 100 6 l 6 300. В качестве d были взяты данные для α = 85 и d = 200. Полученные результаты показаны на рисунке 1.2 – в координатах α, d • Зафиксированы значения параметров r = 200, h = 1000 при фиксиро- ванной длине коридора, равной 1800. Параметры α и n изменяются в следующих границах: 80 6 α 6 90, 3 6 n 6 22. В качестве d были взяты данные для α = 85, n = 12. Полученные результаты показаны на рисунке 1.3 – в координатах α, n. 10 150 200 250 80 82 84 86 88 90 150 200 250 80 82 84 86 88 90 Рис. 1.2. Целевая функция Px(~m) (слева) и Py(~m) (справа) 5 10 15 20 80 82 84 86 88 90 5 10 15 20 80 82 84 86 88 90 Рис. 1.3. Целевая функция Px(~m) (слева) и Py(~m) (справа) в случае постоянной ширины коридора 11 Если в двухмерном случае переборные алгоритмы требуют O(n 2 ), при ше- сти параметрах сложность имеет порядок O(n 6 ), что приводит к практически неразрешимой задаче. 12 Глава 2 МЕТОДЫ РЕШЕНИЯ ЭКСТРЕМАЛЬНЫХ ЗАДАЧ Основные методы поиска экстремума делятся на направленные, ненаправ- ленные и переборные. С общим представлением об этих методах, и их реали- зацией, можно ознакомиться в [16], [15], [21] 2.1. Градиентные методы Градиентные методы относятся к направленным методам. В таких мето- дах пробная точка перемещается в допустимой области значений аргумен- тов в направлении, связанном с антиградиентом. Градиентные методы хо- рошо сходятся, если целевая функция гладкая, дифференцируема на всем пространстве поиска и имеет один экстремум на пространстве поиска. При решении градиентными методами задач нахождения глобального экстрему- ма многопараметрических сложно вычисляемых функций возникают следу- ющие проблемы: • нахождение градиента функции в точке требует вычисления по мень- шей мере 2N значений функции в точке, где N-размерность простран- ства параметров; • градиентные методы имеют тенденцию сходиться к локальным мини- мумам целевых функций; • даже при наличии гладких функций градиентные методы при опреде- лённых условиях могут потребовать большого количества итераций для достижения условия сходимости; • градиентные методы чувствительны к зашумлённости значений целевой функции 13 Рис. 2.1. возможные направления движения 2.1.1. Метод локального градиентного спуска Этот метод достаточно прост. В случае функции n переменных задаётся размер шага по каждому из n направлений. 1. Выбирается пробная точка P1 2. Вычисляются значения целевой функции в каждой из 3 n − 1 соседних точек Pi , i ∈ {2 . . . 3 n} (рисунок 2.1). 3. Пусть точка Pm такая, что F(Pm) = min F(Pi), i ∈ {1 . . . 3 n} 4. Если Pm = P1 то алгоритм заканчивается. 5. Иначе P1 = Pm, переход к п. 2 В двумерном случае на каждом шаге требуется проверить 3 2 − 1 = 8 зна- чений функции. В изложенном варианте метод использует фиксированную длину шага по сетке, поэтому достаточно большая длина шага не позволяет близко подойти к требуемому экстремуму, а слишком маленькая длина шага увеличивает время работы алгоритма. Результат также существенно зави- сит от выбора начальной точки. Ещё одним недостатком (впрочем, общим для всех градиентных методов) является то, что метод склонен сходиться к локальным минимумам и для нахождения глобального минимума требуется испытать какое-то количество пробных точек. Этот метод был применён для минимизации значения двухмерной функ- ции: в выбранной модели r = 200, n = 7, h = 1000.80 6 α 6 90, 100 6 l 6 300. В качестве d были взяты данные для α = 85; l = 200. В качестве шагов по направлению были выбраны значения, равные 1/100 ча- сти от размеров рассматриваемой области оптимизации: для угла δα = 0.1, 14 150 200 250 300 82 84 86 88 90 Рис. 2.2. результаты работы метода локального градиентного спуска 1 1 1 4 4 4 1 0 1 2 3 4 5 1 1 1 2 3 4 5 2 2 2 3 5 5 Рис. 2.3. оптимизация метода локального градиентного спуска для расстояния между трещинами δd = 2. Для того, чтобы уменьшить веро- ятность схождения к локальному экстремуму, вся область оптимизации была разделена на 9 частей, центр каждой из которых был выбран в качестве на- чальной пробной точки. Результаты работы можно видеть на рисунке 2.2. Вычисление значений функции в соседних точках было оптимизировано: не вычислялись значения функции в уже просчитанных точках. Поэтому на каждом шаге вычислялось не 8 новых значений, а примерно в 2 раза меньше: рисунок 2.3. Далее рассмотрим другие градиентные методы: метод Пауэлла, метод ко- ординатного спуска, метод сопряжённых градиентов. 15 2.1.2. Метод координатного спуска В этом методе в качестве входного параметра фигурирует только началь- ная точка: (x 1 0 , x2 0 , . . . , xn 0 ) Далее, движение точки происходит по следующему алгоритму: по очереди ищется минимум одномерной функции по каждому из направлений, например по методу золотого сечения: (x 1 1 , x2 1 , . . . , xn 1 ) = arg min x1 f(x1, x2 0 , . . . , xn 0 ) (x 1 2 , x2 2 , . . . , xn 2 ) = arg min x2 f(x 1 1 , x2, x3 1 . . . , xn 1 ) . . . (x 1 n , x2 n , . . . , xn n ) = arg min xn f(x 1 n−1 , x2 n−1 , . . . , xn−1 n−1 , xn) Далее эта процедура повторяется до тех пора, пока не выполнится условие останова: либо значение функции изменилось меньше, чем на ε, либо значение точки изменилось меньше, чем на δ. Так как, вообще говоря, унимодальной не является ни сама рассматриваемая функция, ни сечения по любому аргу- менту, то метод золотого сечения может не сойтись к глобальному минимуму по направлению. Поэтому в качестве метода поиска минимума одномерной функции можно использовать и другие способы: например переборный алго- ритм с фиксированным шагом, выбор наименьшего значения функции среди нескольких случайно брошенных точек (Монте-Карло), либо метод половин- ного деления. 2.1.3. Метод Пауэлла Естественным обобщением метода координатного спуска, является метод Пауэлла. Пусть { ~h i , i ∈ {1 . . . n}} ∈ X ⊆ Rn - множество линейно-независимых векторов из X - области оптимизации, ~x0 - начальная точка. 16 1: while True do 2: for i = 1, . . . , n do 3: ~xi = ~xi−1 + λ i~h i , где λ i = arg min λi f(~xi−1 + λ i~h i ) 4: end for 5: for i = 1, . . . , n − 1 do 6: ~h i = ~h i+1 7: end for 8: ~h n = ~xn − ~x0 9: λ n = arg min λn f(~xn + λ n (~xn − ~x0 )) 10: ~x0 = ~x0 + λ n (~xn − ~x0 ) 11: end while То есть, фактически, метод Пауэлла - это метод, в котором мы опять ищем наименьшее значение функции по направлению, но на этот раз направления меняются в зависимости от значений функции. В качестве критериев остано- ва могут быть выбраны такие же, как и в методе координатного спуска. 2.1.4. Метод сопряжённых градиентов Обобщая метод Пауэлла, получаем метод сопряжённых градиентов. Для этого введём понятие сопряжённых векторов: векторы S~ 1, . . . , S~ n на- зывают сопряжёнными, если    S~T i HS~ j = 0, i 6= j, i, j = 1, . . . , n S~T i HS~ i > 0, i = 1, . . . , n , где H - матрица вторых производных (матрица Гессе) целевой функции f(~x) Формализация: Сначала задаём начальное приближение: ~x0, k = 0: 17 1: loop 2: S~j k = −∇f(~xk), ~xj k = ~xk 3: for j = 0, . . . , n − 2 do 4: λ = arg min λ f(~xj k + λS~j k ) 5: ~xj+1 k = ~xj k + λS~j k 6: ω = k∇f(~xj+1 k )k 2 k∇f(~xj k )k 2 7: S~j+1 k = −∇f(~xj+1 k ) + ωS~j k 8: end for 9: ~xk+1 = ~xj+1 k 10: k = k + 1 11: end loop 2.1.5. Вероятность схождения градиентных методов к глобально- му экстремуму При использовании градиентных методов глобальный оптимум может быть найден гарантировано только для выпуклых функций. При невыпук- лых функциях, следовательно, содержащих локальные экстремумы, отлич- ные от глобального, в множестве точек пространства параметров Vfull будет существовать некоторое количество подмножеств начальных точек, которые сходятся к локальным экстремумам. Схождения итеративного процесса гра- диентного метода требуется, чтобы начальная точка принадлежала соответ- ствующему множеству Vext. Таким образом, процесс нахождения глобального экстремума есть вероятностный процесс и вероятность нахождения глобаль- ного экстремума есть функция от количества начальных пробных точек M (условие случайности порождения начальной точки является обязательным) и отношения размера множеств pext = Vext/Vfull и равна psolve = 1 − 1 − Vext VfullM . 18 Например, если отношение pext = 0.1, то при 20 пробных точках pext = 1 − 0.9 20 = 0.888. 2.2. Стохастические методы 2.2.1. Метод роя частиц Этот метод оптимизации основан на модели роя, в каждый момент вре- мени есть некоторая популяция частиц, называемая роем, который двигается и меняется согласно некоторым простым законам. Эти перемещения удовле- творяют принципу оптимальности: если полученное положение является для частицы более выгодным, она двигается в указанном направлении. Пусть оптимизируемая функция находится в множестве G ∈ R n Пусть рой содержит N частиц, для каждой из которых определена координата ~xi и скорость ~vi . Также введём параметры: ~pi - наилучшее известное положение частицы, и ~g - наилучшее известное положение роя. Тогда алгоритм МРЧ будет выглядеть следующим образом:

for i = 1, . . . , N do ~xi ∼ U(G) 3: ~pi = ~xi if f(~pi) < f(~gi) then ~g = ~pi 6: end if ~vi ∼ U(diam(G) n ) end for 9: loop for i = 1, . . . , N do ~rg, ~rp ∼ U(0, 1) 12: ~vi = ω~vi + ϕp~rp × (~pi − ~xi) + ϕg~rg × (~g − ~xi) 19 ~xi = ~xi + ~vi if f(~xi) < f(~pi) then 15: ~pi = ~xi if f(~pi) < f(~g) then ~g = ~pi 18: end if end if end for 21: end loop Числа ω, ϕp, ϕg - являются параметрами метода. 2.3. Эволюционные методы Эволюционные методы используют элементы случайного поиска в про- странстве задачи с привнесением элементов детерминированности. Суть эво- люционных методов – отбор наилучших объектов. Время поиска экстремума по сравнению с чисто стохастическими методами может быть уменьшено на несколько порядков. Детерминированность заключается в моделировании природных процес- сов отбора, размножения и наследования, происходящих по определенным правилам. В качестве случайного элемента выступает, например, мутация. В работе [14] впервые в литературе был описан метод дифференциальной эво- люции. Относительно современное (2009 г.) представление об нейросетевых, в частности, эволюционных методах, можно получить в [17]. 2.3.1. Метод дифференциальной эволюции Генерация начальной популяции. Первый шаг: выбирается N произвольных векторов X1, X2, . . . , XN из пространства RN , в котором находится целевая функция, требующая мини- 20 мизации, называемые начальной популяцией. Их выбор обычно связан с по- становкой задачи, если известен априорный вид функции, или её некоторые свойства, либо вектора имеют равномерное или нормальное распределение. Число N является одним из параметров метода. Воспроизводство потомков. Шаг генерации нового поколения происходит: для любого вектора T из старой популяции берётся три произвольных вектора из этого же поколения: Xi , Xj , Xk, и по ним вычисляется мутантный вектор V по формуле: V = Xi+ F(Xj − Xk). Параметр F - один из параметров метода - это так называемая сила мутации. Таким образом, в методе дифференциальной эволюции (далее - ДЭ) используется внутренний источник шума – разность между случайными представителями поколения. Мутация. На следующем шаге метода с вектором V производятся мутации: вводит- ся третий параметра метода p - вероятность мутации, с которой потомок W наследует какой-либо из признаков родителя T. Фактически N раз разыгры- вается Бернуллиевская случайная величина с математическим ожиданием p: Wi = Ti с вероятностью p и Wi = Vi c вероятностью 1−p для каждого призна- ка i = 1, . . . , n. Таким образом, каждый вектор из популяции претерпевает путь T → V → W. Отбор. Он происходит на каждом шаге алгоритма. После вычисления вектора- потомка W сравниваются значения целевой функции для него и для его ро- дителя T. В новой популяции остаётся тот, на котором целевая функция принимает меньшее значение. Таким образом, размер популяции при этом не меняется, и каждый раз «выживает сильнейший», более приспособленный вектор. Разумеется, существует вероятность того, что популяцию покинет вектор, который через несколько поколений привёл бы к лучшему результа- ту. Вероятность этого уменьшает правильный выбор размера популяции. 21 Рассматриваемый метод позволяет динамически моделировать особенно- сти рельефа оптимизируемой функции. В алгоритме ДЭ источником шума является разность между случайно выбранными векторами текущей популя- ции. В случае сложного рельефа это особенно актуально, так как плотность распределения популяции выше вблизи локальных минимумов функции. По- явление точек популяции вблизи глобального минимума приводит к увели- чению вероятности миграции точек из зон локального минимума. Как и все эволюционные алгоритмы, ДЭ очень зависит от своих парамет- ров: иногда даже небольшое изменение какого-либо параметра может приве- сти к улучшению результата. Если популяция мала и время вычисления фиксировано, она успеет со- здать большое количество поколений, но вероятность схождения к локаль- ному экстремуму повышается. Слишком большой размер популяции может привести к тому, что число поколений станет недостаточным для нахожде- ния глобального экстремума. Вопрос об теоретически оптимальном размере популяции в алгоритмах с мутацией остаётся открытым. Увеличивая силу мутации F, мы увеличиваем стохастическую составля- ющую алгоритма. Локальная структура целевой функции при этом практи- чески не используется. Скорость сходимости при этом сильно уменьшается. При небольшой же силе мутации алгоритм приближается к градиентным ме- тодам, так как за счет подстройки облака точек к структуре функции он фактически строит приближение градиента. К особенностям эволюционных алгоритмов можно отнести: • сложную настройка основных параметров, от которой зависит возмож- ное вырождение, либо неустойчивость решения; • для многоэкстремальных функций – попадание в ловушки локальных экстремумов; • изолированность (поиск иголки в стоге сена). Впрочем, это является проблемой для любого метода оптимизации, так как функция не дает 22 никакой информации, подсказывающей возможную область оптимума. Так как заранее выбрать оптимальные значения размера популяции, силы мутации и параметров отбора, не представляется возможным, в настоящей работе они были выбраны из средне-рекомендуемых. Эффективность решения мы будем пытаться увеличивать в процессе ра- боты алгоритма: следя за движением популяции по полю решений. Как толь- ко популяция начала сгущаться в области некоторой точки – потенциального локального минимума, нужно оценивать предполагаемое значение этого ми- нимума по какой-нибудь интерполяционной формуле, и если оно достаточно сильно отличается от 0, то фиксируем полученную зону, и с помощью мута- ции делаем разброс этих точек, с учётом исследованных областей. 2.3.2. Математическая модель метода ДЭ Метод дифференциальной эволюции является вероятностным методом, поэтому представляется логичным ввести функцию распределения вероятно- сти, или же её плотность [18]. Для наших целей требуется определить, каким образом эта функция меняется при исполнении одного шага метода. Решим для начала следующую задачу. Пусть двумерная целевая функция f(x, y) принимает всего три различ- ных значения: f(x, y) =    0, (x, y) ∈ C0 1, (x, y) ∈ C1 2, (x, y) ∈ C2 Таким образом, вся область определения функции разбивается на три мно- жества: C0, C1, C2. Пусть теперь (x1, y1) - случайный вектор с плотностью распределения ρ1(x, y); (x2, y2) - независимый от него случайный вектор с плотностью распределения ρ2(x, y). Зададимся вопросом, какова вероятность того, что значение целевой функции на первом случайном векторе больше 23 значения целевой функции на втором. Иными словами, найдём P{f(x1, y1) > f(x2, y2)}. Для того, чтобы определить вероятность того, что значение целевой функции на первом случайном векторе равно 0, нужно проинтегрировать её плотность вероятности по области C0: P f(x1, y1) = 0 = Z C0 ρ1(x, y)dxdy. Аналогичным образом определятся вероятности P f(x1, y1) = 1 , P f(x1, y1) = 2 Заметим теперь, что P f(x1, y1) > f(x2, y2) = P f(x1, y1) = 0, f(x2, y2) = 1 + +P f(x1, y1) = 0, f(x2, y2) = 2 + P f(x1, y1) = 1, f(x2, y2) = 2 = = Z C0 Z C1 ρ1(x, y)ρ2(ˆx, yˆ)dxdydxdˆ yˆ + Z C0 Z C2 ρ1(x, y)ρ2(ˆx, yˆ)dxdydxdˆ yˆ+ + Z C1 Z C2 ρ1(x, y)ρ2(ˆx, yˆ)dxdydxdˆ yˆ Аналогичным образом вероятность ищется и для случая целевой функции, принимающей конечное число значений. Теперь перейдём к рассмотрению произвольной функции. Для этого рассмотрим представление функции в виде линий уровня: 24 Определение: множество точек LC = f(x, y) = C называется линией уровня функции f(x, y). Будем считать, что для любого C мера множества LC равна 0: у рас- сматриваемой функции нет точек, в окрестности которых она является константой. Также будем считать, что LC является кусочно-непрерывной кривой; большинство функций являются именно такими. Тогда при любом C кривую LC можно параметризовать параметром t, принадлежащим отрезку [0, 1]. Рассмотрим, например, случай квадратичной целевой функции f(x, y) = x 2 + y 2 , определённой на квадрате [−1, 1] × [−1, 1]. Тогда кривая    x(t, C) = C cos(2πt) y(t, C) = C sin(2πt) C ∈ [0, 1]; t ∈ [0, 1] является окружностью радиуса C;    x(t, C) = C cos 2πt + arccos 1 C · (2[4t] + 1 − 8t) y(t, C) = C sin 2πt + arccos 1 C · (2[4t] + 1 − 8t) , C ∈ [1, √ 2]; t ∈ [0, 1/4], где [4t] означает целую часть числа 4t, является пересечением окружности радиуса C ∈ [1, √ 2] и квадрата [−1, 1] × [−1, 1]. На рисунке 2.4 показаны линии уровня функции f(x, y) = x 2 + y 2 . Пусть Cmin и Cmax - соответственно минимальное и максимальное значе- 25 Рис. 2.4. Линии уровня квадратичной функции f(x, y) = x 2 + y 2 ние, которые функция принимает на рассматриваемой области. Тогда P{f(x1, y1) > f(x2, y2)} = = C Zmax Cmin C Zmax C ρ1(x(t, C), y(t, C)) · ρ2(x(τ, Cˆ), y(τ, Cˆ))dtdτ dCdC. ˆ Теперь перейдём непосредственно к рассмотрению самого метода диффе- ренциальной эволюции. Вероятность того, что у мутантного вектора в ре- зультате скрещивания поменялись обе координаты, равна p 2 . Плотность ве- роятности распределения такого вектора будет ρ22(x, y) = Z G Z G ρ(x−F(x2−x3), y−F(y2−y3))·ρ(x2, y2)·ρ(x3, y3)dx2dy2dx3dy3, где G - область минимизации. Вероятность того, что у мутантного вектора поменяется только координата x равна p(1 − p), и плотность вероятности в 26 этом случае равна: ρ21(x, y) = Z G Z G ρ(x − F(x2 − x3), y) · ρ(x2, y2) · ρ(x3, y3)dx2dy2dx3dy3, аналогично: ρ12(x, y) = Z G Z G ρ(x, y − F(y2 − y3) · ρ(x2, y2) · ρ(x3, y3)dx2dy2dx3dy3, ρ11(x, y) = ρ(x, y). Заметим также, что Z G ρ22(x, y)dxdy = ˆρ22 6= 1, по той причине, что, вооб- ще говоря, мутантный вектор мог получиться вне области G. Аналогичным образом вводятся ρˆ21 и ρˆ12. Таким образом, плотность вероятности пробного вектора равна: ρprob(x, y) = ((1−p) 2+(1−p)p(1−ρˆ12)+(1−p)p(1−ρˆ21)+p 2 (1−ρˆ22))·ρ(x, y)+ +(1 − p)p · ρ12(x, y) + (1 − p)p · ρ21(x, y) + p 2 · ρ22(x, y) Заметим, что выполняется условие нормировки: Z G ρprob(x, y)dxdy = = ((1−p) 2 + (1−p)p(1−ρˆ12) + (1−p)p(1−ρˆ21) +p 2 (1−ρˆ22))· Z G ρ(x, y)dxdy+ +p(1 − p) Z G ρ12(x, y)dxdy + p(1 − p) Z G ρ21(x, y)dxdy + p 2 Z G ρ22(x, y)dxdy = = ((1 − p) 2 + (1 − p)p(1 − ρˆ12) + (1 − p)p(1 − ρˆ21) + p 2 (1 − ρˆ22)) · 1+ +p(1 − p) · ρˆ12 + p(1 − p)ˆρ21 + p 2 ρˆ22 = 27 = (1 − p) 2 + 2p(1 − p) + p 2 = 1 По рассчитанной ранее формуле вероятность того, что значение функции на пробном векторе меньше, чем на исходном векторе, равна: pprob = P{f(xprob, yprob) < f(x, y)} = = C Zmax Cmin C Zmax C ρ(x(t, C), y(t, C)) · ρprob(x(τ, Cˆ), y(τ, Cˆ))dtdτ dCdC. ˆ Следовательно, новая плотность вероятности будет равна ρnew(x, y) = pprob · ρprob(x, y) + (1 − pprob) · ρ(x, y) Заметим, что и в этом случае также выполняется условие нормировки: Z G ρnew(x, y)dxdy = 1. Таким образом, мы привели расчётные формулы для изменения плотности вероятности случайно брошенной точки. Слабым местом приведённых фор- мул является их чрезмерная громоздкость, а именно - четырёхмерные инте- гралы, численное интегрирование которых с необходимой точностью требует больших вычислительных затрат. Заметим также, что полученные формулы для плотности вероятности не зависят от размера популяции N, что с пер- вого взгляда кажется не вполне логичным. Однако это легко объясняется. Пусть, например, после 100 итераций вероятность оказаться в ε-окрестности глобального минимума > 0.8: Z Oε(x0,y0) ρ(x, y)dxdy > 0.9 Тогда вероятность, того, что ни одна точка из популяции размером N = 28 5 не попадёт в эту окрестность, равна (1 − 0.8)5 = 0.00032. При размере популяции N = 10, такая вероятность равна (1 − 0.8)10 = 1.024 · 10−7 , что существенно меньше. Вероятность того, что хотя-бы 5 точек из популяции попадут в искомую окрестность, равна X 10 k=5 C k 100.8 k 0.2 10−k = 0.993631, в то время, как вероятность попадания всех пяти точек из популяции разме- ром N = 5 равна 0.8 5 = 0.32768 Для получения дальнейших результатов требуется получить произведение двух больших многочленов, которое удобно осуществлять с помощью быст- рого преобразования Фурье. Быстрое преобразование Фурье Пусть имеется два многочлена степени n: P(x) и Q(x). Зададимся вопросом: за какое количество элемен- тарных операций (сложений и умножений) можно получить их произведение. Самый простой способ - умножение каждого слагаемого первого многочлена на каждое слагаемое второго, требует O(n 2 ) операций. Алгоритм, предло- женный Карацубой [20] в 1962 году требует O(n log3 2 ) операций. Алгоритм, использующий быстрое преобразование Фурье требует O(n log n) операций, при этом при n порядка 104 алгоритм Карацубы выигрывает во времени за счёт меньшей линейной константы, а при n большего порядка по времени выигрывает быстрое преобразование Фурье. Метод быстрого преобразования Фурье основан на быстром подсчёте значения многочлена в 2n точках - за 2n log(2n) операций. Вначале подсчи- таем значения обоих многочленов на одинаковом наборе из 2n точек, затем перемножим значения функции в каждой паре и получим значения функции 29 степени 2n в 2n точках, по которым можно однозначно восстановить мно- гочлен. Обычно в качестве точек, в которых считается значение функции берут комплексные корни из единицы. Обозначим εn = εn,1 = e i 2π n - главное значение корня n-ой степени из еди- ницы. Тогда все остальные корни будут некоторой степенью главного корня: εn,k = (εn) k = e i 2πk n , k = 0, . . . , n − 1 Пусть P(x) = p0x 0+p1x 1+. . .+pn−1x n−1 . Тогда дискретным преобразованием Фурье (ДПФ) называется преобразование: DF T(p0, p1, . . . , pn−1) = (y0, y1, . . . , yn−1) = = (P(εn,0), P(εn,1), P(εn,n−1) = (P(ε 0 n ), P(ε 1 n ), . . . , P(ε n−1 n )), которое вектору коэффициентов многочлена ставит в соответствие вектор значений многочлена на наборе комплексных корней единицы. Аналогичным образом определяется обратное преобразование Фурье: InverseDF T(y0, y1, . . . , yn−1) = (p0, p1, . . . , pn−1), которое задаёт противоположное соответствие. Быстрое преобразование Фурье (БПФ) позволяет вычислить ДПФ за O(n log n). Пусть многочлен P(x) степени n, где n - степень двойки, n > 1. P(x) = p0x 0 + p1x 1 + . . . pn−1x n−1 Представим его в виде суммы двух полиномов: с чётными и с нечётными 30 степенями номерами коэффициентов: P0(x) = p0x 0 + p2x 1 + . . . pn−2x n/2−1 P1(x) = p1x 0 + p3x 1 + . . . pn−1x n/2−1 Тогда, нетрудно видеть, что: P(x) = P0(x 2 ) + xP1(x 2 ) Полученные многочлены P0 и P1 имеют вдвое меньшую степень, чем исход- ный многочлен P. Построим быстрое преобразование Фурье для многочлена P, пользуясь им для P0 и P1. Пусть {y 0 k } n/2−1 k=0 = DF T(P0) и {y 1 k } n/2−1 k=0 = DF T(P1); {yk} n−1 k=0 = DF T(P). Заметим, что тогда: yk = y 0 k + ε k n y 1 k yk+n/2 = P(ε k+n/2 n ) = P0(ε 2k+n n ) + ε k+n/2 n A1(ε 2k+n n ) = y 0 k − ε k n y 1 k Пусть T(n) - количество операций, используемых для вычисления БПФ, то- гда выполняется следующее уравнение: T(n) = 2T(n/2) + θ(n), которое имеет решение T(n) = O(n log n) [19] Непосредственной проверкой можно убедиться, что обратное преобразование Фурье имеет такую же асимптотику, так как формулы для обратных преоб- разованием будут отличаться лишь знаком и константой: yk = X n−1 j=0 pjε kj n pk = 1 n X n−1 j=0 yjε −kj n 31 ДПФ для перемножения многочленов P и Q используется следующим обра- зом: • производим прямое преобразование Фурье для исходных многочленов в 2n точках, векторы y = DF T(P) и z = DF T(Q) являются 2n-мерными. • перемножаем покомпонентно векторы y и z: t = y × z • производим обратное преобразование Фурье для вектора t: P × Q = InverseDF T(t) Заметим, что каждый шаг алгоритма занимает не более, чем O(n log n) эле- ментарных операций, значит и весь алгоритм имеет ту же асимптотику. Данный алгоритм был реализован, и была проверена его погрешность по сравнению с обычным перемножением многочленов «столбиком». Для n = 221 были сгенерированы два многочлена степени n, коэффициенты которых - случайные на отрезке [0, 1] действительные числа. Затем подсчитана норма L2 разницы полученных многочленов, которая получилась порядка 10−7 , что означает среднюю погрешность коэффициента 10−12 и является хорошим ре- зультатом. Обобщим алгоритм быстрого преобразования Фурье на случай многочле- нов, зависящих от двух переменных. Пусть P(x, y) = X n i=1 X n j=1 pi,jx i y j , где n для удобства - это некоторая степень двойки. Определим дискретное преобразование Фурье, как набор значений функции на парах чисел, каждое из которых является n-ым корнем из единицы: DF T(P(x, y)) = (P(ε i n , εj n )|i, j = 0, . . . , n − 1) = {zi,j} n−1 i,j=0 Любой многочлен можно представить в виде P(x, y) = P0(x 2 , y2 ) + xP1(x 2 , y2 ) + yP2(x 2 , y2 ) + xyP3(x 2 , y2 ) 32 Пусть {z 0 i,j} n/2−1 i,j=0 = DF T(P0(x, y)); {z 1 i,j} n/2−1 i,j=0 = DF T(P1(x, y)); {z 2 i,j} n/2−1 i,j=0 = DF T(P2(x, y)); {z 3 i,j} n/2−1 i,j=0 = DF T(P3(x, y)) Тогда выполняются следующие рекуррентные формулы:    zi,j = z 0 i,j + ε i n z 1 i,j + ε j n z 2 i,j + ε i+j n z 3 i,j zi,j+n/2 = z 0 i,j + ε i n z 1 i,j − ε j n z 2 i,j − ε i+j n z 3 i,j zi+n/2,j = z 0 i,j − ε i n z 1 i,j + ε j n z 2 i,j − ε i+j n z 3 i,j zi+n/2,j+n/2 = z 0 i,j − ε i n z 1 i,j − ε j n z 2 i,j + ε i+j n z 3 i,j Таким образом, подсчёт сложности алгоритма даёт нам следующую рекур- рентную формулу: T(n) = 4 · T(n/2) + θ(n), решением которой [19] будет асимптотика T(n) = O(n 2 log n), такой же будет и сложность всего алгоритма перемножения многочленов. Аналогичным образом можно использовать быстрое преобразование Фу- рье и для перемножение многочленов, зависящих от любого количество пе- ременных. Численное интегрирование Не умаляя общности, будем рассматри- вать функцию на квадрате K = [0, 1] × [0, 1]: С помощью линейной замены координат можно добиться, чтобы минимизируемая область G содержалась в этом квадрате; на множестве K/G определим функцию величиной, заведомо большей априорной максимальной оценки. На каждом шаге вместо плотности вероятности случайной величины будем рассматривать его сеточное приближение, а именно, пусть: ρi,j - вероятность попадания случайной величины в квадрат 33 Ki,j = [i/Napr,(i + 1)/Napr] × [j/Napr,(j + 1)/Napr], где Napr - точность разби- ения - количество частей, на которое мы делим каждую сторону квадрата. Тогда, можно утверждать, что N Xapr i=1 N Xapr j=1 ρi,j = 1 Пусть, также, fi,j - значение целевой функции в центре квадрата Ki,j Подсчитаем значение ρ new k,l - вероятность попадания случайного вектора в квадрат Ki,j после одной итерации метода. Этот могло произойти 5 способа- ми: • пробный вектор лучше исходного вектора из старого поколения, для вектора (ˆx, yˆ) из старого поколения сформировался мутантный вектор, для которого операция скрещивания не поменяла ни одной координаты • пробный вектор лучше исходного вектора из старого поколения, опера- ция скрещивания поменяла координату x • пробный вектор лучше исходного вектора из старого поколения, опера- ция скрещивания поменяла координату y • пробный вектор лучше исходного вектора из старого поколения, опера- ция скрещивания поменяла обе координаты • пробный вектор хуже исходного вектора из старого поколения, или вы- шел за границу рассматриваемой области. Остановимся сначала на первом случае. Так как значение целевой функ- ции может только уменьшаться, то для вектора T из старого поколения вы- полняется: fk,l < f(T). Таким образом, вероятность выбора вектора из ста- рого поколения, равна X ki ,li :fki ,li>fk,l ρki ,li На рисунке 2.5 изображена эволюция вектора T. Далее, формируется мутантный вектор V , независимо от вектора из старого 34 Kk,l V, W Kki,li T Рис. 2.5. первый случай: по вектору T из старого поколения строится мутантный вектор V , который равен пробному вектору W; f(W) = f(V ) < f(T) поколения. Определим вероятность того, что мутантный вектор попадёт в искомый квадрат. Она вычисляется по формуле: V = (x1 + F(x2 − x3), y1 + F(y2 − y3), где (x1, y1), (x2, y2) и (x3, y3) - три вектора из старой популяции. Пусть при этом (x1, y1) ∈ Kk1,l1 ; (x2, y2) ∈ Kk2,l2 ; (x3, y3) ∈ Kk3,l3 Тогда должно выполняться условие:    [k1 + F(k2 − k3)] = k [l1 + F(l2 − l3)] = l (∗) Искомая вероятность ρ 22 x,y будет равна: ρ 22 k,l = X выполняется (∗) ρk1,l1 · ρk2,l2 · ρk3,l3 Пусть ρˆ(x, y) = N Xapr i=1 N Xapr j=1 ρi,jx i y j , 35 Kk,l W Kk,li T Kt,l V Рис. 2.6. второй случай: по вектору T из старого поколения строится мутантный вектор V , который переходит в вектор W заменой координаты x из вектора T; f(W) < f(T) тогда ρ 22 k,l будет равно коэффициенту при x k y l многочлена ρ3(x, y) = norm[ˆρ(x, y) · ρˆ(x F , yF ) · ρˆ(x −F , y−F )], где под нормализацией (norm) подразумевается взятие целых частей от всех степеней. Например: norm[2x 2.7 y 3.4 + 3x −0.5 y 2 − 2x 3 y 4 ] = 2x 2 y 3 + 3x −1 y 2 − 2x 3 y 4 Окончательно, имеем вероятность попадания в квадрат Kk,l в первом случае: ρ 22,new k,l = p 2 ×   X ki ,li :fki ,li>fk,l ρki ,li   × coeff ρ3(x, y), k, l Перейдём теперь к рассмотрению второго случая (рисунок 2.6): Опять же, для вектора T из старого поколения выполняется: fk,l < f(T). Но на этот раз, так как координата x мутантного вектора поменялась, значит эта координата осталось прежней у вектора T из старого поколения. Таким 36 Kk,l W Kk,t V Kki,l T Рис. 2.7. третий случай: по вектору T из старого поколения строится мутантный вектор V , который переходит в вектор W заменой координаты y из вектора T; f(W) < f(T) образом, вероятность выбора вектора из старого поколения, равна X k,li :fk,li>fk,l ρk,li Так как координата x мутантного вектора поменялась, достаточно того, что- бы он попал в строку под номером l, значит ρ 2,1 k,l = N Xapr t=1 coeff ρ3(x, y), t, l Окончательно, вероятность попадания в квадрат Kk,l во втором случае есть: ρ 21,new k,l = = p(1 − p) ×   X li :fk,li>fk,l ρk,li   ×   N Xapr t=1 coeff ρ3(x, y), t, l   Аналогично рассматривается третий случай: (рисунок 2.7) ρ 12,new k,l = 37 Kk,l T, W Kki,li V Рис. 2.8. четвёртый случай: по вектору T из старого поколения строится мутантный вектор V , который переходит обратно в вектор T заменой обоих координат из вектора T; f(W) = F(T) > f(V ) = p(1 − p) ×   X ki :fki ,l>fk,l ρki ,l   ×   N Xapr t=1 coeff ρ3(x, y), k, t   В четвёртом случае вектор не поменяется, при этом мутантный вектор может иметь любые координаты, при которых значение целевой функции меньше fk,l: ρ 11,new k,l = (1 − p) 2 × ρk,l × X ki ,li :fki ,lifk,l ρki ,li , X ki ,li :fki ,li< ε и 6 ∃zk : ∃zl 6∈ z : |zk − zl | 6 ε . Заметим, что понятие зоны консенсуса с показателем ε не является эквивалентным понятию множества точек, лежащих внутри гипершара с радиусом ε и что D(z) > ε. Можно обосновать введение такого понятия, как зона консенсуса, следу- ющими соображениями. • Исходное множество разбивается на множество непересекающихся мно- жеств. • Для овражных функций можно предположить, что точки, принадлежа- щие зоне консенсуса, склонны формировать множество, близкое к дну оврага. Алгоритм разбиения исходного множества точек на множество множеств точек, принадлежащих зонам консенсуса с показателем притяжения ε можно описать следующим образом: Модификация метода заключалась в следующем: 1. После каждой итерации всё множество точек разбивалось на K непе- ресекающихся множеств точке Zi , входящих в зоны консенсуса. 43 Algorithm 1 Декомпозиция исходного множества точек на множества зон консенсуса 1: procedure ConsensusDecomposition(A, ε) 2: A : set of points 3: ε : attraction value 4: toP robe : stack of points 5: s : set of points 6: ret : set of set of points 7: A ← ∅ 8: toP robe.clear 9: while A 6= ∅ do . There are points to process 10: p ← A.f irst . p - first element in A 11: A ← A − p . Remove p from A 12: toP robe.push(p) . Add p to probe stack 13: s ← p . Current set now contain p only 14: while toP robe 6= ∅ do 15: k ← toP robe.pop . Get probe point from stack 16: for all j ∈ A do . Check all not processed points 17: if distance(k, j) 6 ε then 18: s ← s ∪ j . Add j to current set 19: A ← A − j . Remove j from A 20: toP robe.push(j) 21: end if 22: end for 23: end while 24: ret ← ret ∪ s . Add new formed set to result 25: end while 26: return ret 27: end procedure 44 2. Для каждой зоне консенсуса определялся параметр Yi , равный мини- мальному значению функций точек, входящих в данную зону Yi = min f(Pj ), ∀Pj ∈ Zi 3. Определялась зона консенсуса k, содержащая минимальное Yi : k : Yk = min i∈{1,...,K} Yi 4. Если ∃Yi : Yi < , то алгоритм завершён и в качестве решения выбира- ется та точка Ps из зоны консенсуса k, для которой 5. В каждой зоне консенсуса, содержащей более Q точек вычисляется точ- ка аппроксимированного минимума по P точкам. Если такая точка су- ществует, то она в ней вычисляется целевая функция и из множества {P + 1} выбирается P точек, имеющих наименьшие значения. Ps : f(Ps) = min i∈{1,...,N} Yi 2.3.4. Нахождение прогностических точек Вначале рассмотрим двухмерный случай: для нескольких точек (xi , yi), i = 1, N популяции требуется построить двухмерную параболу, мак- симально близкую к этим точкам. Общий вид двухмерной параболы это P2(x, y) = Ax2 + Bxy + Cy2 + Dx + Ey + F. Пусть целевая функция обозначается как f0(x, y). Пусть P2 = P2(x1, y1), . . . , P2(xn, yn) T ; f0 = f0(x1, y1), . . . ,f0(xn, yn) T Требуется минимизировать вектор P2 − f0 по некоторой норме: обычно в ка- честве таких норм берут одну из L1, L2, L∞. В данной работе мы рассмотрим 45 норму L2: Ftlq(A, B, C, D, E, F) = min A,...,F kP2 − f0kL2 = = X N i=1 Ax2 i + Bxiyi + Cy2 i + Dxi + Eyi + F − f0(xi , yi) 2 Для нахождения экстремума данной функции воспользуемся необходимым условием: ∂Ftlq ∂A = ∂Ftlq ∂B = ∂Ftlq ∂C = ∂Ftlq ∂D = ∂Ftlq ∂E = ∂Ftlq ∂A = 0 Для удобства обозначим αi,j = X N k=1 x i k y j k ; βi,j = X N k=1 x i k y j k f0(xk, yk) Тогда после несложных преобразований, необходимые условия записываются в виде системы линейных уравнений:    α4,0A + α3,1B + α2,2C + α3,0D + α2,1E + α2,0F = β2,0 α3,1A + α2,2B + α1,3C + α2,1D + α1,2E + α1,1F = β1,1 α2,2A + α1,3B + α0,4C + α1,2D + α0,3E + α0,2F = β0,2 α3,0A + α2,1B + α1,2C + α2,0D + α1,1E + α1,0F = β0,1 α2,1A + α1,2B + α0,3C + α1,1D + α0,2E + α0,1F = β1,0 α2,0A + α1,1B + α0,2C + α1,0D + α0,1E + α0,0F = β0,0 46 Таким образом требуется решить систему линейных уравнений, с положи- тельно определённой симметричной матрицей:   α4,0 α3,1 α2,2 α3,0 α2,1 α2,0 β2,0 α3,1 α2,2 α1,3 α2,1 α1,2 α1,1 β1,1 α2,2 α1,3 α0,4 α1,2 α0,3 α0,2 β0,2 α3,0 α2,1 α1,2 α2,0 α1,1 α1,0 β0,1 α2,1 α1,2 α0,3 α1,1 α0,2 α0,1 β1,0 α2,0 α1,1 α0,2 α1,0 α0,1 α0,0 β0,0   В n-мерном случае Pn(z1, . . . , zn) = X 16i6j6n Aijzizj + X 16i6n Bizi + C Ftlq(A1,1, A1,2, . . . , An,n, B1, . . . , Bn, C) = = X N k=1 X 16i6j6n Ai,jz k i z k j + X 16i6n Biz k i + C − f0(z k i , zk j ) 2 Нетрудно заметить, что ∂Ftlq ∂Aij∂Apq = X N k=1 z k p z k q z k i z k j := α p,q i,j ∂Ftlq ∂Aij∂Bp = X N k=1 z k p z k i z k j := α p i,j = α i,j p ∂Ftlq ∂Aij∂C = X N k=1 z k i z k j := αi,j = α i,j ∂Ftlq ∂Bi∂Bp = X N k=1 z k p z k i := α p i = α i p 47 ∂Ftlq ∂Bi∂C = X N k=1 z k i := αi = α i ∂Ftlq ∂ 2C = α = N Пусть, также βi,j = X N k=1 f0(z k 1 , . . . , zk n )z k i z k j βi = X N k=1 f0(z k 1 , . . . , zk n )z k i β = X N k=1 f0(z k 1 , . . . , zk n ) Таким образом оптимальные коэффициенты (A1,1, A1,2, . . . , An,n, B1, . . . , Bn, C) будут решением системы:   α 1,1 1,1 α 1,2 1,1 · · · α n,n 1,1 α 1 1,1 · · · α n 1,1 α1,1 β1,1 α 1,1 1,2 α 1,2 1,2 · · · α n,n 1,2 α 1 1,2 · · · α n 1,2 α1,2 β1,2 . . . . . . . . . . . . . . . . . . . . . . . . . . . α 1,1 n,n α 1,2 n,n · · · α n,n n,n α 1 n,n · · · α n n,n αn,n βn,n α 1,1 1 α 1,2 1 · · · α n,n 1 α 1 1 . . . αn 1 α1 β1 . . . . . . . . . . . . . . . . . . . . . . . . . . . α 1,1 n α 1,2 n · · · α n,n n α 1 n · · · α n n αn βn α 1,1 α 1,2 · · · α n,n α 1 · · · α n α β   Для тестовых задач значения параметров алгоритма следующие: • ψ = 0.2 • θ = 0.1 • ζ = 0.5 48 2.3.5. Результаты модельных экспериментов сравнения методов ДЭ и ДЭА В данном разделе сравниваются результаты испытаний классического ме- тода дифференциальной эволюции и его модификации, применяющий ап- проксимацию для генерации пробных точек в новом поколении. Модельные эксперименты проводились для размерности оптимизационной задачи, рав- ной 2-м. Оптимизационные задачи большей размерности - тема для отдель- ных исследований. Для определения критерия эффективности алгоритма введём коэффици- ент эффективности K, равный отношению количества вычисления целевой функции модифицированного и немодифицированного алгоритмов. Меньшие значения K означают лучшие результаты. 0.9 0.95 1 1.05 1.1 10 15 20 25 30 35 40 45 50 percent solved, f2 OLD NEW а) 0 200 400 600 800 1000 10 15 20 25 30 35 40 45 50 point computations, f2 OLD NEW б) Рис. 2.13. Сравнение сходимости (слева) и количества вычислений целевой функции алгоритмов ДЭ(OLD) и ДЭА(NEW) для модельной функции f2 На рис. 2.13, 2.14, 2.15, 2.16 представлены результаты модельного экспе- римента для целевых функций f2, f3, f4 и f5 соответственно. Для целевой функции f2 наблюдается улучшение сходимости метода при небольших значениях параметра N (N 6 10). Коэффициент эффективности улучшается при увеличении параметра N и достигает 0.56 для N = 50. Метод демонстрирует хороший результат. Для целевой функции f3 улучшение сходимости метода наблюдается при всех N в исследованной области значений N. Более того, метод ДЭА схо- 49 0.4 0.5 0.6 0.7 0.8 0.9 1 10 15 20 25 30 35 40 45 50 percent solved, f3 OLD NEW а) 0 200 400 600 800 1000 1200 1400 1600 10 15 20 25 30 35 40 45 50 point computations, f3 OLD NEW б) Рис. 2.14. Сравнение сходимости (слева) и количества вычислений целевой функции алгоритмов ДЭ(OLD) и ДЭА(NEW) для модельной функции f3 дится в 100% случаях уже при N > 6. Коэффициент эффективности улуч- шается при увеличении параметра N и достигает 0.29 для N = 50, то есть, количество вычислений ЦФ почти в 3.4 раза меньше, чем в метода ДЭ при значительно лучшей сходимости метода ДЭА. 0.4 0.5 0.6 0.7 0.8 0.9 1 1.1 10 15 20 25 30 35 40 45 50 percent solved, f4 OLD NEW а) 0 500 1000 1500 2000 10 15 20 25 30 35 40 45 50 point computations, f4 OLD NEW б) Рис. 2.15. Сравнение сходимости (слева) и количества вычислений целевой функции алгоритмов ДЭ(OLD) и ДЭА(NEW) для модельной функции f4 Для целевой функции f4, опять же, модифицированный метод лучше схо- дится при небольших N (N 6 12) и коэффициент эффективности K = 0.88 почти на всей области исследования. Для целевой функции f5, модифицированный метод демонстрирует нескольку лучший процент сходимости на всей области исследования и K = 0.87 для N = 50. В таб. 2.1 представлены коэффициенты эффективности алгоритма для выбранных целевых функций для размера популяции в 50 точек. Даже в наихудшем случае овражной функции метод позволяет уменьшить вычисли- 50 0 0.2 0.4 0.6 0.8 1 10 15 20 25 30 35 40 45 50 percent solved, f5 OLD NEW а) 0 1000 2000 3000 4000 5000 10 15 20 25 30 35 40 45 50 point computations, f5 OLD NEW б) Рис. 2.16. Сравнение сходимости (слева) и количества вычислений целевой функции алгоритмов ДЭ(OLD) и ДЭА(NEW) для модельной функции f5 Целевая функция f2 f3 f4 f5 Коэффициент эффективности K 0.56 0.29 0.87 0.86 Таблица 2.1 Коэффициент эффективности K модифицированного алгоритма для ряда целевых функций тельную сложность задачи на 13%. 51 Глава 3 ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ 3.1. Использование параллельных вычислений в методах оптими- зации Важным свойством решаемой задачи является то, что вычисление зна- чения целевой функции в точке существенно (на несколько порядков) пре- восходит затраты на исполнение самих алгоритмов оптимизации. Если мы модифицируем алгоритм оптимизации таким образом, что это приведёт у уменьшению количества вычисления целевой функции, то это может ока- заться боле эффективным вариантом по сравнению с немодифицированным алгоритмом. Другим важным свойством алгоритма является возможность вычисления значения целевой функции в нескольких точках одновременно. Назовём устройство вычисления целевого значения функции агентом испол- нения (это может быть вычислительное ядро или группа ядер на одном вы- числительном узле или на разных вычислительных узлах). Введём следующие обозначения: • T - количество вычислений целевой функции, необходимое для решения задачи. • n - размерность решаемой задачи. • K - коэффициент параллелизма алгоритма, который определим как отношение времени исполнения алгоритма при количестве агентов ис- полнения, равном единице, к времени исполнения этого же алгоритма при неограниченном количестве агентов исполнения. Можно различать мгновенный коэффициент параллелизма Kt и совокупный коэффици- ент параллелизма Ka, который и определяет общую производитель- ность. • N - количество точек, участвующих в процессе минимизации. 52 В метод дифференциальной эволюции на каждой итерации вычисляется ровно N значений целевой функции, которые все могут вычисляться незави- симо друг от друга. Таким образом, K = N. 3.2. Сравнение практической реализации методов 3.2.1. Вычислительная система, использованная для расчётов Серия экспериментов проводилась на следующей вычислительной систе- ме, собранной в программно реализованный кластер: 1. Intel i7-3615QM 4\*2.3/3.1GHz HyperThreading/16GB RAM 2. Intel i7-3615QM 4\*2.3/3.1GHz HyperThreading/10GB RAM 3. Intel i7-3615QM 4\*2.3/3.1GHz HyperThreading/8GB RAM 4. Intel i7-2670QM 4\*2.2/2.9GHz HyperThreading/8GB RAM 5. Intel i5-2500K 4\*3.4/4.0GHz/8GB RAM 6. Intel Core2Quadro Q6000 4\*2.4GHz/4GB RAM В качестве транспортного уровня использовался Gigabit Ethernet. На узле №1 исполнялась управляющая программа которая реализовывала основные алгоритмы оптимизации. Ввиду относительно невысокого расхода вычисли- тельных ресурсов управляющей программой, на всех узлах (№№1-6) испол- нялись программы, реализующие вычисление целевых функций. Управляю- щий узел данной GRID-системы по мере исполнения оптимизационного ал- горитма формировал очередь заданий (inputQueue) на исполнение узлами (scheduling). Исполнительные узлы получали задания у управляющего уз- ла (выданное на исполнение задание перемещается в очередь исполняемых, pendingQueue), вычисляли значение целевой функции в заданной точке и пересылали результат управляющему узлу. Управляющий узел формировал очередь исполненных заданий, completedQueue, в которую перемещались за- дания из pendingQueue и к которой имела доступ функция, реализующая оп- 53 тимизационный алгоритм. После исполнения запланированных заданий ал- горитм оптимизации в зависимости от полученных значений формировал но- вую очередь заданий и так до завершения алгоритма. Все исследуемые алго- ритмы были модифицированы для возможности параллельного исполнения. Степень возможного параллелизма определялась размером данной очереди - чем больше размер очереди запланированных заданий, тем больше возмож- ностей для исполнительных узлов получить задание. 3.2.2. Метод локального градиентного спуска Данный метод требует относительно вычисление относительно небольшо- го количества точек целевой функции. Он достаточно быстро сходится к гло- бальному оптимуму на унимодальных функциях. Поскольку этот метод является разновидностью сеточных методов, его точность и скорость схождения зависит от величины шага сетки. При боль- шом шаге сетки метод сходится достаточно быстро. Уменьшение шага сетки прямо пропорционально увеличивает параметр T. Решение тестовых задач по данному методу показало, что метод неизбеж- но сходится к ближайшим локальным минимумам. Для нахождения глобаль- ного минимума требуется значительное число пробных точек, в несколько раз превышающее количество пробных точек, достаточных для аналогичной схо- димости других методов. дело в том. что данный метод обладает свойством «близорукости», то есть он рассматривает только пробные точки в окрестно- сти текущей. В градиентных же методах применялся поиск по направлению методом золотого сечения отрезка, который мог находить новую минималь- ную пробную точку в пределах выбранного отрезка, а не только в пределах окрестности. 54 3.2.3. Градиентные методы Исследовались эксплуатационные параметры трёх градиентных методов - метода координатного спуска, при котором движение пробной точки проис- ходит строго по векторам, параллельным ортам; метод Пауэлла, при котором начальная группа из N итераций проводится по ортогональным векторам, за- тем производится корректировка базиса и следующие группы по N итераций проводится во новому базису; метод сопряжённых градиентов, при которых и первая и последующие группы итераций проводятся по базису, вычисляе- мому согласно значениям производных в пробной точке. Все методы требуют вычисления частных производных в пробных точ- ках а так же поиска оптимального значения по направлению. После ряда экспериментов на тестовых функциях было обнаружено, что метод крайне охотно сходится ко всем локальным минимумам (выбор отрезка достаточной длины для методов поиска по направлению немного сглаживал эту пробле- му) и что для увеличения вероятности решения глобальной задачи требуется значительное количество пробных точек. 3.2.4. Метод роя частиц Реализован классический вариант метода. Для приведённых тестовых функций данный метод для нахождения глобального минимума требовал вычисления значения целевых функций большего на порядки (5-70 раз) по сравнению с градиентными методами. Лучшие результаты были получены при следующих параметрах: • коэффициент скорости затухания = 0.9 • степень локального притяжения = 0.1 • степень глобального притяжения = 0 • степень притяжения элитарных точек = 1.0 • количество заменяемых наихудших точек = 1 55 0 200 400 600 800 1000 0 10 20 30 40 50 60 70 % solved, f32 CG DE Рис. 3.1. Сравнение сходимости алгоритмов сопряжённых градиентов и дифференциальной эволюции для функции f32 0 5000 10000 15000 20000 25000 0 10 20 30 40 50 60 70 point computations, f32 CG DE Рис. 3.2. Сравнение количества вычислений целевой функции алгоритмов сопряжённых градиентов и дифференциальной эволюции для функции f32 В связи с большим количеством вычисления значения пробных функций для тестовых примеров данный метод не применялся для решения основной задачи. После того, как в соревновании алгоритмов остались два - метод сопря- жённых градиентов и метод дифференциальной эволюции Из графиков видно, что при одинаковом количестве пробных точек для метода дифференциальной эволюции сходимость много выше а количество вычислений целевой функции много меньше. Например, при количестве проб- ных точек N = 30 метод ДЕ потребовал в среднем из 1000 испытаний 2308 вычислений целевой функции и сошёлся в 100.0% случаев, то метод СГ потре- 56 бовал в среднем 10240 вычислений целевой функции и сошёлся всего в 6.6% случаев. Если же за критерий взять одинаковую сходимость, то одинаковую сходимость в 71% дают метод ДЕ, который использует 11 пробных точек и требует 827 вычислений значения функции и метод СГ, который использует 500 точек и требует 221408 вычислений пробной функции. Заметим, что ты поставили методы в не совсем равные условия - функция f32 имеет 102489 локальных экстремумов на области [-600,600][-600,600] а градиентные методы склонны сходиться именно к локальным минимумам. Метод дифференциальной эволюции так же был модифицирован для дис- кретных параметров. 3.3. Решение обратной задачи сейсморазведки В качестве тестового варианта обратной задачи сейсморазазведки была выбрана задача с пятью параметрами из раздела 1.1. при следующих значе- ниях параметров: r = 250 n = 7 h = 660 α = 84.4 0 d = 100 3.3.1. Решение переборными методами Для определения гладкости целевой функции по узлам сетки параметров m были построены графики значений целевой функции для двух постановок задач: • Зафиксированы значения параметров r = 200, n = 7, h = 1000. Два параметра изменяются в следующих границах: 80 6 α 6 90 и 100 6 l 6 300. В качестве d были взяты данные для α = 85 и d = 200. Полученные результаты показаны на рисунке 3.3 – в координатах α, d 57 150 200 250 80 82 84 86 88 90 150 200 250 80 82 84 86 88 90 Рис. 3.3. Целевая функция Px(~m) (слева) и Py(~m) (справа) 5 10 15 20 80 82 84 86 88 90 5 10 15 20 80 82 84 86 88 90 Рис. 3.4. Целевая функция Px(~m) (слева) и Py(~m) (справа) в случае постоянной ширины коридора • Зафиксированы значения параметров r = 200, h = 1000 при фиксиро- ванной длине коридора, равной 1800. Параметры α и n изменяются в следующих границах: 80 6 α 6 90, 3 6 n 6 22. В качестве d были взяты данные для α = 85, n = 12. Полученные результаты показаны на рисунке 3.4 – в координатах α, n. Если в двухмерном случае переборные алгоритмы требуют O(n 2 ), при ше- сти параметрах сложность имеет порядок O(n 6 ), что приводит к практически неразрешимой задаче. 58 3.3.2. Решение градиентными методами Попытка применить метод сопряжённого градиентов для решения данной задачи к успеху не привела. При любом количестве пробных точек решение сходилось к локальным минимумам, при этом количество вычислений целе- вой функции было на несколько порядков больше (решение было остановлено после 32600 вычислений целевой функции). 3.3.3. Решение методом дифференциальной эволюции iter calc r α d h n f 4 120 212.7 85.2 100.0 530.3 6 0.000673546 8 240 248.6 85.7 116.4 656.4 7 0.000379036 12 360 248.6 85.7 116.4 656.4 7 0.000379036 16 480 253.0 84.9 100.0 660.0 7 0.000360932 20 600 253.0 84.9 100.0 660.0 7 0.000360932 24 720 250.8 85.3 100.0 678.1 7 0.000350787 28 840 252.6 84.8 100.0 671.3 7 0.000350245 32 960 252.6 84.8 100.0 671.3 7 0.000350245 36 1080 250.8 84.4 100.0 663.8 7 0.000309307 40 1200 250.8 84.4 100.0 663.8 7 0.000309307 44 1320 250.8 84.4 100.0 663.8 7 0.000309307 48 1440 250.8 84.4 100.0 663.8 7 0.000309307 139 4170 252.0 83.9 104.5 660.7 7 0.000308937 278 8340 250.6 84.1 100.9 659.0 7 0.000308730 В таблице представлен результат решения данного варианта задачи мето- дом дифференциальной эволюции на 30 точках. Колонка iter обозначает ко- личество проведённых итераций, calc - количество вычислений целевой функ- ции, f - значение целевой функции. Остальные колонки содержат значения инвертированных параметров. Видно, что уже после 48 итераций достигну- 59 то достаточно близкое решение. Дальнейшее уточнение решения требует всё более увеличивающегося количества итераций. Одной из проблем данной ре- ализации метода дифференциальной эволюции, которую предстоит разре- шить, является выбор автоматического критерия останова. 60 ВЫВОДЫ 3.4. Заключение В настоящей работе проведено исследование применимости алгоритмов многомерной оптимизации для решения одного из вариантов обратной зада- чи сейсморазведки. Алгоритмы проанализированы по критериям сходимости и возможности использования в GRID системах (многоагентных системах). Предложен вариант метода дифференциальной эволюции (метод зон консен- суса), уменьшающий вероятность схождения метода к локальным миниму- мам. Показана применимость и перспективность разработанного метода для решения задач такого типа. В частности, время решения пятимерной зада- чи оказалось приемлемым. Для данной задачи установлено, что градиент- ные методы требуют значительно большего количества вычисления целевой функции и имеют тенденцию схождения к локальным минимумам, достаточ- но далёким от глобального. Для определения оптимального критерия сходи- мости варианта метода дифференциальной эволюции, применённом в работе требуются дальнейшие исследования, как и для определения оптимальных параметров предложенного метода зон консенсуса. 61 Список литературы 1. Чеверда В. А. Восстановление скоростного строения неоднородных сред методом полного обращения волновых сейсмических полей. 2. Аникиев Д.В., Каштан Б.М., Благовещенский А.С., Мулдер В.А. Точ- ный динамический метод для решения обратной задачи сейсмики на основе интегральных уравнений Гельфанда-Левитана. 3. Сухарев А.Г., Тимохов А.В., Федоров В.В. Курс методов оптимизации. 4. Петров И.Б., Лобанов А.И. Лекции по вычислительной математике. 5. Гилл Ф., Мюррей У., Райт М. Практическая оптимизация. 6. Storn R. and Price K. Differential Evolution — A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces. 7. Жданов М.С. Геофизическая электромагнитная теория и методы. 8. Самарский А.А., Вабищевич П.Н. Численные методы решения обрат- ных задач математической физики. 9. Федоренко Р.П. Введение в вычислительную физику. 10. Квасов И.Е. Численное моделирование волновых процессов в гетеро- генных твёрдых деформируемых средах. 11. Кабанихин С.И. Обратные и некорректные задачи. 12. Васильев А.Н., Тархов Д.А. Нейросетевое моделирование. Принципы. Алгоритмы. Приложения./ СПб.:Изд-во Политехнического ун-та, 2009. - 528 с. 13. Ландау Л.Д.,Лившиц Е.М. Теоретическая физика: Учеб. пособ.: для вузов. В 10 т. Т. VII. Теория упругости.- М. ФИЗМАТЛИТ, 2007., 264с 14. Storn R. and Price K. Differential Evolution — A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces. 15. Гилл Ф., Мюррей У., Райт М. Практическая оптимизация. 16. Сухарев А.Г., Тимохов А.В., Федоров В.В. Курс методов оптимизации. 62 17. Васильев А.Н., Тархов Д.А. Нейросетевое моделирование. Принципы. Алгоритмы. Приложения./ СПб.:Изд-во Политехнического ун-та, 2009. - 528 с. 18. Натан А.А., Горбачёв О.Г., Гуз С.А. Теория вероятностей. 19. Кормен.Т, Лейзерсон.Ч, Ривест.Р, Штайн.К Алгоритмы. Построение и анализ. 20. Карацуба А., Офман Ю. Умножение многозначных чисел на автоматах // Доклады Академии Наук СССР. — 1962. — Т. 145. — № 2. 21. Полак Э. Численные методы оптимизации. Единый подход.// М. Мир, 1974, 376 с. 22. Деянов Р. Программа минимизации овражной функции многих пере- менных// Факультет ВМК МГУ им. Ломоносова/ http://www.mathlab.ru/deyanov/article/spiral-21.12.2009.pdf