Notices on applied mathematics Moscow.: CCAS, 2006. — 56 p.

The algorithms for the superposition search to the optimal regression models choice

Strijov, V., Ptashko O.
Computing Center of the Russian Academy of Sciences, Moscow, Russia
e-mail: strijov@ccas.ru

Abstract

The optimal regression model search procedure is described. The model is defined by a superposition of a smooth functions. Probability density functions of model parameters are used. The parameters are estimated with non-linear optimization methods. A problem of diesel engine pressure modelling presents an application of the method.

РОССИЙСКАЯ АКАДЕМИЯ НАУК
ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР

СООБЩЕНИЯ ПО ПРИКЛАДНОЙ МАТЕМАТИКЕ

В.В. СТРИЖОВ, Г.О. ПТАШКО

АЛГОРИТМЫ ПОИСКА СУПЕРПОЗИЦИЙ ПРИ ВЫБОРЕ ОПТИМАЛЬНЫХ РЕГРЕССИОННЫХ МОДЕЛЕЙ

ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР РАН
МОСКВА, 2006

УДК 519.6

Ответственный редактор
канд. физ.-матем. наук К. В. Воронцов

Описана процедура поиска параметрической регрессионной модели в классе моделей, определенном суперпозициями гладких функций из заданного множества. Для поиска используются оценки плотности распределения параметров элементов моделей. Параметры моделей оцениваются с помощью методов нелинейной оптимизации. Для иллюстрации приведена задача о моделировании изменения давления в камере внутреннего сгорания дизельного двигателя.

Работа поддержана грантом РФФИ 04-01-00401-а “Распознавание и прогнозирование экстремальных ситуаций в сложных системах по многомерным временным рядам наблюдений”.

Рецензенты: канд. физ.-матем. наук Ю. В. Чехович
канд. физ.-матем. наук А. Г. Дьяконов

Научное издание

© Вычислительный центр им. А. А. Дородницына РАН, 2006

1. Введение

Проблема отыскания оптимальной параметрической регрессионной модели имеет большую историю, однако остается одной их самых актуальных в области распознавания образов. А.Г. Ивахненко, еще в 1968 году, предложил метод группового учета аргументов, МГУА [8]. Согласно этому методу модель, доставляющая наилучшее приближение, отыскивается во множестве последовательно порождаемых моделей. В частности, для построения моделей как суперпозиций функций, использовались полиномиальные функции, нейронные сети и некоторые другие функции. А.Г. Ивахненко и его ученики создали ряд алгоритмов синтеза моделей и предложили методы оценки качества моделей.

При порождении конкурирующих моделей появляется задача определения значимости элементов модели. В работе К. Бишопа [9] предложен метод анализа распределения параметров однослойных нейронных сетей посредством гиперпараметров, то есть параметров распределения параметров аппроксимирующих функций. Для каждого элемента нейронной сети оценивается плотность Гауссовского распределения его параметров и делается вывод о том, насколько информативен данный элемент исследуемой регрессионной модели.

Для модификации моделей Ле Кюн предложил метод, называемый методом оптимального отсечения (optimal brain damage) [1]. Этот метод состоит в исключении некоторых, наименее информативных, элементов регрессионной модели с тем условием, что при таком исключении качество аппроксимации уменьшается незначительно. При исключении отдельных элементов модели становится возмож-

ным оценить вклад этих элементов по значениям заданной функции качества аппроксимации.

Проблема сравнения и выбора регрессионных моделей получила новое развитие после ряда публикаций Д. МакКая [4–6], предложившего при выборе модели из заданного множества использовать не информационные критерии, например AIC — Akakie Information Criterion, а двухуровневый Байесовский вывод и правило Оккама. На первом уровне вывода вычисляются плотности вероятностей распределения параметров каждой модели из заданного множества. На втором уровне вывода вычисляется правдоподобие моделей. Правило Оккама состоит в том, что вероятность выбора более сложной модели меньше, чем вероятность выбора более простой модели при сравнимом значении функции качества аппроксимации.

Метод, предлагаемый в данной работе, заключается в следующем. Поиск моделей выполняется по итерационной схеме "порождение-выбор" в соответствии с определенными правилами порождения моделей и критерием выбора моделей. Последовательно порождаются наборы конкурирующих моделей. Каждая модель в наборе является суперпозицией элементов заданного множества гладких параметрических функций. После построения модели, каждому элементу суперпозиции ставится в соответствие гиперпараметр. Параметры и гиперпараметры модели последовательно настраиваются. Из набора выбираются наилучшие модели для последующей модификации. При модификации моделей, по значениям гиперпараметров делаются выводы о целесообразности включения того или иного элемента в модель следующего порождаемого набора.

Поставим задачу нахождения регрессионной модели

нескольких свободных переменных следующим образом. Задана выборка — множество $\{\mathbf{x}_1, ..., \mathbf{x}_N | \mathbf{x} \in \mathbb{R}^M\}$ значений свободных переменных и множество $\{y_1, ..., y_N | y \in \mathbb{R}\}$ соответствующих им значений зависимой переменной. Обозначим оба эти множества как множество исходных данных D.

Также задано множество $G = \{g | g : \mathbb{R} \times ... \times \mathbb{R} \longrightarrow \mathbb{R}\}$ гладких параметрических функций $g = g(\mathbf{w}, \cdot, \cdot, ..., \cdot)$. Первый аргумент функции g — вектор-строка параметров $\mathbf{w}$, последующие — переменные из множества действительных чисел, рассматриваемые как элементы вектора свободных переменных. Рассмотрим произвольную суперпозицию, состоящую из не более чем r функций g. Эта суперпозиция задает параметрическую регрессионную модель $f = f(\mathbf{w}, \mathbf{x})$. Модель f зависит от вектора свободных переменных $\mathbf{x}$ и от вектора параметров $\mathbf{w}$. Вектор $\mathbf{w} \in \mathbb{R}^W$ состоит из присоединенных векторов-параметров функций $g_1, ..., g_r$, то есть, $\mathbf{w} = \mathbf{w}_1 \vdots \mathbf{w}_2 \vdots ... \vdots \mathbf{w}_r$, где $\vdots$ — знак присоединения векторов. Обозначим $\Phi = \{f_i\}$ — множество всех суперпозиций, индуктивно порожденное элементами множества G.

Требуется найти такую модель f_i, которая доставляет максимум функционала $p(\mathbf{w}|D, \alpha, \beta, f_i)$. Этот функционал, определяемый далее, включает искомую модель $f_i(\mathbf{w}, \mathbf{x})$ и ее дополнительные параметры α и β.

2. Выбор регрессионных моделей и гипотеза порождения данных

Общий подход к сравнению нелинейных моделей описан МакКаем [2] и заключается в следующем. Рассмотрим набор конкурирующих моделей $f_1, ..., f_M$. Априорная вероят-

ность модели f_i определена как $P(f_i)$. При появлении данных D апостериорная вероятность модели $P(f_i|D)$ может быть найдена по теореме Байеса,

$$P(f_i|D) = \frac{P(f_i)p(D|f_i)}{\sum_{i=1}^{M} p(D|f_i)P(f_i)},$$

где $p(D|f_i)$ — функция соответствия модели данным. Знаменатель дроби обеспечивает выполнение условия $\sum_{i=1}^{M} P(f_i|D) = 1$.

Вероятности моделей f_1 и f_2, параметры которых идентифицированы по данным D, сравнимы как

$$\frac{P(f_1|D)}{P(f_2|D)} = \frac{P(f_1)p(D|f_1)}{P(f_2)p(D|f_2)}. \tag{1}$$

Отношение $\frac{p(D|f_1)}{p(D|f_2)}$ — есть отношение правдоподобия моделей. Отношение $\frac{P(f_1)}{P(f_2)}$ является априорной оценкой предпочтения одной модели другой. При моделировании отдается предпочтение наиболее простым и устойчивым моделям. Но если априорные оценки $P(f_i)$ моделей одинаковы, то есть, нет причины предпочитать одну модель другой, то их необходимо сравнивать по значениям $p(D|f_i)$:

$$p(D|f_i) = \int p(D|\mathbf{w}, f_i)p(\mathbf{w}|f_i)d\mathbf{w}.$$

Апостериорная плотность распределения параметров $\mathbf{w}$ функции f_i при заданной выборке D равна

$$p(\mathbf{w}|D, f_i) = \frac{p(D|\mathbf{w}, f_i)p(\mathbf{w}|f_i)}{p(D|f_i)}, \tag{2}$$

где $p(\mathbf{w}|f_i)$ — априорно заданная плотность вероятности параметров начального приближения, $p(D|\mathbf{w}, f_i)$ — функция

правдоподобия параметров модели, а знаменатель $p(D|f_i)$ обеспечивает выполнение условия $\int p(\mathbf{w}|D, f_i)d\mathbf{w} = 1$. Он задан интегралом в пространстве параметров $\int p(\mathbf{w}'|D, f_i)p(\mathbf{w}'|f_i)d\mathbf{w}'$. Формулы (2) и (1) называются формулами Байесовского вывода первого и второго уровня.

Рассмотрим регрессию $y = f_i(\mathbf{b}, \mathbf{x}) + \nu$ с аддитивным Гауссовским шумом с дисперсией σ_ν и с нулевым матожиданием. Тогда плотность вероятности появления данных

$$p(y|x, \mathbf{w}, \beta, f_i) \triangleq p(D|\mathbf{w}, \beta, f) = \frac{\exp(-\beta E_D(D|\mathbf{w}, f_i))}{Z_D(\beta)},$$

где $\beta = \frac{1}{\sigma_\nu^2}$. Нормирующий множитель $Z_D(\beta)$ задан выражением

$$Z_D(\beta) = \left(\frac{2\pi}{\beta}\right)^{\frac{N}{2}} \tag{3}$$

и взвешенный функционал ошибки в пространстве данных

$$\beta E_D = \frac{\beta}{2}\sum_{n=1}^{N}(f_i(\mathbf{x}_n) - y_n)^2. \tag{4}$$

Введем регуляризующий параметр α, который отвечает за то, насколько хорошо модель должна соответствовать зашумленным данным. Функция плотности вероятности параметров с заданным гиперпараметром α имеет вид

$$p(\mathbf{w}|\alpha, f_i) = \frac{\exp(-\alpha E_W(\mathbf{w}|f_i))}{Z_W(\alpha)},$$

где α — обратная дисперсии распределения параметров, $\alpha = \sigma_{\mathbf{w}}^{-2}$, а нормирующая константа Z_W определена дис-

персией распределения параметров как

$$Z_W(\alpha) = \left(\frac{2\pi}{\alpha}\right)^{\frac{W}{2}}. \tag{5}$$

Требование к малым значениям параметров [5] предполагает Гауссовское априорное распределение с нулевым средним:

$$p(\mathbf{w}) = \frac{1}{Z_W} \exp\left(-\frac{\alpha}{2}||\mathbf{w}||^2\right).$$

Так как переменные α и β являются параметрами распределения параметров модели, в дальнейшем будем называть их гиперпараметрами. Исключая нормирующую константу Z_W, которая не зависит от параметров $\mathbf{w}$ и логарифмируя, получаем

$$\alpha E_W = \frac{\alpha}{2}||\mathbf{w}||^2. \tag{6}$$

Эта ошибка регуляризирует параметры, начисляя штраф за их чрезмерно большие значения.

При заданных значениях гиперпараметров α и β выражение (2) для фиксированной функции f_i будет иметь вид

$$p(\mathbf{w}|D, \alpha, \beta) = \frac{p(D|\mathbf{w}, \beta)p(\mathbf{w}|\alpha)}{p(D|\alpha, \beta)}.$$

Записывая функцию ошибки в виде $S(\mathbf{w}) = \alpha E_W + \beta E_D$, получаем

$$p(\mathbf{w}|D, \alpha, \beta, f_i) = \frac{\exp(-S(\mathbf{w}|f_i))}{Z_S(\alpha, \beta)}, \tag{7}$$

где Z_S — нормирующий множитель.

3. Нахождение параметров модели

Рассмотрим итеративный алгоритм для определения оптимальных параметров $\mathbf{w}$ и гиперпараметров α, β при заданной модели f_i. Корректный подход заключается в интегрировании всех неизвестных параметров и гиперпараметров. Апостериорное распределение параметров определяется как

$$p(\mathbf{w}|D) = \iint p(\mathbf{w}, \alpha, \beta|D) d\alpha d\beta = \iint p(\mathbf{w}|\alpha, \beta, D) p(\alpha, \beta|D) d\alpha d\beta, \tag{8}$$

что требует выполнить интегрирование апостериорного распределения параметров $p(\mathbf{w}|\alpha, \beta, D)$ по пространству, размерность которого равна количеству параметров. Вычислительная сложность этого интегрирования весьма велика. Интеграл может быть упрощен при подходящем выборе начальных значений гиперпараметров.

Приближение интеграла заключается в том, что апостериорная плотность распределения гиперпараметров $p(\alpha, \beta|D)$ имеет выраженный пик в окрестности наиболее правдоподобных значений гиперпараметров $\alpha^{\text{MP}}, \beta^{\text{MP}}$. Это приближение известно как аппроксимация Лапласа [6]. При таком допущении интеграл (8) упрощается до

$$p(\mathbf{w}|D) \approx p(\mathbf{w}|\alpha^{\text{MP}}, \beta^{\text{MP}}, D) \iint p(\alpha, \beta|D) d\alpha d\beta \approx p(\mathbf{w}|\alpha^{\text{MP}}, \beta^{\text{MP}}, D).$$

Необходимо найти значения гиперпараметров, которые оптимизируют апостериорную плотность вероятности параметров, а затем выполнить все остальные расчеты, включающие $p(\mathbf{w}|D)$ при фиксированных значениях гиперпараметров.

Для нахождения функционала $p(\mathbf{w}|\alpha, \beta, D)$, который использует апостериорное распределение параметров, рассмотрим аппроксимацию ошибки $S(\mathbf{w})$ на основе рядов Тейлора второго порядка:

$$S(\mathbf{w}) \approx S(\mathbf{w}^{\text{MP}}) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^{\text{MP}})^T A(\mathbf{w} - \mathbf{w}^{\text{MP}}). \qquad (9)$$

В выражении (9) нет слагаемого первого порядка, так как предполагается, что $\mathbf{w}^{\text{MP}}$ определяет локальный минимум функции ошибки, то есть

$$\frac{\partial S(\mathbf{w}^{\text{MP}})}{\partial w_\xi} = 0$$

для всех значений ξ. Матрица A — это матрица Гессе функции ошибок:

$$A = \nabla^2 S(\mathbf{w}^{\text{MP}}) = \beta \nabla^2 E_D(\mathbf{w}^{\text{MP}}) + \alpha I.$$

Обозначим первое слагаемое правой части через H, тогда $A = H + \alpha I$.

Подставив полученное приближенное значение $S(\mathbf{w})$ в (7) и обозначив $\Delta\mathbf{w} = \mathbf{w} - \mathbf{w}^{\text{MP}}$, получим

$$p(\mathbf{w}|\alpha, \beta, D) = \frac{1}{\hat{Z}_S} \exp\left(-S(\mathbf{w}^{\text{MP}}) - \frac{1}{2}\Delta\mathbf{w}^T A \Delta\mathbf{w}\right),$$

Оценим нормирующую константу $\hat{Z}_S$, необходимую для аппроксимации кривой Гаусса, как

$$\hat{Z}_S = \exp(-S(\mathbf{w}^{\text{MP}}))(2\pi)^{\frac{W}{2}} (\det A)^{-\frac{1}{2}}. \qquad (10)$$

Максимизируем функцию $p(D|\alpha, \beta)$, изменяя значения гиперпараметров α и β. Это можно выполнить, интегрируя

функцию плотности вероятности данных по пространству параметров $\mathbf{w}$:

$$p(D|\alpha,\beta) = \int p(D|\mathbf{w},\alpha,\beta)p(\mathbf{w}|\alpha,\beta)d\mathbf{w} = \int p(D|\mathbf{w},\alpha,\beta)p(\mathbf{w}|\alpha)d\mathbf{w}, \tag{11}$$

где второй интеграл справедлив по причине того, что распределение параметров не зависит от дисперсии шума в силу гипотезы о Гауссовском распределении шума. Для упрощения вычислений мы допускаем, что распределение $p(\alpha,\beta)$ является равномерным.

Используя (4), (6), запишем (11) в виде

$$p(D|\alpha,\beta) = \frac{1}{Z_D(\beta)}\frac{1}{Z_D(\alpha)}\int \exp(-S(\mathbf{w})))d\mathbf{w}.$$

Из (3), (5), (10) и предыдущего выражения получим

$$\ln p(D|\alpha,\beta) = -\alpha E_W^{\text{MP}} - \beta E_D^{\text{MP}} - \frac{1}{2}\ln|A| + \frac{W}{2}\ln\alpha + \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi). \tag{12}$$

Для того, чтобы оптимизировать это выражение относительно α, найдем производную

$$\frac{d}{d\alpha}\ln|A| = \frac{d}{d\alpha}\ln\left(\prod_{j=1}^{W}\lambda_j + \alpha\right) = \frac{d}{d\alpha}\sum_{j=1}^{W}\ln(\lambda_j+\alpha) = \sum_{j=1}^{W}\frac{1}{\lambda_j+\alpha} = \text{tr}(A^{-1}). \tag{13}$$

В этом выражении $\lambda_1, ..., \lambda_W$ — собственные значения матрицы H. Так как функция ошибки на данных не является квадратичной функцией параметров, как при линейной или RBF регрессии, то непосредственно оптимизировать величину α невозможно, гессиан не является константой, а зависит от параметров $\mathbf{w}$. Так как мы принимаем $A = H + \alpha I$

для вектора $\mathbf{w}^{\text{MP}}$, который зависит от выбора α, то собственные значения H косвенным образом зависят от α. Таким образом, формула (13) игнорирует переменные в выражении $d\lambda_j/\alpha$.

С использованием этого приближения, производная (12) с учетом α равна

$$\ln p(D|\alpha,\beta) = -E_W^{\text{MP}} - \frac{1}{2}\sum_{j=1}^{W}\frac{1}{\lambda_j+\alpha} + \frac{W}{2\alpha}.$$

Приравнивая последнее выражение к нулю и преобразовывая, получаем выражение для α

$$2\alpha E_W^{\text{MP}} = W - \sum_{j=1}^{W}\frac{\alpha}{\lambda_j+\alpha}. \tag{14}$$

Обозначим вычитаемое правой части через γ

$$\gamma = \sum_{j=1}^{W}\frac{\alpha}{\lambda_j+\alpha}.$$

Те компоненты суммы, в которых $\lambda_j \gg \alpha$ привносят вклад, близкий к единице, а те компоненты суммы, в которых $0 < \lambda_j \ll \alpha$, привносят вклад, близкий к нулю.

Для нахождения гиперпараметра β рассмотрим задачу оптимизации (12). Обозначим через μ_j собственные значение матрицы $\nabla^2 E_D$. Так как $H = \beta\nabla^2 E_D$, то $\lambda_j = \beta\mu_j$ и следовательно,

$$\frac{d\lambda_j}{d\beta} = \mu_j = \frac{\lambda_j}{\beta}.$$

Отсюда,

$$\frac{d}{d\beta}\ln|A| = \frac{d}{d\beta}\sum_{j=1}^{W}\ln(\lambda_j+\alpha) = \frac{1}{\beta}\sum_{j=1}^{W}\frac{\lambda_j}{\lambda_j+\alpha}.$$

Дифференцируя, как и в случае нахождения α, мы находим, что оптимальное значение β определено как

$$2\beta E_D^{\text{MP}} = N - \sum_{j=1}^{W}\frac{\lambda_j}{\lambda_j+\alpha} = N - \gamma. \tag{15}$$

Способ вычисления оптимальных значений гиперпараметров α и β описан в следующем разделе.

4. Процедура поиска оптимальной модели

Поиск оптимальной модели происходит на множестве порождаемых моделей на каждой итерации алгоритма. Перед работой алгоритма заданы множество измеряемых данных D и множество гладких функций G. Задан начальный набор конкурирующих моделей, $F_0 = \{f_1, ..., f_M | f \in \Phi\}$, в котором каждая модель f_i есть суперпозиция функций $\{g_{ij}\}_{j=1}^{r_i}$. Каждой функции g_{ij} — элементу модели f_i ставится в соответствие гиперпараметр α_{ij}, характеризующий начальную плотность распределения вектора параметров $\mathbf{b}_{ij}$ этой функции. Каждой модели f_i поставлен в соответствие гиперпараметр β_i начального приближения. Параметры i-й модели назначаются исходя из априорного распределения данных, определяемых значением β_i. Далее

выполняется последовательность шагов, приведенных ниже, которые повторяются заданное количество раз.

1. Методом сопряженных градиентов [12] минимизируются штрафные функции $S_i(\mathbf{w})$ для каждой модели $f_i, i = 1, ..., M$. Отыскиваются параметры моделей $\mathbf{w}_i^{\text{MP}}$.

2. После нахождения параметров $\mathbf{w}_i^{\text{MP}}$ определяются, исходя из (14) и (15), новые значения гиперпараметров α_{ij}^{new} и β_i^{new}. Гиперпараметр β_i функции f_i вычисляется для всего набора данных и равен

$$\beta_i^{\text{new}} = \frac{N - \gamma_i}{E_D(f_i)}.$$

Гиперпараметр α_{ij} вычисляется для каждой функции g_{ij} из суперпозиции f_i и равен

$$\alpha_{ij}^{\text{new}} = \frac{\gamma_i}{E_W(\mathbf{b}_{ij})}.$$

Здесь значения функционалов γ_i и $E_W(\mathbf{b}_{ij})$ вычисляется только для подмножества тех параметров $\mathbf{b}_{ij}$ из множества $\mathbf{w}_i$, которые являются параметрами функции g_{ij}. Изменение гиперпараметров повторяется итерационно до тех пор пока локальный минимум $S_i(\mathbf{w})$ не останется постоянным.

3. Заданы следующие правила построения производных моделей $f'_1, ..., f'_M$. Для каждой модели f_i строится производная модель f'_i. В f_i выбирается функция g_{ij} с наименьшим значением α_{ij}. Выбирается произвольная модель f_ξ из $F_0 \backslash \{f_i\}$ и ее произвольная функция $g_{\xi\zeta}$. Модель f' порождается из модели f путем замещения функции g_{ij} с ее аргументами на функцию $g_{\xi\zeta}$ с ее аргументами.

4. С заданной вероятностью η каждая модель f'_i подвергается изменениям. В изменяемой модели выбирается j-тая

функция, причем закон распределения вероятности выбора функции $p(j)$ задан. Из множества G случайным образом выбирается функция g' и замещает функцию g_j. Гиперпараметр α_{ij} этой функции определяется как $\max\limits_{j}(\alpha_{ij})$. Вектор параметров этой функции $\mathbf{b}_{ij}$ равен нулю или назначается при задании G.

5. При выборе моделей из объединенного множества родительских и порожденных моделей в соответствии с критерием $S(\mathbf{w})$ выбираются M наилучших, которые используются в дальнейших итерациях.

5. Оптимизация параметров модели

Для нахождения вектора параметров регрессионной модели мы использовали распространенные методы нелинейной оптимизации с ограничениями: метод Левенберга-Марквардта [7] и метод доверительных областей [12, 13].

Метод Левенберга-Марквардта — модификация метода Гаусса-Ньютона. Он заключается в нахождении минимума суммы квадратов нелинейных функций

$$F(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{n}(d_i(x))^2, \text{ где } d_i = y_i - f(\mathbf{x}_i).$$

Пусть $J(\mathbf{x})$ — якобиан функции $d_i(\mathbf{x})$. Тогда направление поиска задано вектором $\mathbf{s}$ — решением системы уравнений

$$(J^T J + \lambda I)\mathbf{s} = -J^T\mathbf{d},$$

где вектор λ содержит неотрицательные числа, I — единичная матрица. Для некоторого значения Δ, соответствующе-

го λ, вектор $\mathbf{s}$ является решением задачи с ограничениями

$$\frac{1}{2}||J\mathbf{s}+\mathbf{d}||_2^2,$$

при условии $||\mathbf{s}||_2 \leqslant \Delta$.

Метод доверительных областей в настоящее время является одним из самых результативных при решении прикладных задач оптимизации. Его основной идеей является аппроксимация минимизируемой функции ρ некоторой несложной функцией φ, которая адекватно описывает ρ в окрестности (доверительной области) $N_{\mathbf{b}}$ точки $\mathbf{b}$. Пробный шаг $\mathbf{s}$ вычисляется в области $N_{\mathbf{b}}$ как

$$\mathbf{s} = \arg\min\{\varphi(\mathbf{s})|\mathbf{s} \in N_{\mathbf{b}}\}.$$

Вектор начального приближения $\mathbf{w}$ заменяется вектором $\mathbf{w}+\mathbf{s}$ при выполнении $f(\mathbf{w}+\mathbf{s}) < f(\mathbf{w})$. В противном случае область N изменяется, и пробный шаг повторяется.

Квадратичная аппроксимация φ определена первыми двумя членами ряда Тейлора, приближающего ρ в точке $\mathbf{w}$; область N сферической формы. Пробный шаг

$$\mathbf{s} = \arg\min\{\frac{1}{2}\mathbf{s}^T H\mathbf{s}+\mathbf{s}^T\mathbf{g}|\ ||Ds||_2 \leqslant \Delta\},$$

где $\mathbf{g}$ — градиент ρ в точке $\mathbf{w}$, H — гессиан, D — диагональная матрица и $\frac{1}{\Delta} - \frac{1}{||\mathbf{s}||_2} = 0$.

6. Численный эксперимент

Регрессионная модель как композиция функций одного или нескольких аргументов имеет вид

$$f = g_1(\mathbf{w}_1, g_2(\mathbf{w}_2, g_3(\mathbf{w}_3, ..., ...), ...), g_r(\mathbf{w}_r)). \qquad (16)$$

Структура модели может быть как произвольной, так полностью или частично фиксированной. *Сложностью модели* $f(\mathbf{w}, \mathbf{x})$ мы считаем общее число параметров, необходимых для построения модели заданного качества, другими словами — число компонент вектора параметров $\mathbf{w}$. Это число ограничивается при решении прикладной задачи.

6.1. Модель с фиксированной структурой

Ниже описывается пример построения регрессионной модели фиксированной структуры. Объектом моделирования является кривая одной свободной переменой, представленная набором измерений. На рис. 1 сплошной кривой показаны исходные данные. По оси абсцисс отложено значение свободной переменной, по оси ординат — значение зависимой переменной. Всего выборка содержит четыре тысячи отсчетов.

Алгоритм построения модели работает следующим образом (см. рис. 2).Даны обучающая и тестовая выборки, назначенное экспертами множество G гладких функций и набор моделей начального приближения. Эти модели могут назначаться экспертами исходя из характера прикладной проблемы либо генерируется случайным образом, как это сделано в данном примере. Множество функций G, из композиции которых построены регрессионные модели, приведено в таблице 1. Оно организовано в виде набора записей, с полями:

1) порядковый номер модели,

2) имя функции в реализации Matlab,

3) начальное значение вектора параметров.

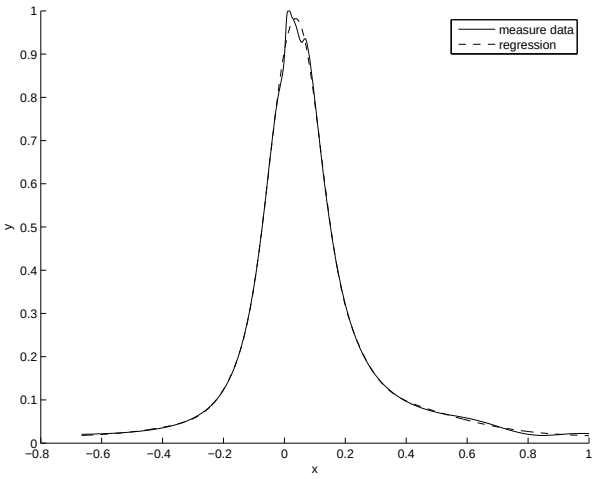


Рис. 1. Исходная и восстановленная выборка

Алгоритм использует эту базу для создания нескольких конкурирующих моделей начального приближения. Далее работа с этими моделями ведется по принципу генетического отбора: параметры моделей настраиваются оптимизационным алгоритмом, вычисляется качество моделей, выбираются наилучшие для дальнейшей модификации.

Зафиксируем структуру модели в виде линейной комбинации функций из нижней части таблицы 1, тогда модель f представима как сумма

$$f = w_0 + \sum_{i=1}^{r} g_i.$$

Обозначим параметры i-й функции из таблицы 1 через вектор-строку β_i. Тогда вектор параметров этой модели бу-

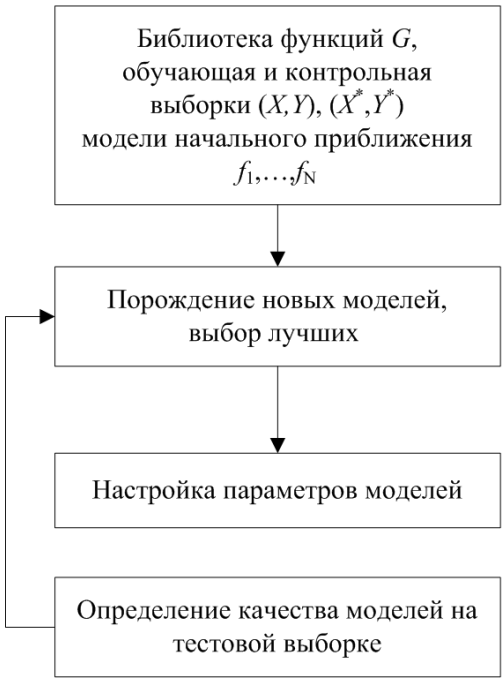


Рис. 2. Схема построения модели

дет $\mathbf{w} = [\beta_1, ..., \beta_r]^T$, а модель —

$$f = g_1(\beta_1, \mathbf{x}) + ... + g_r(\beta_r, \mathbf{x}).$$

№	Функция	Описание	Параметры
Функции двух переменных аргументов, $y = g(\beta, x_1, x_2)$			
1	plus	$y = x_1 + x_2$	–
2	minus	$y = x_1 - x_2$	–
3	times	$y = x_1 x_2$	–
4	divide	$y = \frac{x_1}{x_2}$	–
Функции одного переменного аргумента, $y = g(\beta, x)$			
5	mult	$y = ax$	a
6	add	$y = x + a$	a
7	gaussian	$y = \frac{\lambda}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\xi)^2}{2\sigma^2}\right)$	λ, σ, ξ
8	linear	$y = ax + b$	a, b
9	parabolic	$y = ax^2 + bx + c$	a, b, c
10	cubic	$y = ax^3 + bx^2 + cx + d$	a, b, c, d
11	logsig	$y = \frac{\lambda}{1+\exp(-\sigma(x-\xi))} + a$	λ, σ, ξ, a
12	tansig	$y = \frac{\lambda}{1+\exp(-2\sigma(x-\xi))} + a$	λ, σ, ξ, a

Таблица 1. Множество G базовых функций

Для выбора лучшей композиции используется генетический оптимизационный алгоритм. Он получает популяцию из N индивидов, каждый из которых соответствует своей модели. Индивид — это запись, состоящая из r полей и содержащая список функций, включенных в модель. Каждое поле содержит номер функции из G. Популяция — матрица из N строк и r столбцов. Количество строк и столбцов задаются перед запуском алгоритма. Алгоритм выполняет следующие шаги:

1) генерация начальной популяции (или получение ранее сгенерированной),

2) оценка качества каждого индивида популяции (вычисление ошибки ρ),

3) выбор индивидов для скрещивания (обмена значениями полей),

4) мутация полученных, в результате скрещивания, индивидов (случайное изменение номера функции),

5) оценка качества новых индивидов,

6) добавление новых индивидов в существующую популяцию, выбор N наилучших индивидов для дальнейших шагов,

7) переход к третьему шагу, до тех пор, пока не будет получена модель заданного качества ρ^*, либо пока не будет выполнено заданное количество шагов.

Более подробно работа алгоритма описана в [15].

Настройка параметров и оценка качества модели на рис. 1 показана в виде двух нижних блоков. Для настройки параметров алгоритм нелинейной оптимизации принимает описание модели в виде индивида и декодирует его в модель. Начальные параметры алгоритма он получает из библиотеки моделей. Алгоритм возвращает значение ошибки и настроенные параметры. Для оценки качества модели был использован функционал среднеквадратичной ошибки $\rho = \sum_{i=1}^{n}(y_i - f(x_i))^2$. Время настройки параметров одной модели в этом примере составляло около минуты, то

есть искомая модель автоматически построена в приемлемое время (один-два дня) одним компьютером.

Модель $\sum g(\beta, \cdot)$	Число параметров	$\mu(\rho_1)$	$\mu(\rho_2)$	$\mu(\rho_3)$
ggctll	19	0.009	0.002	0.021
ggctt	16	0.013	0.003	0.032
ggplll	18	0.013	0.003	0.025
gggc	13	0.015	0.003	0.035
ggg	10	0.016	0.004	0.038
ggll	13	0.016	0.004	0.033
ggn	15	0.016	0.004	0.033
ggl	10	0.023	0.006	0.045
gg	7	0.024	0.007	0.046
gp	6	0.049	0.015	0.061
Легенда: g — gaussian, n — linear, p — parabolic, c — cubic, t — tansig, l — logsig				

Таблица 2. Оценка качества моделей различной сложности

Для получения модели выборки использовались линейные комбинации из $2, ..., 6$ функций. В таблице 2 показаны несколько лучших полученных моделей. Первая колонка содержит названия включенных в модель функций; вторая колонка — число параметров используемых при настройке модели, включая веса; третья, четвертая, пятая колонки — среднюю относительную ошибку на всей выборке. Усреднение проводилось по ста реализациям кривой на всех указанных в таблице моделях

$$\mu(\rho) = \frac{1}{200} \sum_{l=1}^{200} \rho^{(l)}.$$

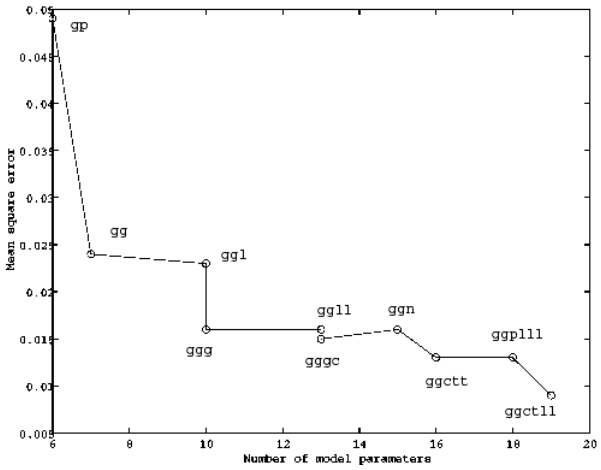


Рис. 3. Отношение сложности модели и ошибки регрессии

Первая метрика — среднеквадратичная относительная ошибка

$$\rho_1 = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - f(\mathbf{x}_i)}{\max(y_i)}\right)^2},$$

вторая — средняя относительная ошибка

$$\rho_2 = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - f(\mathbf{x}_i)|}{\max(y_i)},$$

третья — максимальная относительная ошибка

$$\rho_3 = \max_{i=1,\ldots,n}\frac{|y_i - f(\mathbf{x}_i)|}{\max(y_i)}.$$

Рисунок 3 показывает соотношение сложности и качества модели. По оси абсцисс отложено число параметров

модели, а по оси ординат — ее качество. Для дальнейшей работы экспертами была выбрана модель ggg (восстановленные регрессионные значения показаны на рис. 1 прерывистой кривой), как наиболее простая и доставляющая удовлетворительное, для данного эксперимента, качество. Ее развернутый вид

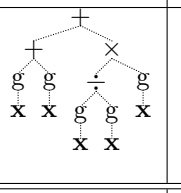
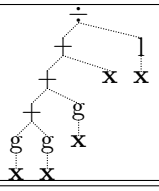
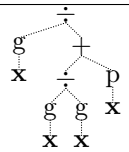
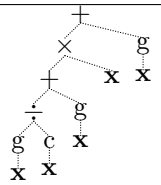
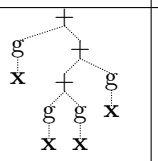
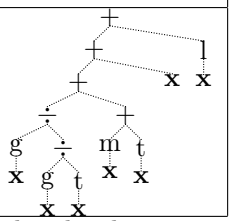
$$y = w_0 + \sum_{i=1}^{3} \frac{\lambda_i}{\sqrt{2\pi}\sigma_i} \exp\left(-\frac{(x-\xi_i)^2}{2\sigma_i^2}\right).$$

6.2. Модель со свободной структурой

Для вышеописанных данных построены модели (16) со свободной структурой. Использованы функции одного и двух аргументов из таблицы 1.

Отличие алгоритма синтеза моделей от описанного в предыдущем разделе в следующем. Каждый индивид имеет собственное число полей, равное количеству входящих в модель функций g. Каждое поле соответствует вершине t дерева, описывающего модель. Поле несет номер ι вершины t_ι, номера смежных вершин и имя функции g_ι. Операция скрещивания заключается в обмене двух индивидов поддеревьями при условии, что сложность новой модели не превосходит заданную максимальную. Операция мутации заключается в произвольном изменении имени функции одной из вершин при условии сохранения числа аргументов этой функции. В качестве моделей начального приближения были взяты четыре модели небольшой сложности, полученные в результате решения задачи синтеза моделей с фиксированной структурой.

Модели со свободной структурой имею меньшую ошибку, но в связи со сложной структурой настраиваются

№ модели	1	2	3
Число параметров	20	16	15
Описание			
№ модели	4	5	6
Число параметров	16	16	21
Описание			

Легенда: g — gaussian, p — parabolic, c — cubic, l — logsig, m — multiply, t — tansig, + — plus, × — times,÷ — divide

Таблица 3. Описание выбранных моделей

неустойчиво. При малом изменении параметров может произойти значительное изменение полученной зависимости. По этой причине итерационный алгоритмов нелинейной оптимизации может не сойтись. Для решения этой проблемы используются алгоритмы стохастической оптимизации, в частности, оптимизация отжига. Для контроля качества работы производился неоднократный запуск алгоритма нелинейной оптимизации с небольшим случайным изменением значений параметров.

В таблице 3 показаны шесть лучших моделей, полученных в результате работы алгоритма. Таблица 4 содержит

результаты настройки параметров выбранных алгоритмом моделей. Значения ошибок сравнимы со значениями, показанными в таблице 2. Из этих таблиц видно, что среднеквадратичная ошибка моделей со свободной структурой меньше или сравнима с ошибкой моделей с фиксированной структурой.

Отношение зависимости сложности модели и ее ошибки не является монотонной функцией, как на рис. 3.

№модели	Число парам.	$\mu(\rho_1)$	$\mu(\rho_2)$	$\mu(\rho_3)$
1	20	0.0025	0.0051	0.0362
2	16	0.0037	0.0053	0.0325
3	15	0.0032	0.0053	0.0355
4	16	0.0035	0.0055	0.0338
5	16	0.0034	0.0061	0.0421
6	21	0.0030	0.0063	0.0429

Таблица 4. Оценка качества моделей различной сложности

На рис. 4 показана кривая, полученная моделью № 2. Модель позволяет адекватно приблизить двойной максимум кривой, в отличие от модели представленной рисунком 1.

Развернутый вид полученной модели

$$y = (ax + b)^{-1} \left(x + \sum_{i=1}^{3} \frac{\lambda_i}{\sqrt{2\pi}\sigma_i} \exp\left(-\frac{(x - \xi_i)^2}{2\sigma_i^2} \right) \right).$$

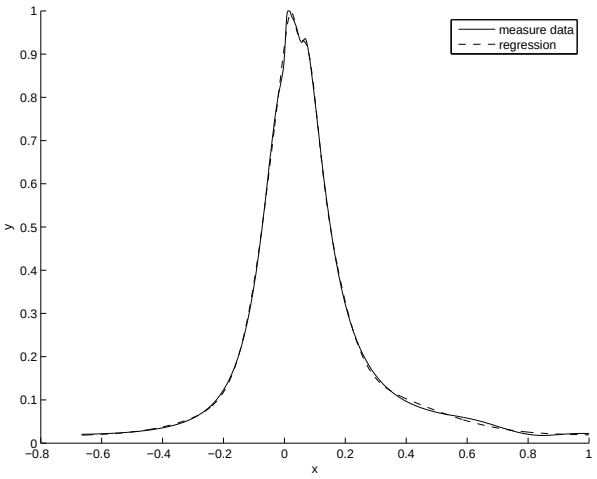


Рис. 4. Восстановленная выборка, полученная моделью № 2

7. Задача о выборе непараметрических преобразований

Организация эксперимента при поиске производных описаний в рассмотренной ниже задачи предполагает участие эксперта. Экспертом задается список G непараметрических преобразований, допустимых значений длин окон этих преобразований и задается одна из наиболее адекватных многопараметрических регрессионных моделей f.

На рис. 7. показана схема организации вычислительного эксперимента. Состояние системы, изменяющееся со временем, измеряется в моменты времени и представляется в временных рядов. Одновременно с состоянием измеряется и некоторое значение отклика системы. Для построения функции, которая с заданной точностью восста-

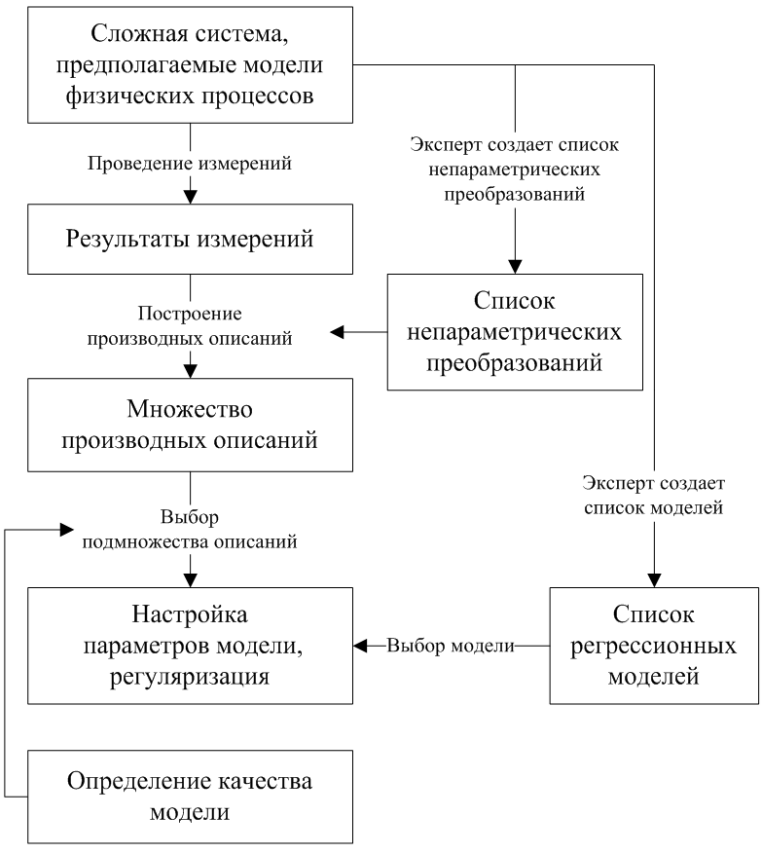


Рис. 5. Схема организации вычислительного эксперимента

навливает значение отклика, по результатам измерения состояния предлагается следующий подход. После проведения измерений описания состояний системы отображаются в производные описания. Множество отображений вместе их суперпозициями, которые также являются отображениями, задается экспертами на этапе планирования эксперимента. На рисунке это множество называется библиотекой непараметрических преобразований.

Результатом заданных отображений является расширенное описание состояния системы. Из этого множества посредством генетического оптимизационного алгоритма производится выбор подмножества из наиболее информативных описаний, т.е. тех описаний, по которым можно восстановить отклик с наименьшей ошибкой. Эти описания являются регрессорами — независимыми переменными некоторой регрессионной модели из множества моделей, заданных экспертами. На рисунке это множество называется библиотекой моделей. Построенная для решения рассматриваемой задачи библиотека ядер состоит из девяти функций, которые производят сотни новых описаний; библиотека моделей состоит из нескольких несложных многопараметрических моделей.

После выбора модели выполняется настройка и регуляризация ее параметров. По отображению контрольной выборки, в данном случае контрольного временного ряда, оценивается точность модели и оценивается качество данных. При неудовлетворительной точности модели, выбор подмножества производных описаний повторяется.

Таким образом, предлагаемый метод поиска регрессионной функции заключается в нахождении композиции двух функций: функции, которая отображает векторное про-

странство исходных описаний в пространство расширенного набора описаний, и функции, которая отображает пространство расширенного набора описаний в пространство откликов. Первая функция отыскивается в заданном множестве непараметрических преобразований, вторая функция отыскивается в заданном семействе регрессионных моделей. Для нахождения функций используются методы стохастической оптимизации.

Рассмотрим задачу восстановления регрессии с помощью фиксированной многопараметрической модели f (в частности, с помощью одной из нейросетевых моделей) и множества непараметрических преобразований $G = \{g_1, ..., g_l\}$.

Заданы временные ряды $\mathbf{x}_1, ..., \mathbf{x}_n, \mathbf{y}$ — множество последовательных элементов $\mathbf{x} = \mathbf{x}(t), \mathbf{y} = \mathbf{y}(t), t = 1, ..., m$. Временные ряды разделены, как и в задаче (**??**), на обучающую и тестовую выборки. Задано конечное множество $G = \{g | g : \mathbf{x} \longrightarrow \mathbf{z}\}$, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^m$ функций g, выполняющих непараметрическое преобразование временных рядов. Кроме временного ряда $\mathbf{x}$, аргументом функции g_i является длина скользящего окна ω_i.

Требуется из декартова произведения $X \times G$ временных рядов и преобразующих функций выбрать такое подмножество заданной мощности r, которое бы удовлетворяло условию

$$\min_{g,\mathbf{x}} \rho(\mathbf{y}^*, f[\mathbf{w}, g_1(\omega_1, \mathbf{x}^*_\cdot), ..., g_r(\omega_r, \mathbf{x}^*_\cdot)]), \tag{17}$$

где $\mathbf{w}, \omega = \arg\min \rho(\mathbf{y}, f[(\mathbf{w}, g_1(\omega_1, \mathbf{x}_\cdot), ..., g_r(\omega_r, \mathbf{x}_\cdot)])$.

Другими словами, требуется выбрать такие функции и ряды $g_i(\mathbf{x}_j)$, которые бы доставляли минимальную ошибку ρ на фиксированной модели f.

Фиксированная модель f является сложной многопараметрической моделью, для настройки параметров $\mathbf{w}$ которой требуется большое время (порядка минут, в зависимости от количества параметров и длины временных рядов). В начале вычислительного эксперимента модель выбирается из таблицы 5 заданных моделей класса нейронных сетей. Модели подробно описаны в [16]. Параметр $\mathbf{w}$ модели получается присоединением элементов. Например, для первой модели $\mathbf{w} = [b, w_1, ..., w_r]^T$. Переменные аргументы $s_1, ..., s_r$ есть элементы временных рядов $\mathbf{z}_1, ..., \mathbf{z}_r$ в момент времени t. Константа N — число нейронов в нейросетевых моделях. Функция sigm — назначаемая монотонная функция активации нейрона.

Непараметрические преобразования g приведены в таблице 6. Все преобразования, кроме g_1 и g_8 выполнялись следующим образом (подробное описание непараметрических методов см. в [17]). Назначалась длина окна ω, внутри окна проводилось преобразование и результат помещался в центральный элемент окна. Затем окно смещалось на один элемент временного ряда и операция повторялась. Первые и последние элементы временного ряда, преобразование для которых было выполнить невозможно, вычислялись при уменьшенном значении ω.

В качестве преобразования g_1 использовался метод "гусеница" (Singular Structure Analysis), описанный в [18]. Параметр ω задавал длину окна свертки. Преобразование g_8 являлось кольцевым сдвигом влево элементов временного ряда на величину ω.

Алгоритм нахождения непараметрических преобразований работает по следующей схеме.

1. Генетический алгоритм указывает номера r преобра-

Название	Описание	Параметры
Линейная модель (LS)	$f = b + \sum_{i=1}^{r} w_i s_i$	b, w
Квадратичная поверхность (QS)	$f = b + \sum_{i=1}^{r} v_i s_i + \sum_{i,j=1}^{r} w_{ij} s_i z_j$	b, v, w
Функции радиального базиса (RBF)	$f = \sum_{j=1}^{N} w_j e^{-a_j^2 \sum_{i=1}^{r} (s_i - c_{ij})^2} + b_j$	a, b, c, w
Многослойный перцептрон (MLP)	$f = \sum_{j=1}^{N} v_j \mathrm{sigm}\left(\sum_{i=1}^{r} (w_{ij} s_i + b_{ij}) \right) + a_j$	a, b, v, w

Таблица 5. Назначаемые многопараметрические модели f

№	Название	Описание
g_1	ssa	восстановленные значения [19] первой главной компоненты матрицы свертки [18] последовательных элементов временного ряда
g_2	variance	дисперсия значений элементов внутри окна
g_3	mean	среднее значение элементов внутри окна
g_4, g_5	min (max)	минимальное (максимальное) значение внутри окна
g_6	max-min	разница максимального и минимального значений внутри окна
g_7	nomean	вычет среднего значения из последовательных элементов временного ряда
g_8	shift	сдвиг по времени значений одного ряда относительно всех остальных

Таблица 6. Множество G непараметрических преобразований

зований из G и номера их аргументов $\mathbf{x}$. Создаются производные временные ряды $\mathbf{z}_1, ..., \mathbf{z}_r$. При этом длины окон ω берутся заданными и корректируются позднее.

2. Настраиваются параметры регрессионной модели f. Вычисляется ошибка ρ на обучающей выборке.

3. Корректируются значения $\omega_1, ..., \omega_r$ по заданной схеме. При проведении нижеописанного вычислительного эксперимента эти значения выбирались из множества назначенных. Уточняются параметры регрессионной модели f, происходит дообучение нейронной сети.

4. В популяции, элементами которой являются модели f, происходит обмен подструктурами с целью получения наиболее оптимального подмножества преобразований (см. описание в предыдущем разделе).

Рассмотрим результаты вычислительного эксперимента. Исходные данные — 18 временных рядов, из них одна переменная — время, шестнадцать — независимые переменные $\mathbf{x}_1, ..., \mathbf{x}_{16}$ и одна зависимая переменная $\mathbf{y}$. Каждый временной ряд содержит 1800 значений. Временной ряд по условиям эксперимента был разбит на блоки по 200 отсчетов, причем в качестве тестовой выборки использовалось 2 из 9 блоков.

В качестве параметрической регрессионной модели была назначена модель MLP из таблицы 5.

Генетический алгоритм, выбирающий наиболее информативные преобразования использовал для селекции хромосомы, несущие номера производных описаний. Каждое

поле хромосомы соответствует одному производному описанию, входящему в модель f. Поле содержит номер (i, j) описания $g_i(\omega_i, \mathbf{x}_j)$. После выполнении операций скрещивания и мутации полученные хромосомы проверяются на отсутствие повторного вхождения одного и того же производного описания.

Нижеприведенные результаты получены для $r = 5$ производных описаний. Остальные переменные не оказали существенного влияния на качество модели, т.е. при включении дополнительной переменной ошибка уменьшалась незначительно.

На рисунке 6 показаны временные ряды выбранных переменных, измеренных при проведении эксперимента. По оси абсцисс отложено время.

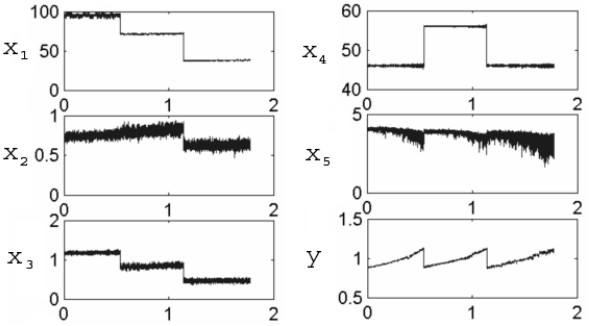


Рис. 6. Выбранные переменные и зависимая переменная

Таблица 7 содержит список выбранных непараметрических преобразований, которые применялись к исходным временным рядам. Значения ω длин окон настраивались простым перебором наиболее вероятных для данного эксперимента значений.

Переменная	Преобразование	Длина окна ω
$\mathbf{x}_1$	ssa	70
$\mathbf{x}_2$	variance	6
$\mathbf{x}_3$	mean	12
$\mathbf{x}_4$	min	6
$\mathbf{x}_5$	variance	6
$\mathbf{y}$	shift	6

Таблица 7. Выбранные переменные и их преобразования

В качестве примера настройки величины сдвига ω_8 производного описания $g_8(\mathbf{x})$ приведем рисунок 7. При изменении значения ω_8 от 1 до 14 значение ошибки измерялось более чем в два раза. Наименьшая ошибка достигается при значении сдвига равном 6.

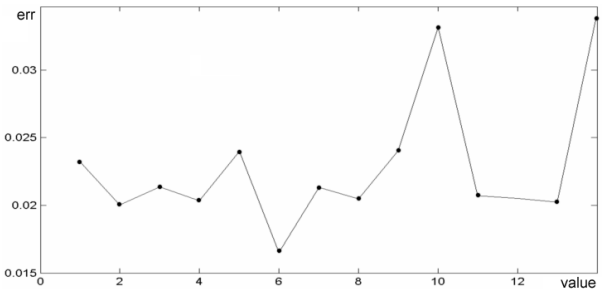


Рис. 7. Зависимость ошибки ρ модели от параметра преобразования

8. Программный комплекс

Для практического применения решения описанной задачи был создан программный комплекс в системе Matlab 7.0. Комплекс разбит на несколько программных модулей. Каждый модуль представлен каталогом в файловой системе. На данный момент комплекс представлен двумя модулями: `code` и `func`. Модуль `code` содержит основные функции для работы с моделями в рамках описанного решения. Модуль `func` содержит функции-компоненты генерируемых регрессионных моделей. В качестве примера работы с программным комплексом можно ознакомиться с Matlab-файлом `start.m`, который находится в корне дерева каталогов программного комплекса.
Далее подробно описывается содержимое каждого модуля.

8.1. Модуль code

Модуль содержит функции для работы с моделями регресии. Сюда входят функции для ввода моделей и функций-компонент из внешних файлов, функции для оптимизации параметров моделей и другие.

1. Matlab-функция: `composer5`
   Файл: `composer5.m`.
   Описание: Функция строит модели регресии, имеющие аддитивную структуру, используя описанные выше методы
   Параметры:

   - `x` - значения независимой переменной
   - `y` - значения зависимой переменной

- `wghts` - веса зависимой переменной
- `registry` - множество функций-компонент
- `mdlpop` - начальная популяция моделей
- `options` - дополнительные опции оптимизации

Функция возвращает:

- функция выводит на экран различную информацию о найденых моделях

2. Matlab-функция: `gradient5`
   Файл: `gradient5.m`.
   Описание: Функция возвращает аппроксимацию градиента модели, заданной в виде функции, в заданных точках.
   Параметры:

   - `f` - модель в виде Matlab-функции
   - `b` - вектор параметров модели
   - `x` - значения независимой переменной

   Функция возвращает:

   - `g` - аппроксимация градиента

3. Matlab-функция: `hessian5`
   Файл: `hessian5.m`.
   Описание: Функция возвращает аппроксимацию гессиана функции ошибок, заданной, как $\sum(f(\mathbf{b}, x)-y)^2$.
   Параметры:

   - `f` - модель в виде Matlab-функции

- `b` - вектор параметров модели
- `x` - значения независимой переменной
- `y` - значения зависимой переменной

Функция возвращает:

- `H` - аппроксимация гессиана функции ошибок

4. Matlab-функция: `newton5`
   Файл: `newton5.m`.
   Описание: Функция производит настройку заданной модели (нелинейной регресии), используя Matlab-функцию `nlinfit`.
   Параметры:

   - `x` - значения независимой переменной
   - `y` - значения зависимой переменной
   - `w` - веса зависимой переменной
   - `func` - модель в виде Matlab-функции
   - `b0` - начальный вектор параметров модели

   Функция возвращает:

   - `[yerr,b1,y1]` - сумма квадратов отклонений зависимой переменной от значений модели после оптимизации, вектор параметров, значения модели после процесса настройки

5. Matlab-функция: `nlinfit5`
   Файл: `nlinfit5.m`.
   Описание: Функция производит настройку заданной

модели (нелинейной регресии), используя Matlab-функцию `nlinfit`. В качестве функции ошибок используется функция $\frac{\beta}{2}\sum(y - f(\mathbf{b}, x))^2 + \frac{\alpha}{2}\sum \mathbf{b}^2$ с двумя регуляризационными параметрами α и β.
Параметры:

- `X` - значения независимой переменной
- `y` - значения зависимой переменной
- `model` - модель в виде Matlab-функции
- `beta` - начальный вектор параметров модели
- `alp` - регуляризационный параметр α
- `bet` - регуляризационный параметр β
- `options` - дополнительные опции оптимизации (см. Matlab-функцию `nlinfit`)

Функция возвращает:

- `[beta,r,J]` - вектор настроенных параметров, отклонения зависимой переменной от значений модели после оптимизации, якобиан модели при значениях вектора параметров `beta` (см. Matlab-функцию `nlinfit`)

6. Matlab-функция: `func2mdl`
   Файл: `func2mdl.m`.
   Описание: Функция создает сетевую структуру (в виде дерева) регрессионной модели на основе Matlab-функции, заданной текстовой строкой. Более подробный формат задания модели в виде текстовой строки можно посмотреть в комментариях к Matlab-функции

`model_grabber`.
Параметры:

- `func` - модель в виде Matlab-функции, представленная текстовой строкой
- `b` - вектор параметров модели

Функция возвращает:

- `n` - сетевую структуру модели в виде дерева

7. Matlab-функция: `mdl2func`
   Файл: `mdl2func.m`.
   Описание: Функция создает модель в виде Matlab-функции, представленной текстовой строкой на основе заданной сетевой структуры регрессионной модели. Более подробный формат задания модели в виде текстовой строки можно посмотреть в комментариях к Matlab-функции `model_grabber`.
   Параметры:

   - `n` - сетевая структура модели в виде дерева

   Функция возвращает:

   - `[func,b]` - модель в виде Matlab-функции, представленная текстовой строкой и вектор параметров модели

8. Matlab-функция: `model_grabber`
   Файл: `model_grabber.m`.
   Описание: Функция создает Matlab-функцию на основе заданного текстового описания модели и заданного множества функций-компонент. Более подробный

формат задания модели в виде текстовой строки можно посмотреть в комментариях к Matlab-функции.
Параметры:

- `model_input` - модель в виде текстовой строки
- `registry` - множество функций-компонент

Функция возвращает:

- `[model_output,parcounter]` - модель в виде Matlab-функции и число параметров модели

9. Matlab-функция: `model_releaser`
   Файл: `model_releaser.m`.
   Описание: Функция создает текстовую строку, описывающую модель, заданную в виде Matlab-функции. Более подробный формат задания модели в виде текстовой строки можно посмотреть в комментариях к Matlab-функции `model_grabber`.
   Параметры:

   - `model_input` - модель в виде Matlab-функции

   Функция возвращает:

   - `[model_output]` - модель в виде текстовой строки

10. Matlab-функция: `model_download`
    Файл: `model_download.m`.
    Описание: Функция создает множество функций-компонент и начальную популяцию моделей на основе данных из заданных файлов. В качестве примера формата входных файлов можно посмотреть файлы,

прилагающиеся к программному комплексу.
Параметры:

- `registry_filename` - файл с описанием функций компонент
- `mdl_filename` - файл с описанием начальной популяции моделей

Функция возвращает:

- `[registry, mdlpop]` - множество функций-компонент и популяцию моделей

11. Matlab-функция: `svr_get_popidx`
Файл: `svr_get_popidx.m`.
Описание: Функция выбирает наилучшие модели из двух популяций в соответствии с заданными индексами качества.
Параметры:

    - `evals1`, `evals2` - индексы качества моделей для двух популяций
    - `max_mdlnum` - максимальное количество моделей в новой популяции

    Функция возвращает:

    - `[idx1, idx2]` - номера моделей в соответствующих популяциях

12. Matlab-функция: `svr_node_mutation`
Файл: `svr_node_mutation.m`.

Описание: Функция производит процесс мутации заданной модели путем замены функций-компонент случайным образом.
Параметры:

- `net` - модель регрессии в виде дерева
- `cut` - максимальное количество элементов модели (функций-компонент), доступных для мутации
- `registry` - набор функций-компонент, которые и только которые доступны для мутации

Функция возвращает:

- `net` - мутировавшая модель

13. Matlab-функция: `svr_node_exchange`
Файл: `svr_node_exchange.m`.
Описание: Функция производит процесс обмена компонентами между заданными моделями, представленными в виде деревьев. Обмен происходит в виде обмена поддеревьями, выделенными из регрессионных моделей.
Параметры:

- `net1`, `net2` - модели регрессии в виде деревьев
- `cut1`, `cut2` - количество элементов (функций-компонент) для каждой модели, доступных для включения в поддерево для обмена

Функция возвращает:

- `[net1, net2]` - массив из двух моделей

14. Matlab-функция: `svr_model_compare`
    Файл: `svr_mdl_compare.m`.
    Описание: Функция производит поиск заданной модели в списке моделей.
    Параметры:

    - `net` - модель регрессии в виде дерева
    - `generated` - список моделей

    Функция возвращает:

    - `thenew` - 0, если модель присутствует в списке, 1 - в противном случае

15. Matlab-функция: `svr_gather_mdlpop`
    Файл: `svr_gather_mdlpop.m`.
    Описание: Функция объединяет две популяции регрессионных моделей в одну.
    Параметры:

    - `mdlpop`, `mdlpopnew` - модели регрессии в виде деревьев
    - `idx1`, `idx2` - номера моделей в популяции, соответствующие индексам качества
    - `evals`, `evalsnew` - индексы качества моделей для каждой популяции

    Функция возвращает:

    - `[mdlpop,evals]` - популяцию моделей и индексы качества моделей в популяции

8.2. Модуль func

Модуль содержит функции-компоненты, использующиеся для генерации регрессионных моделей. Все Matlab-функции имеют схожую сигнатуру, состоящую из нескольких параметров, первый из которых - это вектор параметров **w** и параметров-аргументов реализуемой функции.

1. Функция: $f(x) = x_1 + x_2$.
   Matlab-функция: `f_2plus(w,x,x2)`
   Файл: `f_2plus.m`.
   Описание: функция `f_2plus` производит поэлементное сложение двух матриц.
   Параметры:
   - `x`, `x2` - матрицы размера $K \times N$
   - `w` - не используется
2. Функция: $f(x) = x_1 - x_2$.
   Matlab-функция: `f_2minus(w,x,x2)`
   Файл: `f_2minus.m`.
   Описание: функция `f_2minus` производит поэлементное вычитание двух матриц.
   Параметры:
   - `x`, `x2` - матрицы размера $K \times N$
   - `w` - не используется
3. Функция: $f(x) = x_1 \cdot x_2$.
   Matlab-функция: `f_2times(w,x,x2)`
   Файл: `f_2times.m`.

Описание: функция `f_2times` производит по-элементное умножение двух матриц.
Параметры:

- `x`, `x2` - матрицы размера $K \times N$
- `w` - не используется

4. Функция: $f(x) = \frac{x_1}{x_2}$.
   Matlab-функция: `f_2divide(w,x,x2)`
   Файл: `f_2divide.m`.
   Описание: функция `f_2divide` производит по-элементное деления двух матриц.
   Параметры:

   - `x`, `x2` - матрицы размера $K \times N$
   - `w` - не используется

5. Функция: $f(x) = \alpha x$.
   Matlab-функция: `f_mult(w,x,dummy)`
   Файл: `f_mult.m`.
   Описание: функция `f_mult` применяет указанную выше функцию $f(x)$ по-элементно к матрице размера $K \times N$.
   Параметры:

   - `x` - матрица размера $K \times N$
   - `w[1]` - соответствует α
   - `dummy` - не используется

6. Функция: $f(x) = \alpha + x$.
   Matlab-функция: `f_plus(w,x,dummy)`
   Файл: `f_plus.m`.

Описание: функция `f_plus` применяет указанную выше функцию $f(x)$ по-элементно к матрице размера $K \times N$.
Параметры:

- `x` - матрица размера $K \times N$
- `w[1]` - соответствует α
- `dummy` - не используется

7. Функция: $f(x) = \alpha x + \beta$.
   Matlab-функция: `f_linear(w,x,dummy)`
   Файл: `f_linear.m`.
   Описание: функция `f_linear` применяет указанную выше функцию $f(x)$ по-элементно к матрице размера $K \times N$.
   Параметры:

   - `x` - матрица размера $K \times N$
   - `w[1]` - соответствует α
   - `w[2]` - соответствует β
   - `dummy` - не используется

8. Функция: $f(x) = \alpha x^2 + \beta x + \gamma$.
   Matlab-функция: `f_parabolic(w,x,dummy)`
   Файл: `f_parabolic.m`.
   Описание: функция `f_parabolic` применяет указанную выше функцию $f(x)$ по-элементно к матрице размера $K \times N$.
   Параметры:

   - `x` - матрица размера $K \times N$

- `w[1]` - соответствует α
- `w[2]` - соответствует β
- `w[3]` - соответствует γ
- `dummy` - не используется

9. Функция: $f(x) = \alpha x^3 + \beta x^2 + \gamma x + \delta$.
   Matlab-функция: `f_cubic(w,x,dummy)`
   Файл: `f_cubic.m`.
   Описание: функция `f_cubic` применяет указанную выше функцию $f(x)$ по-элементно к матрице размера $K \times N$.
   Параметры:

   - `x` - матрица размера $K \times N$
   - `w[1]` - соответствует α
   - `w[2]` - соответствует β
   - `w[3]` - соответствует γ
   - `w[4]` - соответствует δ
   - `dummy` - не используется

10. Функция: $f(x) = \frac{\alpha}{\sqrt{2\pi}\gamma}\exp\left(-\frac{(x-\beta)^2}{2\gamma^2}\right) + \delta$
    Matlab-функция: `f_gaussian(w,x,dummy)`
    Файл: `f_gaussian.m`
    Описание: функция `f_gaussian` применяет указанную выше функцию $f(x)$ по-элементно к матрице размера $K \times N$.
    Параметры:

    - `x` - матрица размера $K \times N$

- `w[1]` - соответствует α
- `w[2]` - соответствует β
- `w[3]` - соответствует γ
- `w[4]` - соответствует δ
- `dummy` - не используется

11. Функция (файл: `f_quadratic.m`)

12. Функция: $f(x) = \alpha \sin(\beta x + \gamma) + \delta$
    Matlab-функция: `f_sin(w,x,dummy)`
    Файл: `f_sin.m`
    Описание: функция `f_sin` применяет указанную выше функцию $f(x)$ по-элементно к матрице размера $K \times N$.
    Параметры:

    - `x` - матрица размера $K \times N$
    - `w[1]` - соответствует α
    - `w[2]` - соответствует β
    - `w[3]` - соответствует γ
    - `w[4]` - соответствует δ
    - `dummy` - не используется

13. Функция: $f(x) = \tanh(x)$.
    Matlab-функция: `f_tanh(w,x)`
    Файл: `f_tanh.m`.
    Описание: функция `f_tanh` применяет указанную выше функцию $f(x)$ по-элементно к матрице размера $K \times N$.
    Параметры:

- `x` - матрица размера $K \times N$
- `w` - не используется

14. Функция: $f(x) = \alpha(\tanh(\gamma x - \beta) + 1) + \delta$.
Matlab-функция: `f_tansig(w,x,dummy)`
Файл: `f_tansig.m`.
Описание: функция `f_tansig` применяет указанную выше функцию $f(x)$ по-элементно к матрице размера $K \times N$. В качестве реализации функции tanh используется Matlab-функция `tansig`.
Параметры:

    - `x` - матрица размера $K \times N$
    - `w[1]` - соответствует α
    - `w[2]` - соответствует β
    - `w[3]` - соответствует γ
    - `w[4]` - соответствует δ
    - `dummy` - не используется

15. Функция: $f(x) = \alpha \left(\frac{1}{\exp(\beta - \gamma x) + 2} \right) + \delta$.
Matlab-функция: `f_logsig(w,x,dummy)`
Файл: `f_logsig.m`.
Описание: функция `f_logsig` применяет указанную выше функцию $f(x)$ по-элементно к матрице размера $K \times N$. В реализации функции используется Matlab-функция `logsig`.
Параметры:

    - `x` - матрица размера $K \times N$
    - `w[1]` - соответствует α

- `w[2]` - соответствует β
- `w[3]` - соответствует γ
- `w[4]` - соответствует δ
- `dummy` - не используется

9. Заключение

Сложные регрессионные модели, например, нейронные сети при обработке результатов измерений часто имеют большое число параметров и получаются переобученными. Для достижения результатов в построении несложных и достаточно точных моделей была поставлена задача о выборе композиции функций. Эта задача была проиллюстрирована для параметрических и непараметрических функций двумя примерами. Широкий выбор моделей, которые можно использовать в регрессионном анализе, а также множество методов оптимизации позволяют надеяться на плодотворность описанного метода.

При восстановлении регрессии большую роль играет предварительная обработка данных, так как существующие методы восстановления регрессии восстанавливают регрессию неудовлетворительно в связи с тем, что искомые алгоритмы преобразования входных данных не вполне соответствуют моделям регрессии. Была поставлена и решена задача генерации и выбора алгоритмов, создающих набор производных описаний, которые являются входными для процедур восстановления нелинейной регрессии.

Предлагаемый метод заключается в нахождении композиции двух операторов: оператора, отображающего пространство исходных временных рядов в пространство рас-

ширенного набора описаний и оператора, отображающего пространство расширенного набора описаний в пространство откликов. Первый оператор отыскивается в заданном семействе эвристических алгоритмов, второй оператор отыскивается в заданном семействе регрессионных моделей. Для нахождения операторов используются методы стохастической оптимизации.

Список литературы

[1] *LeCun, Y., Denker, J. S., and Solla, S. A.* Optimal brain damage // Touretzky, D.S., ed. Advances in Neural Information Processing Systems. Morgan Kaufmann, San Mateo, CA, 1990. P. 598–605.

[2] *MacKay, D.* Information, inference, learning algorithms. Cambridge University Press, 2003.

[3] *MacKay, D.* Hyperparameters: optimise or integrate out? // Heidberger, G., ed. Maximum entropy and Bayesian Methods. Santa Barbara, Dordrecht: Kluwer, 1993.

[4] *MacKay, D.* Bayesian interpolation // Neural Computation 4(3), 1992. P. 415–447.

[5] *Nabney, I.T.* NETLAB: Algorithms for pattern recognition. Springer, 2004. P. 330.

[6] *MacKay, D.* Choice of basis for Laplace approximation // Machine Learning, vol. 33(1), 1998.

[7] Levenberg, K. A Method for the Solution of Certain Problems in Last Squares. Quart. Appl. Math. Vol. 2, pp. 164–168, 1944.

[8] Malada, H.R., Ivakhnenko, A. G. Inductive Learning Algorithms for Complex Systems Modeling. CRC Press, 1994.

[9] Bishop, C.M., Tipping, M.E. Bayesian regression and classification. In J. Suykens, G. Horvath, S. Basu, C. Micchelli, and J. Vandewalle (Eds.), Advances in Learning

Theory: Methods, Models and Applications, Volume 190, pp. 267–285. IOS Press, NATO Science Series III: Computer and Systems Sciences.

[10] Group Method of Data Handling, http://www.gmdh.net/.

[11] Nikolaev, N. Iba, H. Accelerated Genetic Programming of Polynomials, Genetic Programmimg and Evolvable Machines. Kluwer Academic Publ., vol.2(3), pp.231–257.

[12] Branch, M.A., Coleman, T.F., Li, Y. A Subspace, Interior, and Conjugate Gradient Method for Large-Scale Bound-Constrained Minimization Problems. SIAM Journal on Scientific Computing, vol. 21(1), pp. 1–23, 1999.

[13] Coleman, T.F., Y. Li, An Interior, Trust Region Approach for Nonlinear Minimization Subject to Bounds. SIAM Journal on Optimization, vol. 6, pp. 418–445, 1996.

[14] Goldberg, D. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Professional, 1989.

[15] Chipperfield, A., Fleming, P., Pohlheim, H., Fonesca, C. Genetic Algorithm Toolbox. User's guide. University of Sheffield, 1994.

[16] Demuth, H., Beale, M. Neural Network Toolbox User's Guide. The MathWorks, Inc., 2000.

[17] Хардле, В. Прикладная непараметрическая регрессия. М: Мир, 1993.

[18] Golyandina, N., Nekrutkin, V., Zhigljavsky, A. Analysis of Time Series Structure SSA and Related Techniques. CHAPMAN & HALL/CRC, 2001.

[19] Рао, С.Р. Линейные статистические методы и их применения. М: Наука, 1968. С. 530–533.