Выбор математической модели: баланс между сложностью и близостью к измерениям

А.В. Соколов, В.В. Волошинов

***Аннотация*—Предлагается методика выбора математических моделей на основе баланса между сложностью и точностью соответствия имеющимся экспериментальным данным. В основе методики: 1) набор (параметрических) семейств моделей, пригодных для удовлетворительного воспроизведения измерений; 2) формализация понятия сложности модели (для выбранного семейства); 3) процедура перекрестной проверки для оценки погрешности моделирования измерений; 4) поиск оптимального компромисса между сложностью модели и близостью к измерениям на основе минимизации погрешности моделирования измерений. Рассматривается пример, демонстрирующий применение методики. Формулируется общая постановка задачи. Обсуждаются вопросы программной реализации в распределенной вычислительной среде.**

***Ключевые слова* — идентификация, обратные задачи, регуляризация, распределенные вычисления, платформа Everest.**

I. Введение

В настоящее время математическое моделирование является одним из основных инструментов научного познания. Математические модели используются практически при любом исследовании реальных объектов, особенно в том случае, когда цель исследования носит прикладной характер. Разнообразие типов моделей приводит к проблеме выбора: один и тот же объект может описываться разными моделями. Например, существует значительное количество математических моделей типа “хищник-жертва” с более или менее полным математическим анализом их формальных свойств. Другим актуальным примером является моделирование динамики глюкозы в крови. В этом случае, количество различных моделей измеряется десятками (см. обзоры [1] и [2]).

В связи с этим возникают вопросы о критериях выбора модели, соответствии модели целям исследования, адекватности модели исследуемому объекту, погрешности описания, области применимости и т.д. Ответы на эти вопросы могут быть получены лишь при наличии достаточной информации об объекте. Очевидно, что чем больше количественной информации (измерений) и чем она точнее, чем больше качественной информации о поведении и свойствах объекта, тем более сложными, адекватными и точными могут быть построенные на их основе модели, тем шире может быть область их использования. Таким образом, выбор математической модели реального объекта является результатом компромисса между количеством и качеством исходной информации об объекте и сложностью модели.

Обычно выбор модели осуществляется от “простого к сложному” – сначала выбирается относительно простая модель, которая усложняется (путем добавления дополнительных неизвестных коэффициентов, переменных, учета дополнительных процессов и т.д.) до тех пор, пока её решение не будет “удовлетворительно” соответствовать измерениям.

Однако существует и другой подход. Он состоит в выборе из достаточно широкого семейства моделей той, сложность которой соответствует количеству и качеству (точности) измерений. Для его реализации необходимо: задать семейство моделей и сформулировать критерий выбора, содержащий меру сложности модели и меру соответствия модели измерениям.

Формализация этого подхода приводит к различным типам задач: вариационным задачам; обратным задачам с регуляризацией [3]; задачам сплайн аппроксимации [4]; задачам непараметрической регрессии [5]; задачам предсказательного моделирования [6]; задачам машинного обучения [7] и др.

II. Демонстрационный пример

Применим предлагаемую методику для восстановления (на основе измерений) зависимости *x(t)*

$$\bar{x}(t) = \sin\left(\sqrt{k}t\right)\exp\left(-\frac{\mu}{2}t\right) + \Delta x, \quad (1)$$

где $k=1.4$, $\mu=0.4$, $\Delta x=1.2$.

Построим следующий набор измерений (здесь и далее 1..n обозначает множество чисел {1,2,…,n}):

$$D: \ \{z_k, t_k\},\ k \in \mathrm{K},\ \mathrm{K} = 1..k_{\max}, \quad (2)$$

где $t_k = t_{\min} + k * h_t$,

$h_t = (t_{\max} - t_{\min})/k_{\max}$,

$t_{\min} = -1.0, \quad t_{\max} = 2.5,$

$z_k = \bar{x}(t_k) + \varepsilon_k$, ε_k - случайная ошибка с нулевым

Статья получена 6 августа 2018. Работа выполнена при поддержке РФФИ, проект 18-07-01175 и 17-07-00027

А.В. Соколов, Институт проблем передачи информации им. А.А. Харкевича РАН, Институт геохимии и аналитической химии им. В.И. Вернадского РАН (e-mail: alexander.v.sokolov@gmail.com).

В.В. Волошинов, Институт проблем передачи информации им. А.А. Харкевича РАН (e-mail: vv_voloshinov@iitp.ru).

средним и дисперсией равной 0.1. “Искаженные” таким способом измерения представлены на рис. 1.

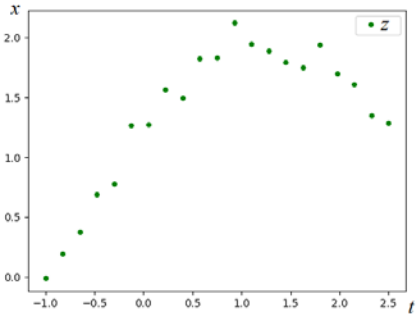


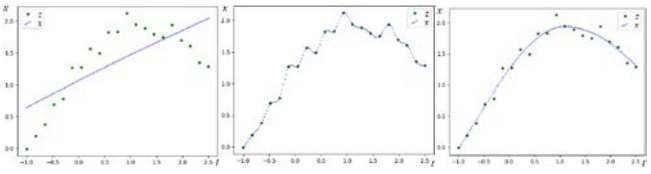
Рис. 1. Исходные данные демонстрационного примера

“Забудем” на время о том, какая формула использовалась для генерации данных и о величине дисперсии. Поставим задачу определить эти неизвестные, пользуясь методами математического моделирования и специальной методики анализа исходных измерений. Для этого рассмотрим несколько математических моделей с целью получить оценки погрешностей моделирования измерений, которые и будут играть роль критерия при сравнении моделей.

Найти аналитическое решение для соответствующих оптимизационных постановок, за редким исключением, затруднительно. Для построения аппроксимирующих численных моделей искомые непрерывные функции заменяются на сеточные или полиномиальные, их производные и интегралы - на разностные аналоги. Численные модели были реализованы на языке программирования Python. При решении оптимизационных задач использовались сервисы оптимизации в среде распределенных вычислений Everest [8] (см. более подробно – в разделе V).

Модель 0. Функция. Сплайн аппроксимация. Выберем в качестве модели дважды дифференцируемую функцию *x(t)*:

$$x = x(t),\ x(\cdot) \in C^2(t_{\min}, t_{\max}) \qquad (3)$$



а б в

Рис. 2. Различные варианты аппроксимации данных

Прямая на рис. 2а является простейшим вариантом линейной аппроксимации, но она проходит слишком далеко от исходных измерений. Ее противоположностью является кривая на рис. 2б – слишком сложная (переобученная) модель, воспроизводящая все ошибки измерений. Наконец, кривая на рис. 2в соответствует оптимальному балансу между близостью модели (функции) к измерениям и ее простотой. Она получена методом регуляризованной оптимизационной идентификации.

Для получения решений с различным соотношением «отклонение от измерений» / «сложность модели» воспользуемся сплайн аппроксимацией, состоящей в минимизацией функционала

$$F(x,K,\alpha) = \frac{1}{|K|}\sum_{k\in K}(z_k - x(t_k))^2 + \alpha\int_{t_{min}}^{t_{max}}\left(\frac{d^2x}{dt^2}\right)^2 dt \to \min_{x(\cdot)} \qquad (4)$$

$$x(\cdot) \in C^2(t_{min}, t_{max})$$

Первое слагаемое - мера соответствия модели измерениям (среднеквадратичное отклонение), второе – взвешенная (с весом $\alpha > 0$) мера сложности модели как мера кривизны функции *x(t)*, интеграл квадрата второй производной.

Известно [4], что решение задачи сплайн аппроксимации (4) существует и единственно (при достаточно простых условиях на узлы аппроксимации) при любом $\alpha > 0$ и представляет собой обычный кубический сплайн. При $\alpha \to 0$ второе слагаемое стремится к 0 и мы получим задачу сплайн интерполяции: поиск функции минимальной кривизны, проходящей через заданные точки (см. рис. 2б). При $\alpha \to \infty$ второе слагаемое подавляет первое, и задача сводится к поиску прямой (нулевой кривизны) с минимальной суммой квадратов отклонений от измерений (см. рис. 2а) методом линейной регрессии.

Рассмотренные предельные случаи являются недостаточными при поиске сложных (нелинейных) функций, описывающих сильно зашумленные измерения – обычно требуется найти некоторое промежуточное значение α, при котором «плавная» траектория модели будет проходить достаточно близко к измерениям, сглаживая случайные погрешности (см. рис. 2в). Выбор такого оптимального веса α может быть проведен минимизацией оценки погрешности решения, полученной на независимом экспериментальном материале (верификация) или на основе процедуры перекрестного оценивания (cross-validation).

Процедура перекрестного оценивания состоит в многократном разбиении набора измерений на две части, одна из которых (обучающая последовательность) используется для нахождения аппроксимирующей функции (путем минимизации функционала (4)), а вторая (тестирующая последовательность) – для оценки погрешности найденной аппроксимации.

Простейший вариант процедуры перекрестного оценивания (leave-one-out) состоит в использовании тестирующей выборки состоящей из одной точки (с номером k) и обучающей выборки, состоящей из оставшегося набора измерений $K\backslash k$ (множество измерений (2) с одним удаленным элементом $\{z_k, t_k\}$). Для каждого такого разбиения (и заданного α) ищется решение *x*, обеспечивающее минимум (4):

$$x^{\alpha}_{K\backslash k} = \underset{x(.)}{Arg\,min}\{F(x(\cdot), K\setminus k, \alpha)\},\ \ k = 0..k_{max}\ , \qquad (5)$$

Полученные решения используются для расчета отклонений решения модели от измерений:

$$z_k - x^{\alpha}_{K\backslash k}(t_k),\ \ k = 0..k_{max}\,.$$

Осреднение квадратов найденных отклонений по всем разбиениям приводит к итоговой оценке погрешности

$$\sigma^{\alpha} = \sqrt{\frac{1}{|K|}\sum_{k\in K}\left(z_k - x^{\alpha}_{K\setminus k}(t_k)\right)^2}\ ,$$

минимизация которой (как функции α) и приводит к сбалансированной модели, оптимально сочетающей близость к измерениям и простоту.

Таким образом, поиск оптимального α сводится к двухуровневой задаче математического программирования - поиску такого α, при котором отклонение от измерений минимально (верхний уровень оптимизации):

$$\alpha^* = \underset{\alpha}{Arg\,min}\sqrt{\frac{1}{|K|}\sum_{k\in K}\left(z_k - x^{\alpha}_{K\setminus k}(t_k)\right)^2} \qquad (6)$$

где $x^{\alpha}_{K\setminus k}$ рассчитываются по (5) (нижний уровень оптимизации).

Отклонение от измерений, соответствующее найденному оптимальному α^*, будем называть ***погрешностью моделирования измерений***:

$$\sigma^* = \sqrt{\frac{1}{|K|}\sum_{k\in K}\left(z_k - x^{\alpha^*}_{K\setminus k}(t_k)\right)^2} \qquad (7)$$

Именно этот показатель будет использоваться в качестве меры соответствия модели исходным измерениям (2).

Наконец, для найденного α^* определим (оптимальное) решение (на полном наборе измерений K):

$$x^* = x^{\alpha^*}_{K} = \underset{x(\cdot)}{Arg\,\min}\left\{F\left(x(\cdot), K, \alpha^*\right)\right\} \qquad (8)$$

и его среднеквадратичное отклонение от измерений (Mean Squared Error):

$$mse^* = \sqrt{\frac{1}{|K|}\sum_{k\in K}\left(z_k - x^*(t_k)\right)^2} \qquad (9)$$

Таким образом, процедура построения «сбалансированного» решения $x^*(t)$ модели (3) для заданного множества измерений (2) состоит в поиске значения веса α^* в функционале (4), при котором достигается минимум (6) при условиях (5). Найденное α^* позволяет определить погрешность моделирования измерений σ^* (7), окончательное решение $x^*(t)$ (8) и его среднеквадратичное отклонение от измерений (9).

Результаты приведены на рис. 3 и в строке 0 табл. 1.

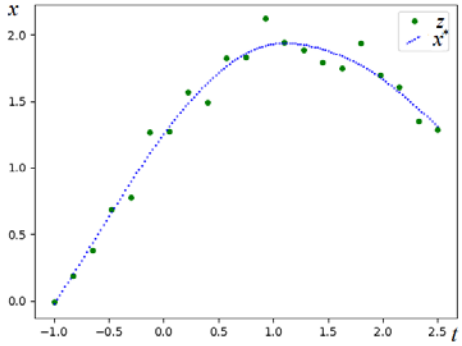


Рис. 3. Сплайн аппроксимация: z – измерения, x^* – найденное решение.

Построенная модель *0*, представляет собой один из простейших способов аппроксимации данных и не несет в явном виде информации об исследуемом дифференциальном процессе. Однако полученную оценку точности можно использовать как некий "эталон", с которым можно сравнивать оценки, полученные на основе более сложных моделей.

Модель 1. Дифференциальное уравнение 1-го порядка. Применим рассмотренную выше технологию выбора сбалансированного решения к модели, содержащей две неизвестные функции $x(t)$ и $f(x)$, связанные дифференциальным уравнением:

$$\frac{dx}{dt} = f(x),\quad x(t)\in \mathrm{C}^2(t_{\min}, t_{\max}),\ f(x)\in \mathrm{C}^2(x_{\min}, x_{\max}), \qquad (10)$$

где x_{min} и x_{max} задают интервал изменения значений $x(t)$.

В этом случае можно определить сложность модели не через ее траекторию ($x(t)$), а через функцию, определяющую динамику процесса - правую часть дифференциального уравнения ($f(x)$). Тогда критерий выбора (функционал) примет вид:

$$F(x, f, K, \alpha) = \frac{1}{|K|}\sum_{k\in K}(z_k - x(t_k))^2 + \alpha\int_{x_{min}}^{x_{max}}\left(\frac{d^2 f}{dx^2}\right)^2 dx\ , \qquad (11)$$

Процедура построения «сбалансированного» решения $x^*(t), f^*(x)$ для заданного множества измерений (2) состоит в поиске такого значения веса α в функционале (11), при котором достигается минимум оценки погрешности (6), где для всех $k = 0..k_{\max}$

$$\left(x^{\alpha}_{K\setminus k}, f^{\alpha}_{K\setminus k}\right) = \underset{x,f}{Arg\,min}\left\{F(x, f, K\setminus k, \alpha):\ \frac{dx}{dt} = f(x)\right\} \qquad (12)$$

Найденное α^* позволяет определить погрешность моделирования измерений σ^* по формуле (7) и окончательное решение.

$$\left(x^*, f^*\right) = \underset{x,f}{Arg\,\min}\left\{F\left(x, f, K, \alpha^*\right):\ \frac{dx}{dt} = f(x)\right\} \qquad (13)$$

Результаты приведены на рис. 4 и в строке 1 табл. 1. Переход от модели 0 к модели 1 (от произвольной гладкой функции к функции-решению уравнения (10)) сокращает возможности описания измерений (2) функцией $x(t)$. Оказалось, что при этом отбрасываются существенные возможности описания измерений (2) и множество допустимых решений, заданных уравнением (10), является «слишком узким». Действительно, решениями (10) могут быть лишь монотонные функции $x(t)$, что явно недостаточно для аппроксимации исследуемых измерений. Об этом свидетельствует и заметное (по сравнению с моделью 0) увеличение погрешности моделирования данных (сравнение строк 1 и 0 в табл. 1).

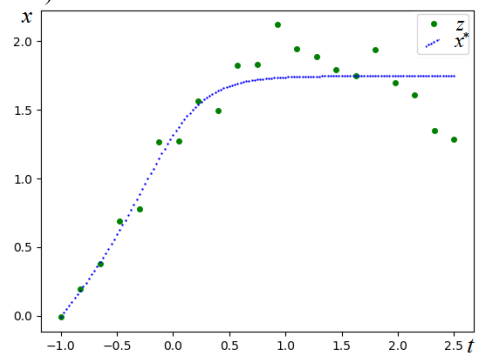


Рис. 4. Дифференциальное уравнение 1-ого порядка: z – измерения, x^* – найденное решение.

Модель 2. Дифференциальное уравнение 2-го порядка. Пусть модель содержит две неизвестные функции $x(t)$ и $f(x,v)$, связанные дифференциальным уравнением:

$$\frac{d^2x}{dt^2} = f\left(x, \frac{dx}{dt}\right),\ x \in \mathrm{C}^2(t_{\min}, t_{\max}),\ f \in \mathrm{C}^2(G), G \subset \mathrm{R}^2, \qquad (14)$$

где двумерная область G содержит возможные траектории $\left(x(t), x'(t)\right)$. Здесь правая часть зависит не только от x, но и от скорости его изменения $v(t) = x'(t)$.

Вновь, определим сложность модели через характеристику гладкости функции, определяющей динамику процесса - правой части дифференциального уравнения. Тогда критерий выбора примет вид:

$$F(x, f, K, \alpha_x, \alpha_v) = \frac{1}{|K|}\sum_{k\in K}(z_k - x(t_k))^2 + \\ + \iint\limits_G \left[\alpha_x^2\left(\frac{\partial^2 f}{\partial x^2}\right)^2 + 2\alpha_x\alpha_v\left(\frac{\partial^2 f}{\partial x \partial v}\right)^2 + \alpha_v^2\left(\frac{\partial^2 f}{\partial v^2}\right)^2\right] dxdv, \qquad (15)$$

Здесь требуется найти уже два весовых коэффициента, каждый из которых определяет сложность модели по «своей» переменной.

Процедура построения «сбалансированного» решения $x^*(t), f^*(G)$ аналогична процедуре, использованной для модели 1, при замене модели и критерия на (14) и (15).

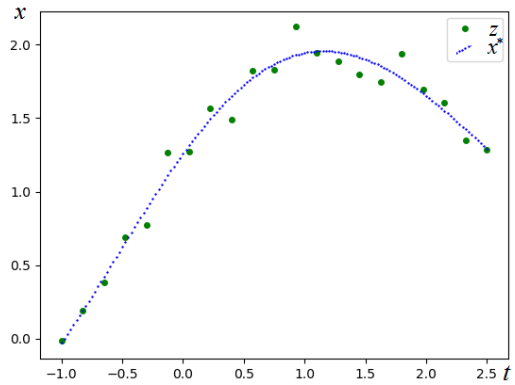


Рис. 5. Дифференциальное уравнение 2-ого порядка: z – измерения, x^* – найденное решение.

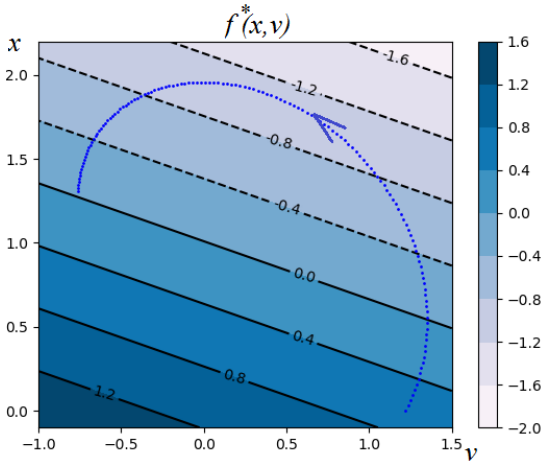


Рис. 6. Дифференциальное уравнение 2-ого порядка: правая часть уравнения $f(x,v)$ (линии уровня) и траектория (кривая в координатах x,v).

Результаты моделирования приведены на рис. 5 и Рис 6 и в строке 2 табл. 1. Их анализ позволяет сделать следующие выводы.

1. Погрешности в строках 0 и 2 практически одинаковы, т.е. переход от модели 0 к модели 2 (от произвольной гладкой функции к функции-решению уравнения (14)) не сужает возможностей моделирования измерений (2).

2. Изолинии, на рис. 6, являются параллельными, равноотстоящими друг от друга прямыми, т.е. график правой части это плоскость, а $f(x,v)$ линейная функция.

Модель 3. Осциллятор с вязким трением. Вывод о линейности правой части уравнения (14) по x и v указывает на возможность ее упрощения. Выберем параметризацию осциллятора в вязкой среде:

$$\frac{d^2x}{dt^2} = -k(x - \Delta x) - \mu\frac{dx}{dt}, \qquad (16)$$

где k – жесткость пружины, Δx – смещение точки ее крепления, μ - вязкость среды (в формуле Стокса).

Результаты моделирования приведены в строке 3 табл. 1. График траектории аналогичен тому, что изображен на рис. 5. Предположение о линейной зависимости оказалось удачным – погрешность моделирования данных несколько уменьшилась. **По-видимому, исследуемые измерения соответствуют динамики осциллятора с вязким трением.**

Промежуточные итоги. Единый подход к поиску решений, сочетающих близость к измерениям и простоту, был применен к четырем моделям. Использование одних и тех же измерений и одинаковой методики оценки погрешности (cross-validation) позволило провести сравнение полученных результатов и выбрать модель, наиболее подходящую для исследуемых данных (см. табл. 1).

Таблица 1. Погрешность моделирования измерений (σ^*) и среднеквадратичное отклонение найденного решения от измерений (mse^*) для различных моделей.

№	Модель	σ^*	mse^*
0	Функция. Сплайн-сглаживание	0.1144	0.0907
1	Дифференциальное уравнение 1-ого порядка	0.2093	0.1869
2	Дифференциальное уравнение 2-ого порядка	0.1146	0.0924
3	Осциллятор с вязким трением	0.1106	0.0932

Числовые результаты, приведенные в табл. 1, иллюстрируют предложенную схему постепенного уточнения математической модели наблюдаемого физического явления с количественной оценкой качества такого уточнения. Если оценка погрешности моделирования измерений (σ^*) модифицированной модели уменьшилась - мы на верном пути. С этой точки зрения, переход от модели 0 к модели 2 и от модели 2 к модели 3 представляется оправданным, а переход от модели 0 к модели 1 – ошибочным.

Еще одним показателем качества модели является среднеквадратичное отклонение найденного решения (модели) от измерений (mse^*), которое характеризует «свободу» выбора модели. Действительно, при принятии «правильной» дополнительной гипотезы, множество допустимых решений сужается за счет отбрасывания лишних частей (сужает свободу выбора), что приводит к

уточнению модели, в том числе к уточнению оценки mse^* (см. динамику mse^* в табл. 1).

Проведенные расчеты приводят к выводу, что модель 3 (осциллятор с вязким трением) лучше всего соответствует имеющимся измерениям. Действительно, формула (1), которая использовалась при генерировании измерений, является одним из решений уравнения (16) модели осциллятора с вязким трением [9].

III. Общая постановка задачи

Математическая модель представляет собой набор утверждений относительно свойств исследуемого объекта, формализованных в виде уравнений (например, $x'(t) = f(x)$), неравенств (например, $x'(t) \ge 0$), утверждений о принадлежности переменных к различным пространствам и множествам (например, $\vec{x} \in X$), логических выражений (например, $\exists y : f(y) = 0$) и т.д.

Для простоты представим этот набор утверждений (математическую модель) в виде системы уравнений:

$$\vec{M}(\vec{x}) = 0, \quad (17)$$

где $\vec{x}$ - переменные модели. К ним относятся как «внутренние» переменные модели, например, описывающие траекторию динамической системы (см. Демонстрационный пример), так и искомые параметры модели (например, коэффициенты упругости и вязкого трения в том же примере). Соотношения (17) определяют в пространстве переменных множество допустимых решений так, как это принято в задачах оптимизации (математического программирования):

$$Q = \{\vec{M}(\vec{x}) = 0\}, \quad (18)$$

Обычно, это - достаточно «ёмкое» множество и для его сужения (или для выбора одного из элементов) при моделировании реальных объектов производят измерения некоторых характеристик объекта, что дает набор исходных данных или множество измерений:

$$D: \quad \{z_k, P_k(\vec{x})\},\ k \in K,\ K = 1..k_{max}, \quad (19)$$

где z_k – значение измерения, а $P_k(\vec{x})$ - соответствующий оператор, выражающий измеряемые характеристики через переменные модели $\vec{x}$. Возникающая при этом (неизвестная) погрешность $\varepsilon_k = z_k - P_k(\vec{x})$, объясняется как ошибками измерений, так и неточностью математического описания реального объекта. Далее, для простоты, будем отождествлять набор исходных данных с множеством K индексов пар $\{z_k, P_k(\vec{x})\}$.

Определим (смешанный) критерий выбора:

$$F(\vec{x}, K, \vec{\alpha}) = \frac{1}{|K|} F_F(\vec{x}, K) + F_S(\vec{\alpha}, \vec{x}) \to \min_{\vec{x} \in Q}, \quad (20)$$

который будем использовать для поиска оптимального компромисса между близостью решения к измерениям (первое слагаемое) и сложностью модели (второе слагаемое).

Функционал $F_F(\vec{x}, K) \ge 0$ является мерой погрешности или близости решения модели (17) к измерениям (19). В качестве функционала близости решения к измерениям обычно используется аддитивная мера, т.е. если $K = K_1 \cup K_2$ и $K_1 \cap K_2 = \varnothing$, то $F_F(\vec{x}, K) = F_F(\vec{x}, K_1) + F_F(\vec{x}, K_2)$.

Пример - квадратичное отклонение:

$$F_F(\vec{x}, K) = \sum_{k \in K} (z_k - P_k(\vec{x}))^2. \quad (21)$$

Функционал $F_S(\vec{\alpha}, \vec{x}) \ge 0$ является мерой сложности модели (регуляризирующий функционал Тихонова [3]). Он зависит от искомого решения и вектора неотрицательных коэффициентов $\vec{\alpha} \in \mathrm{R}_+^{\mathrm{N}}$ (иногда стрелка над вектором $\vec{\alpha}$ будет опускаться). Эти коэффициенты определяют баланс между «близостью к данным» и «сложностью» решения. Отдельные компоненты вектора $\vec{\alpha}$ определяют значимость штрафов за различные «аспекты сложности» решения. Предполагается, что функционал сложности $F_S(\vec{\alpha}, \vec{x})$ возрастает по любой компоненте вектора α_i, стремится к 0 при $\alpha_i \to 0$ и стремится к $+\infty$ при $\alpha_i \to +\infty$.

Функционалы сложности выбираются исходя из специфики объекта и его модели (17). Ими могут быть различными характеристиками кривизны функций, входящих в постановку, некоторые интегральные показатели, такие как энергия или энтропия, мера зависимости выходных переменных от входных. Например, в демонстрационный примере в качестве характеристик кривизны для функций одной переменной используется выражение

$$F_S(\alpha, f) = \alpha \int_X \left(\frac{d^2 f}{dx^2} \right)^2 dx,$$

а для функций двух переменных –

$$F_S(\alpha_x, \alpha_y, f) = \alpha_x^2 \iint_{X\,Y} \left(\frac{\partial^2 f}{\partial x^2} \right)^2 dxdy + 2\alpha_x \alpha_y \iint_{X\,Y} \left(\frac{\partial^2 f}{\partial x \partial y} \right)^2 dxdy + \alpha_y^2 \iint_{X\,Y} \left(\frac{\partial^2 f}{\partial y^2} \right)^2 dxdy.$$

При заданном $\vec{\alpha}$ задача идентификации на множестве измерений (19) - это минимизация функционала (20) на множестве допустимых решений модели (18):

$$\vec{x}_{\mathrm{K}}^{\alpha} = Arg \min_{\vec{x}} \{F(\vec{x}, K, \vec{\alpha}) : \vec{x} \in Q\},$$

При $\vec{\alpha} \to +\infty$ решение имеет минимальную сложность, а близость к данным оптимизируется «по остаточному принципу». И, наоборот, при $\vec{\alpha} \to 0$ на первое место выходит минимизация близости решения модели к измерениям и, лишь затем минимизируется сложность. Оба предельных случая, редко представляют практический интерес - требуется найти промежуточное, сбалансированное решение, т.е. такое значение $\vec{\alpha}$, которое обеспечивает компромисс между близостью к измерениям и простотой модели.

Для выбора этого значения $\vec{\alpha}$ применяется процедура перекрёстного оценивания (cross-validation). Для этого множество индексов измерений K разбивается на набор непересекающихся подмножеств, соответствующих независимым (в терминах теории случайных величин) наборам измерений

$$K = \bigcup_{i \in I} K_i,\ K_{\mathrm{i}} \cap K_{\mathrm{j}} = \varnothing,\ i \ne j, \quad (22)$$

(заметим, что в простейшем случае, когда погрешности измерений имеют случайный характер, любые непересекающиеся наборы будут независимыми).

Зададим некоторое $\vec{\alpha}$. Удалим из множества K

подмножество K_i. Найдем минимум критерия (20) на оставшемся наборе данных $K\backslash K_i$ («обучающая» выборка). Пусть соответствующее решение $\vec{x}^{\alpha}_{K\backslash K_i}$:

$$\vec{x}^{\alpha}_{K\backslash K_i} = \underset{\vec{x}}{Arg\,min}\left\{F\left(\vec{x}, K\setminus K_i, \vec{\alpha}\right) : \vec{x}\in Q\right\}, \quad (23)$$

Для найденного $\vec{x}^{\alpha}_{K\backslash K_i}$ определим $F_F\left(\vec{x}^{\alpha}_{K\backslash K_i}, K_i\right)$ - отклонение решения от измерений z_k для $k\in K_i$ (на «контрольной» выборке). Повторяя эту процедуру для всех подмножеств K_i, $i\in I$ и суммируя полученные результаты, получим для заданного $\vec{\alpha}$ «перекрестную» оценку погрешности:

$$\Phi(\alpha) = \sum_{i\in I} F_F\left(\vec{x}^{\alpha}_{K\backslash K_i}, K_i\right).$$

Найдем такой вектор весовых параметров $\vec{\alpha}$, при котором эта оценка минимальна:

$$\vec{\alpha}^* = \underset{\alpha}{Arg\,min}\sum_{i\in I} F_F\left(\vec{x}^{\alpha}_{K\backslash K_i}, K_i\right). \quad (24)$$

и соответствующее решение (на полном наборе измерений K):

$$\vec{x}^* = \vec{x}^{\alpha^*}_{K} = \underset{\vec{x}}{Arg\,min}\left\{F\left(\vec{x}, K, \vec{\alpha}^*\right) : \vec{x}\in Q\right\}$$

Таким образом, вычисление оптимальных весов $\vec{\alpha}$ для набора измерений (19) представляет собой двухуровневую (вариационную) задачу оптимизации: на нижнем уровне для выбранного $\vec{\alpha}$ решается |I| оптимизационных (вариационных) задач перекрестного оценивания (23) и вычисляется перекрестная оценка погрешности, которая минимизируется на верхнем уровне (24).

Для случая использования в качестве меры близости квадратичного отклонения (21) вычислим соответствующую перекрестною оценку, которую будем называть *погрешностью моделирования измерений*

$$\sigma^* = \sqrt{\frac{1}{|K|}\sum_{i\in I}\sum_{k\in K_i}\left(z_k - P_k\left(\vec{x}^{\alpha^*}_{K\backslash K_i}\right)\right)^2}$$

и среднеквадратичное отклонение:

$$mse^* = \sqrt{\frac{1}{|K|}\sum_{k\in K}\left(z_k - P_k\left(\vec{x}^*\right)\right)^2}\ .$$

Практическая реализация такого подхода зависит от сложности модели исследуемого объекта, от количества и качества измерений.

Вообще говоря, указанная постановка приводит к вариационным задачам, если неизвестными в модели являются функции непрерывного аргумента. В последнем случае, для численного решения непрерывные функции заменяются на «сеточные» (с неизвестными значениями в узлах сетки) или полиномиальными (с неизвестными коэффициентами). В результате все сводится к задачам конечномерной оптимизации, для решения которых можно эффективно применять современные пакеты численных методов (для задач математического программирования) и высокопроизводительную (многопроцессорную) вычислительную технику.

IV. Схема численного решения

Таким образом, процедура расчетов соответствует двухуровневой задаче оптимизации:

$$\Phi(\vec{\alpha}) = \sum_{i\in I} F_F\left(\vec{x}^{\alpha}_{K\backslash K_i}, K_i\right) \to \min_{\alpha\ge 0}, \quad (25)$$

где $\vec{x}^{\alpha}_{K\backslash K_i}$ определяются в результате решения независимых задач перекрестной проверки ($i\in I$)

$$F\left(\vec{x}, K\setminus K_i, \vec{\alpha}\right) \to \min_{x} : \vec{x}\in Q \quad (26)$$

Опишем численную схему приближенного решения двухуровневой задачи (25), (26). В ее основе – последовательная аппроксимация целевой функции в задаче (25) многочленом 2-го порядка (по переменным - компонентам вектора $\vec{\alpha}$). Коэффициенты многочлена подбираются, чтобы наилучшим образом аппроксимировать значения функции $\Phi(\alpha)$ для уже «проверенных» значений вектора $\vec{\alpha}$. Т.е. функция $\Phi(\alpha)$ ищется в виде многомерного многочлена

$$\Pi(\boldsymbol{\pi}, \vec{\alpha}) = \sum_{1\le i\le j\le N}\pi_{ij}\alpha_i\alpha_j + \sum_{1\le i\le N}\pi_i\alpha_i\ , \quad (27)$$

где N – размерность вектора $\vec{\alpha}$, коэффициенты $\boldsymbol{\pi} = \left(\left\{\pi_{ij} : 1\le i\le j\le N\right\}, \left\{\pi_i : i = 1..N\right\}\right)$ (всего $\frac{N(N+3)}{2}$), уточняются в ходе вычислений.

Данный подход напоминает приемы суррогатной оптимизации, включая black-box optimization, [10], при минимизации функций с высокими вычислительными затратами на определение их значений при заданных аргументах. Обычно, здесь используется некоторая аппроксимация минимизируемой функции по динамически формируемой выборке пар (вектор аргументов, значение функции) и точность аппроксимации повышается при увеличении мощности это выборки. Часто это основано на предопределенном словаре параметрических зависимостей, параметры которых уточняются при пополнении выборки значений аппроксимируемой (и минимизируемой) функции [11].

Алгоритм SvF строит последовательность значений $\vec{\alpha}^{\nu}, \nu = 1,2,\ldots$ Пусть, на V-ом шаге уже известны значения $\vec{\alpha}^{\nu}$, $\nu = 1..V$, для которых вычислены значения $\Phi^{\nu} = \Phi\left(\vec{\alpha}^{\nu}\right)$. Без ограничения общности (возможно, после перенумерации) можно считать, что Φ^V - минимальное из полученных значений. Будем трактовать $\left\{\Phi^{\nu}, \vec{\alpha}^{\nu}\right\}_{\nu=1}^{V}$ как набор точек в R^{N+1}. Рассмотрим задачу аппроксимации этих точек графиком многочлена (27), причем, чем вектор $\vec{\alpha}^{\nu}$ ближе к «наилучшему» $\vec{\alpha}^V$, тем с большим весом он будет учитываться. Кроме того будем штрафовать за кривизну многочлена с коэффициентом μ (подлежащим выбору):

$$\sum_{\nu=1:V}\exp\left(-\left\|\vec{\alpha}^{\nu} - \vec{\alpha}^V\right\|\right)\left(\Phi^{\nu} - \Pi(\boldsymbol{\pi}, \vec{\alpha}^{\nu})\right)^2 + \mu\left(\sum_{1\le i\le j\le N}\pi_{ij}^2\right) \to \min_{\boldsymbol{\pi}} \quad (28)$$

Пусть $\boldsymbol{\pi}^*(\mu)$ оптимальное значение вектора коэффициентов $\boldsymbol{\pi}$ в этой задаче выпуклого (легко проверить) программирования. Будем выбирать значение μ, минимизируя отклонение значения аппроксимирующего полинома от наилучшего значения

Φ^V. Мы получаем новую двухуровневую задачу:

$$\left|\Pi(\boldsymbol{\pi}^*(\mu),\vec{\alpha}^V)-\Phi^V\right|\to\min_{\mu\ge 0}, \qquad (29)$$

где $\boldsymbol{\pi}^*(\mu)$ – решение задачи «нижнего уровня» (28).

Здесь, в задаче верхнего уровня, нужно выбрать скалярную переменную μ, а задача нижнего уровня эффективно разрешима благодаря ее выпуклости. Поэтому для поиска оптимального коэффициента μ* применим тот или иной известный алгоритм минимизации функции одного переменного [12].

После аппроксимации зависимости $\Phi(\vec{\alpha})$ полиномом $\Pi(\boldsymbol{\pi}^*(\mu^*),\vec{\alpha})$, новое значение вектора $\vec{\alpha}$ находится в результате решения следующей вспомогательной задачи, которая напоминает метод линеаризации Пшеничного-Данилина [13], применяемый к минимизации функции $\Pi(\boldsymbol{\pi}^*(\mu^*),\vec{\alpha})$ по $\vec{\alpha}$:

$$\left(\nabla_{\alpha}\Pi(\boldsymbol{\pi}^*(\mu^*),\vec{\alpha}^V)\right)^{\mathrm{T}}\left(\vec{\alpha}-\vec{\alpha}^V\right)+\frac{1}{2}\left\|\vec{\alpha}-\vec{\alpha}^V\right\|^2\to\min_{\alpha\ge 0}, \quad \text{где}$$

$\nabla_{\alpha}\Pi(\ldots)$ - градиент многочлена Π по $\vec{\alpha}$.

Пусть $\vec{\alpha}^{V+1}$- решение этой выпуклой задачи квадратичного программирования (оно существует и единственно). Вычислим значение $\Phi^{V+1}=\Phi\left(\vec{\alpha}^{V+1}\right)$, решая набор задач нижнего уровня (26). Если величина $\left|\Phi\left(\vec{\alpha}^{V+1}\right)-\Phi\left(\vec{\alpha}^V\right)\right|$ меньше некоторого порогового значения, работа алгоритма прекращается. Если это не так, то схема расчетов повторяется для расширенного набора $\left\{\Phi^{\nu},\vec{\alpha}^{\nu}\right\}_{\nu=1}^{V+1}$.

V. Программная реализация на платформе Everest

Предлагаемая выше схема расчетов основана на решении задач математического программирования. Причем, на этапе вычисления задач нижнего уровня (для перекрестной проверки) требуется решить набор независимых задач. Более того, схема расчетов является итеративной и число итераций заранее неизвестно. Для программной реализации нужны: 1) средства описания задач и динамического формирования структур данных различных «экземпляров» задач; 3) вычислительная среда сервисов оптимизации, которым можно отправить исходные данные для решателей (пакетов численных методов); 4) средства обработки результатов решения и управления всем сценарием расчетов.

Исходя из накопленного опыта, было решено использовать “высокоуровневые” средства, развивающие методику применения алгебраических языков оптимизационного моделирования. Развитие таких языков (AML, Algebraic Modelling Language - в англоязычной литературе) ведется уже более 30 лет. До настоящего времени наиболее популярными из них являются AMPL, GAMS. Основные составляющие AML-систем: сам язык (набор правил для описания оптимизационных моделей); средства автоматического дифференцирования (для численных методов решения нелинейных задач); унифицированный интерфейс взаимодействия с решателями. Важно, что большинство решателей для основных классов задач математического программирования уже оснащены этими интерфейсами.

AML-языки формализуют описание всех составных элементов оптимизационных задач: множества индексов параметров, переменных и ограничений; функции критерия и ограничения. При этом «символьное описание» задачи, т.н. модельное представление, можно отделить от задания конкретных значений наборам индексов, числовым параметрам и т.п. Символьная модель и конкретные данные, представленные, обычно, текстовыми файлами, обрабатываются специальным транслятором. На выходе - специальная структура данных в виде т.н. стаб-файла (stub, в терминологии AMPL), готового для передачи решателям. Для нелинейных задач стаб также содержит правила вычисления первых и вторых производных всех функций задачи математического программирования. Если AML-совместимый решатель находит решения, то он возвращает файл, содержащий значения всех «прямых» и двойственных переменных задачи (множителей Лагранжа при ограничениях). Формат этого файла соответствует стандарту применяемого AML и его содержимое может быть считано транслятором для анализа результатов решения.

Помимо самих задач, AML-языки позволяют описывать сценарии расчетов, содержащие условные переходы, циклы, динамическое формирование новых задач на основе результатов решения предыдущих и т.п. Будучи, по назначению, языками программирования высокого уровня, AMPL и GAMS уже не отвечают требованиям, предъявляемым даже к процедурным языкам. Здесь надо учесть их «почтенный возраст» - они появились в конце 70-х годов прошлого века. Например, в них нет понятия процедуры-функции, все переменные (кроме внутренних индексов циклов или операторов «итерирования») являются глобальными и т.п.

В связи с эти, удобнее применять систему оптимизационного моделирования Pyomo (PYthon Optimization Modeling Objects) [14] http://pyomo.org, основанную на популярном, объектно-ориентированном языке программирования Python (Pyomo представляет собой специализированный Python-пакет). Благодаря тому, что авторы AMPL в 2005 г. «раскрыли» внутренние форматы AMPL-стаба [15], в 2013 г. Pyomo стал совместимым со стандартом AMPL.

Принцип расчетов в системе Pyomo повторяет схему применения языка AMPL в среде сервисов оптимизации на платформе Everest [16]: модель (в форме набора Python-объектов) вместе с исходными данными (либо в виде Python-объектов, либо в формате файлов с данными AMPL-формате) преобразуются в AMPL-стаб, передаваемый сервису оптимизации, к которому подключены решатели, установленные в узлах вычислительной среды. Файл с решением можно считать специальной процедурой пакета Pyomo, для оформления результата и/или подготовки исходных данных новых задач математического программирования. Другие технические подробности взаимодействия с решателями, подключенными к системе Everest изложены в статье [17].

VI. Заключение

В работе изложена методика регуляризованной идентификации моделей на основе обработки набора измерений характеристик моделируемого явления (объекта). Применение методики к различным моделям позволяет количественно сравнить качество моделей с точки зрения соответствия измерениям, содержащих заранее неизвестные погрешности.

Приведенные результаты моделирования динамики осциллятора с вязким трением подтверждают эффективность предложенных методов выбора математических моделей, оценки значимости гипотез и обработки экспериментальных данных. Применение метода выбора модели оптимально соответствующей измерениям и соответствующий анализ погрешностей позволяют не только отсеять ошибки измерений и рассчитать параметры процесса, но и оценить точность моделирования процесса в целом. Величина погрешности моделирования измерений может рассматриваться как численный критерий соответствия модели экспериментальным данным (качества модели) и может использоваться в качестве одного из критериев выбора модели

Практическое применение метода основано на замене вариационных задач на их разностные аппроксимации. Поскольку здесь требуется решать наборы независимых задач математического программирования (для каждой подзадачи перекрестной верификации), то работа алгоритма может быть ускорена за счет одновременного решения указанных задач пулом решателей, установленных в распределенной вычислительной среде.

Эта вычислительная схема характеризуется итеративным решением относительно небольшого набора (десятки) независимых и вычислительноемких задач математического программирования. Для ее реализации в режиме распределенных вычислений предлагается использовать систему Pyomo-Everest, https://github.com/distcomp/pyomo-everest. Решение всех задач выполняется сервисом оптимизации Everest, причем независимые подзадачи могут решаться параллельно под управлением службы балансировки вычислительной нагрузки Everest на ресурсы, подключенных к сервису.

Предложенный подход указывает перспективное направление применения распределенных систем на основе высокоуровневых средств оптимизационного моделирования.

Библиография

[1] A. Makroglou, J. Li, Y. Kuang. “Mathematical models and software tools for the glucose-insulin regulatory system and diabetes: an overview”, in *Proceedings of the 2005 IMACS*, pp. 561 – 565

[2] Гоменюк С.М., Емельянов А.О., профессор, д.ф.-м.н. Карпенко А.П., Чернецов С.А. “Методы прогнозирования оптимальных доз инсулина для больных сахарным диабетом I типа. Обзор”, *Электронное издание “НАУКА и ОБРАЗОВАНИЕ”*. Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл № ФС 77 - 48211. ISSN 1994-0408

[3] Тихонов А.Н. “О математических методах автоматизации обработки наблюдений”, *в сб. Проблемы вычислительной математики.* М.: Изд-во МГУ, 1980. стр 3-17.

[4] Роженко, А.И. Теория и алгоритмы вариационной сплайн-аппроксимации. Новосибирск: Изд. ИВМиМГ СО РАН (2005)

[5] Хардле В. Прикладная непараметрическая регрессия. М.: Мир, 1993. 349 с.

[6] Kuhn, Max, and Kjell Johnson. “Applied predictive modeling”. Vol. 26. New York: Springer, 2013. http://appliedpredictivemodeling.com/, doi 10.1007/978-1-4614-6849-3

[7] Hastie T., Tibshirani R., and Friedman J. Unsupervised learning. In: The elements of statistical learning. New-York: Springer, 2009. P. 485-585. DOI:10.1007/978-0-387-84858-7

[8] Sukhoroslov, O., Volkov, S., Afanasiev, A. “A Web-Based Platform for Publication and Distributed Execution of Computing Applications”, in *Parallel and Distributed Computing*, 14th International Symposium on IEEE, pp. 175-184, (2015)

[9] Самарский А.А., Михайлов А.П. Математическое моделирование. Идеи. Методы. Примеры. - 2-е изд., испр. - М.: Физматлит, 2001. - 320 с. - ISBN 5=9221-0120-X.

[10] Forrester, A., Sobester, A., Keane, A.: Engineering Design via Surrogate Modeling. A Practical Guide. Wiley, New York (2008), DOI 10.1002/9780470770801

[11] Беляев М.Г., Любин А.Д. "Особенности оптимизационной задачи, возникающей при построении аппроксимации многомерной зависимости" в *Тр. конф.“Информационные технологии и системы”(ИТиС’11)*, 2011, с. 415--422, http://itas2011.iitp.ru/pdf/1569478557.pdf

[12] Мину М. Математическое программирование. Теория и алгоритмы. М.: Наука (1990)

[13] Пшеничный, Б.Н. Метод линеаризации. М.: Наука (1983)

[14] Hart, W.E., Laird, C., Watson, J.-P., Woodruff D.L. *Pyomo–optimization modeling in python, vol. 67*: Springer, 238 p., (2012)

[15] Gay, David M. "Writing. nl files", *in Optimization and Uncertainty Estimation* (2005).

[16] Smirnov, S., Voloshinov, V., Sukhosroslov, O. “Distributed Optimization on the Base of AMPL Modeling Language and Everest Platform”, *in Procedia Computer Science*, vol. 101, pp. 313-322 (2016)

[17] Afanasiev A.P., Sokolov A.V., Voloshinov V.V. “Inverse Problem in the Modeling on the Basis of Regularization and Distributed Computing in the Everest Environment” *in Data Analytics and Management in Data Intensive Domains: Collection of Scientific Papers of the XIX International Conference DAMDID / RCDL’2017 (October 10–13, 2017, Moscow, Russia). Eds. L. A. Kalinichenko, etc.* — Moscow: FRC CSC RAS, с. 132-140, (2017) http://damdid2017.frccsc.ru/files/DAMDID_RCDL_2017_Proceedings.pdf

Choice of mathematical model: balance between complexity and proximity to measurements

A.V. Sokolov, V.V. Voloshinov

Abstract—Mathematical model selection method on the basis of a balance between the complexity and the experimental data fitting accuracy is proposed. The method is based on: 1) a set of (parametric) families of models suitable for satisfactory reproduction of measurements; 2) model complexity notion formalization (for the selected family of models); 3) a cross-validation procedure for estimating the error of data modeling; 4) search for the optimal trade-off between the complexity of the model and proximity to measurements based on minimizing the cross-validation error of data modeling. The procedure is explained by the demonstrative case study. General mathematical statement of model evaluation problem is presented. The issues of software implementation in a distributed computing environment are discussed.

***Keywords* — identification, inverse problems, regularization, distributed computing, Everest platform.**