ИНСТРУМЕНТАЛЬНАЯ ПОДДЕРЖКА МОДЕЛИРОВАНИЯ ДИНАМИКИ САМОРАЗВИВАЮЩИХСЯ СИСТЕМ

В.Б. Гусев, П.В. Куракин (Москва)

Введение

К настоящему времени на рынке появилось довольно много *универсальных* пакетов профессиональных математических расчетов. Однако, как показывает практика, они удовлетворяют далеко не все актуальные потребности, и требуется разработка специализированных программных средств – но на основе готовых библиотек и платформ.

В частности, возникает необходимость разработки, в интересах различных государственных служб и управляющих органов, не просто отдельных математических моделей социально – экономических (шире - *саморазвивающихся*) систем, а целых *семейств* таких моделей с разными областями применимости. Кроме этого, необходимы инструменты, которые позволяли бы специалистам этих служб и органов не только тестировать и применять предложенные им модели в разных режимах и ситуациях, но и самостоятельно формировать такие модели в определенной вычислительной среде. Прикладное программное обеспечение (ППО) такого рода уместно назвать «конструктором моделей». Такое ППО уместно отнести к классу систем поддержки принятия решений (СППР).

К конструктору моделей предъявляются несколько иные требования, нежели к представителям традиционных систем численных расчетов. В частности, для конструктора скорость математических вычислений не является определяющим показателем, поскольку речь идет об интеллектуальном средстве, позволяющем пользователю, не являющимся профессиональным математиком и инженером, *в первую очередь формировать* требуемые ему динамические модели, описывающие знакомую ему социально – экономическую реальность (возможно – лишь при опосредованной помощи профессионального математика). Иными словами, для предполагаемого пользователя важнее всего понятным ему способом описать взаимосвязи между количественными показателями в его предметной области и провести вычислительные эксперименты для разных вариантов этих взаимосвязей.

Скорость вычислений и оптимизацию вычислительных алгоритмов следует рассматривать как показатели эффективности только на следующих этапах разработки ППО этого класса.

В докладе описывается класс математических моделей, для которого в ИПУ им. В.А. Трапезникова РАН разрабатывается указанный конструктор, а также принципы разработки такого конструктора.

Формализм саморазвивающихся систем

Рассмотрим возможные формализованные подходы к анализу рассматриваемого класса систем [1].

Под *моделью динамической системы* будем понимать совокупность

$$<\mathbf{Y},\mathbf{P},\mathbf{U},\mathbf{X},\mathbf{T},\Omega>,$$

где $\mathbf{Y}$ – множество значений динамического вектора состояния системы, $\mathbf{P}$ – множество значений вектора параметров системы, $\mathbf{U}$ – множество значений векторов управления, $\mathbf{X}$ – множество значений вектора состояния внешней среды, $\mathbf{T}$ – множество значений параметра модельного времени, Ω – оператор действия системы

$$\mathbf{Z} \xrightarrow{\Omega} \mathbf{Y}.$$

Здесь $\mathbf{Z}$ – множество условий функционирования – прямое произведение множеств $\mathbf{X}, \mathbf{P}, \mathbf{U}$

$$\mathbf{Z} = \mathbf{X} \otimes \mathbf{P} \otimes \mathbf{U}.$$

Под вектором состояния системы понимается векторная функция времени $\mathbf{y}(t) \in \mathbf{Y}$, под вектором параметров – числовой вектор $\mathbf{p} \in \mathbf{P}$, под вектором управлений – векторная функция времени $\mathbf{u}(t) \in \mathbf{U}$, под вектором состояния внешней среды – векторная функция времени $\mathbf{x}(t) \in \mathbf{X}$, $t \in \mathbf{T}$.

Множества $\mathbf{Y}, \mathbf{U}, \mathbf{X}$ принадлежат соответствующим нормированным функциональным пространствам; $\mathbf{P}$ принадлежит многомерному, а $\mathbf{T}$ – одномерному Евклидову пространству.

Если отображение, задаваемое оператором Ω, является однозначным, а вектор состояния внешней среды $\mathbf{x}(t)$ представляет детерминированную зависимость от параметра времени t, такую модель будем называть *детерминированной*.

Примерами недетерминированной модели являются: модель, включающая стохастический вектор состояния внешней среды $\mathbf{x}(t)$, модель с неоднозначным оператором Ω.

Если $\mathbf{T}$ – дискретное множество, то модель будем называть *дискретной*.

Функционирование системы задает ее формализованное описание, на основании которого определяется порядок вычислений вектора состояния системы $\mathbf{y}(t) \in \mathbf{Y}$ при численной реализации оператора действия Ω.

Назовем оператор Θ_ε ε*-вычислимым* относительно оператора Ω, если при $t \in \mathbf{T}$ для заданного $\varepsilon > 0$ имеет место представление

$$\mathbf{y}_\Theta(t) = \Theta_\varepsilon(\mathbf{p}, \mathbf{x}(\tau), \mathbf{u}(\tau), t), \tau < t,$$

и для $\mathbf{y}_\Theta(t)$ выполняется неравенство

$$\| \mathbf{y}_\Theta(t) - \mathbf{y}(t) \| < \varepsilon.$$

Назовем семейство операторов Θ_Ω, *вычислимым* относительно оператора Ω, если при $t \in \mathbf{T}$ и для любого $\varepsilon > 0$ найдется оператор $\Theta_\varepsilon \in \Theta_\Omega$, что выполняется неравенство

$$\| \Theta_\varepsilon(\mathbf{p}, \mathbf{x}(\tau), \mathbf{u}(\tau), t) - \mathbf{y}(t) \| < \varepsilon.$$

К операторам такого семейства относятся, например, операторы, представленные задачей Коши для системы обыкновенных дифференциальных и алгебраических уравнений с начальными условиями, включенными в вектор параметров $\mathbf{p}$.

Будем говорить, что управление $\mathbf{u}(t)$ является *вычислимым* (вычислимой функцией времени), если его можно представить как результат однозначного отображения Ψ тройки векторов $< \mathbf{y}, \mathbf{x}, \mathbf{p} >$ на $\mathbf{U}$, где функции $\mathbf{x}(\tau)$, $\mathbf{y}(\tau)$ заданы на подмножестве $\mathbf{T}$, предшествующем t, т.е.

$$\mathbf{u}(t) = \Psi(\mathbf{y}(\tau), \mathbf{p}, \mathbf{x}(\tau)), \tau \le t.$$

Процедура вычисления вектора управлений $\mathbf{u}(t)$ может использовать самые различные подходы – от датчика случайных чисел до оптимизационных алгоритмов, нацеленных на поиск экстремума для *целевого* (характеристического) *функционала*

$F(\mathbf{y}(\tau),\mathbf{p},\mathbf{x}(\tau)),\tau\in\mathbf{T}$. В последнем случае вычисление отображения Ψ проводится по следующей схеме

$$\mathbf{u}(t)=\arg\operatorname{extr}F(\Theta(\mathbf{p},\mathbf{x}(\tau'),\mathbf{u}(\tau'),\tau),\mathbf{p},\mathbf{x}(\tau),\mathbf{p},\mathbf{x}(\tau)),\ t\le\tau',\tau'\le\tau,\ \tau\in\mathbf{T},\ \Theta\in\Theta_\Omega,$$

что требует знания вектора состояния внешней среды на всем множестве времени **T** и редко реализуется на практике.

Рассчитанное по этой схеме управление в идеальном случае, когда известно значение вектора состояния внешней среды $\mathbf{x}(t)\in\mathbf{X}$, $t\in\mathbf{T}$, удовлетворяет уравнению рефлексии, отображающему действие обратной связи по управлению в рассматриваемой системе

$$\mathbf{u}(t)=\Psi(\Theta(\mathbf{p},\mathbf{x}(\tau),\mathbf{u}(\tau),t),\mathbf{p},\mathbf{x}(\tau)),\tau\le t,\ \Theta\in\Theta_\Omega, \qquad (1)$$

В общем случае для отображения Ψ это уравнение не выполняется. Однако уравнение рефлексии можно применять для организации итеративного процесса расчета $\mathbf{u}(t)$. Этот процесс может, как сходиться, так и расходиться. Компоненты управления могут изменяться в процессе итераций монотонно, циклически, хаотически – в зависимости от свойств отображения Ψ.

Когда целевой функционал включает значения вектора состояния для предстоящих моментов времени, требуется знание *прогноза* вектора состояния $\mathbf{y}(t_p),\ t_p>t$, где t – текущее время. При расчете прогноза вектора состояния $\mathbf{y}(t_p),\ t_p>t$ необходимо принимать во внимание наблюдаемые значения вектора параметров **p**, вектора состояния внешней среды $\mathbf{x}(\tau)\in\mathbf{X},\tau\le t$ и прогноз последнего $\mathbf{x}(\tau)\in\mathbf{X},\ t_p\ge\tau>t$, а также прогноз вектора управления $\mathbf{u}(\tau),\ t_p\ge\tau>t$. Прогноз вектора состояния внешней среды $\mathbf{x}(\tau),\ t_p\ge\tau>t$ для саморазвивающихся систем может содержать неопределенность. Это делает неопределенным также прогноз управления $\mathbf{u}(\tau),\ t_p\ge\tau>t$ и вектора состояния $\mathbf{y}(t_p),\ t_p>t$.

Вектор состояния зависит от управления и наоборот. Таким образом, процедура вычисления вектора управлений использует результаты расчета прогноза вектора состояния, получаемого в результате решения уравнения

$$\mathbf{y}(t_p)=\Theta(\mathbf{p},\mathbf{x}(\tau),\Psi(\mathbf{y}(\tau),\mathbf{p},\mathbf{x}(\tau))),\ t_p\ge\tau>t,\ \Theta\in\Theta_\Omega,$$

которое учитывает рефлексию между векторами состояния и управления и может быть применено в качестве схемы итеративного процесса уточнения прогноза для вектора состояния.

Вычислимое управление $\mathbf{u}(t)$ будем называть автономным, если оно определяется только на основании значений состояния системы $\mathbf{y}(\tau)$, т.е. может быть представлено как результат отображения

$$\mathbf{u}(t)=\Psi^a(\mathbf{y}(\tau)),\tau\in\mathbf{T}.$$

Под моделью автономной саморазвивающейся системы будем понимать модель динамической системы воспроизводства с автономным управлением.

$$\mathbf{y}(t)=\Theta(\mathbf{p},\mathbf{x}(\tau),\Psi^a(\mathbf{y}(\tau))),t>\tau,\ \Theta\in\Theta_\Omega.$$

Характерное общее свойство моделей автономных саморазвивающихся систем состоит в строгой вложенности множества состояний внешней среды, для которого выполняются условия воспроизводства (область воспроизводства $\overline{\mathrm{X}}\subset\mathbf{X}$ может составлять малую часть пространства состояний внешней среды).

Из сказанного следует, что прогноз для саморазвивающихся систем имеет ограниченную точность, а вычислимое управление в общем случае приводит к состояниям, не совпадающим с целевыми и не гарантирует устойчивого поведения системы. Проблема заключается в выборе механизмов вычислимого управления, не обладающих указанными недостатками.

Принципы вычислительной архитектуры «конструктора моделей»

В основу создаваемого «конструктора моделей» положены принципы модульности и итеративного конфигурирования модели пользователем конструктора.

Модульность означает, что любая математическая модель в рассматриваемом классе формируется из элементарных моделей или *микромоделей*. Принято решение считать в качестве элементарной модели либо одну динамическую переменную (в терминах предыдущего раздела, одну составляющую вектора **y**), либо одно вычислительное правило (входящее в множество Ω, в терминах предыдущего раздела). К этой элементарной модели относятся:

1. массив числовых данных, описывающих уже рассчитанную траекторию (временной ряд) этой переменной для прошедшего (с точки зрения моделирования) промежутка модельного времени;

2. массив или массивы числовых данных, описывающих некоторый прогноз динамики этой переменной для будущих промежутков модельного времени;

3. если переменная является стохастической, то в микромодель входит тот или иной статистический закон, т.е. правило генерации случайной траектории; это правило оформляется в виде символьного выражения (технические детали см. в разделе 4);

4. если переменная претерпевает автоэволюцию, должно быть указано вычислительное правило (как правило – рекуррентное), которое связывает новые значения временного ряда переменной с предыдущими – также, в виде символьного выражения;

5. если в модели всей системы (макромодели) присутствует динамическое правило, связывающее изменение данной динамической переменной со значениями других динамических переменных, это правило также должно быть описано в виде символьного выражения;

6. если в макромодели присутствуют функциональные связи между динамическими переменными (они могут рассматриваться как динамические связи, ограничения, так и форма прогноза на будущее), то эти зависимости также должны быть оформлены в виде символьного выражения.

Как можно видеть, это описание соответствует динамическим моделям, которые в явном виде являются итерационными (например – конечно-разностными), что часто применяется в социально-экономическом моделировании.

Выбор программной платформы

Во введении было отмечено, что *универсальные* пакеты профессиональных математических расчетов часто оказываются недостаточными для многих актуальных задач. Причины этого в следующем:

1. универсальные пакеты рассчитаны на математиков или инженеров, знакомых с численными методами и основами программирования;

2. существуют подсистемы таких пакетов с графическим интерфейсом описания моделей (Simulink для MATLAB), но они, как правило, рассчитаны на некоторый типовой набор стандартных физико-технических задач;

3. наиболее популярные универсальные пакеты являются коммерческими.

В связи с этим нами был сделан следующий выбор. В основу разработки конструктора моделей положена платформа Python [2]:

1. это бесплатная среда;

2. язык программирования достаточно простой;

3. имеется множество расширяющих библиотек, в частности - библиотеки численных расчетов NumPy и SciPy; эти библиотеки разработаны представителями научного сообщества (NASA, Los Alamos National Laboratory, CERN) как эквивалентная замена коммерческим продуктам типа MATLAB; программирование ведется на языке высокого уровня, а непосредственно используемые численные методы скомпилированы до уровня команд процессора для повышения скорости вычислений;

4. имеются библиотека символьных вычислений SymPy; благодаря ей возможна указанная в предыдущем разделе опция задания пользователем алгебраического соотношения в качестве микромодели;

5. имеется библиотека математической графики PyPlot математической графики, по изобразительной мощности сопоставимая с возможностями графической подсистемы пакета MATLAB;

6. имеется набор библиотек для создания графического интерфейса пользователя, в котором связями между отдельными переменными можно задавать визуально.

Следует отметить, что пользователю удобнее графический интерфейс, но с точки зрения разработчика уместнее, на начальных этапах разработки, использовать инструменты структурированного текстового описания взаимосвязей микромоделей. Наиболее подходящим языком такого форматированного описания, по нашему опыту, является стандарт JSON [3, 4].

Таким образом, платформа Python вполне соответствует требованиям к описанному типу ППО.

Литература

1. Гусев В.Б. Модели систем с автономным управлением. – М.: ИПУ РАН, 2014. – 284 с.
2. Программирование и научные вычисления на языке Python (https://ru.wikiversity.org/wiki/Программирование_и_научные_вычисления_на_языке_Python).
3. *Р.Д. Зухба, П.В. Куракин, Г.Г. Малинецкий, С.А. Махов, Н.А. Митин, А.П. Сорокин*. «Программно-математические комплексы систем поддержки принятия решений нового поколения». Препринт ИПМ им. М. В. Келдыша РАН № 59 за 2014 г.
4. JSON. Статья Wikipedia - https://ru.wikipedia.org/wiki/JSON