УДК 519.622+539.376
DOI: 10.25559/SITITO.14.201803.521-532

МЕТОДЫ СОЗДАНИЯ ЦИФРОВЫХ ДВОЙНИКОВ НА ОСНОВЕ НЕЙРОСЕТЕВОГО МОДЕЛИРОВАНИЯ

А.Н. Васильев, Д.А. Тархов, Г.Ф. Малыхина
Санкт-Петербургский политехнический университет Петра Великого, г. Санкт-Петербург, Россия

METHODS OF CREATING DIGITAL TWINS BASED ON NEURAL NETWORK MODELING

Alexander N. Vasilyev, Dmitry A. Tarkhov, Galina F. Malykhina
Peter the Great St. Petersburg Polytechnic University, St. Petersburg, Russia

Ключевые слова

Цифровые двойники; машинное обучение; эволюционные алгоритмы; дифференциальные уравнения; разнородные данные; нейросетевое моделирование; искусственные нейронные сети; многослойные нейронные сети.

Аннотация

Предполагается, что к 2021 году около половины компаний будут использовать цифровых двойников разного уровня. Самые простые цифровые модели-двойники могут не использовать машинное обучение, но наибольшее преимущество будут иметь модели, использующие алгоритмы машинного обучения. В данной статье мы предлагаем свой подход к построению цифровых двойников реальных объектов. Мы опираемся на свой унифицированный процесс построения приближённых решений краевых задач для уравнений математической физики и накопленный опыт решения большого числа конкретных задач такого типа. В работе мы представляем пять подходов к построению цифровых моделей-двойников, опирающиеся на разработанные и протестированные нами эволюционные алгоритмы. Особенностью нашего подхода к эволюционным алгоритмам является использование генетических процедур для построения структуры модели и алгоритмов нелинейной оптимизации для подстройки ее параметров. Кроме того, мы предлагаем наш подход к построению многослойных моделей по дифференциальным уравнениям, позволяющий обойтись без трудоёмкой процедуры обучения нейронных сетей. Мы уверены, что предложенные нами подходы позволяют существенно упростить и унифицировать создание и адаптацию (поддержание в актуальном состоянии) цифровых двойников реальных объектов разного рода – технических, биологических, социально-экономических и т.д.

Keywords

Digital twins; machine learning; evolutionary algorithms; differential equations; heterogeneous data; neural network modeling; artificial neural networks; multilayer neural networks.

Abstract

It is assumed that by 2021, about half of the companies will use digital counterparts of different levels. The simplest digital twin models may not use machine learning, but the models using machine learning algorithms will have the greatest advantage. In this article, we offer our approach to the construction of digital twins for real objects. We rely on our unified process of constructing approximate solutions of boundary value problems for equations of mathematical physics and accumulated experience in solving numerous specific problems of this type. In this paper, we present five approaches to the construction of digital twin models based on the evolutionary algorithms developed and tested by us. The peculiarity of our approach to evolutionary algorithms is the use of genetic procedures for constructing the structure of the model and nonlinear optimization algorithms for adjusting its parameters. Also, we propose our approach to the construction of multilayer models upon differential equations, which allows doing without the time-consuming procedure of neural networks training. We are confident that the proposed approaches can significantly simplify and unify the creation and adaptation (keeping up to date) of digital twins for real objects of various kinds – technical, biological, socio-economic, etc.

Об авторах:

Васильев Александр Николаевич, доктор технических наук, доцент, профессор кафедры высшая математика, Институт прикладной математики и механики, Санкт-Петербургский политехнический университет Петра Великого (195251, Россия, г. Санкт-Петербург, ул. Политехническая, д. 29), ORCID: http://orcid.org/0000-0003-0227-0162, a.n.vasilyev@gmail.com

Тархов Дмитрий Альбертович, доктор технических наук, доцент, профессор кафедры высшая математика, Институт прикладной математики и механики, Санкт-Петербургский политехнический университет Петра Великого (195251, Россия, г. Санкт-Петербург, ул. Политехническая, д. 29), ORCID: http://orcid.org/0000-0002-9431-8241, dtarkhov@gmail.com

Малыхина Галина Федоровна, доктор технических наук, профессор, профессор кафедры измерительные информационные технологии, Институт компьютерных наук и технологий, Санкт-Петербургский политехнический университет Петра Великого (195251, Россия, г. Санкт-Петербург, ул. Политехническая, д. 29), ORCID: http://orcid.org/0000-0002-1026-8727, gfmalykhina@gmail.com

Введение

В современной индустрии человек становится элементом производственной технологии, снижающим производительность, надежность и качество продукции. Развитие киберфизических систем, интернета вещей, облачных вычислений и когнитивных вычислений позволяет повысить уровень автоматизации, заменить человека на сложных участках производства. Об этом переходе принято говорить как о четвертой промышленной революции, которая приводит к цифровому производству. Информационные потоки между сенсорами, устройствами и людьми замыкаются через Интернет вещей и людей. Модели цифрового производства агрегируют низкоуровневые данные от сенсоров с высокоуровневой контекстной информацией. На основе этой информации киберфизические системы принимают решения возможно более автономно и, в то же время, предоставляют человеку-оператору агрегированную информацию в удобной визуальной форме. Технологии цифрового производства объединяют технологии искусственного интеллекта, компьютерного моделирования, машинного обучения, облачных вычислений, киберфизических систем.

Компьютерная модель, которая копирует поведение физического объекта, называется *цифровым двойником* [1-3]. Модель симулирует все возможные режимы работы объекта в течение его жизни, учитывает влияние внешних факторов и процессов управления, позволяет предсказывать будущее состояние и поведение физического объекта. Цифровой двойник основан на технологиях искусственного интеллекта, машинного обучения и *аналитического* программирования. Цифровой двойник непрерывно обучается и обновляет свои параметры, получая информацию от множества сенсоров, правильно представляет состояние физического объекта. При обучении им используются текущие данные от сенсоров, от устройств управления, от внешней среды, он объединяет фактические данные со знаниями, полученными от инженеров, опытных специалистов в данной области. Цифровой двойник использует исторические данные, накопленные на предыдущих этапах.

Цифровые модели-двойники используются в интеллектуальных системах диагностики и обслуживания, чтобы предсказать отказ, выдать сообщение в центр интеллектуального обслуживания. Центры могут использовать облачные сервисы [4,5], их целью является достижение близкой к нулю вероятности отказа. Для этого используют агентов для предсказания отказов устройств и для обеспечения их готовности. Модели учитывают метрологическую надежность сенсоров, предсказывают необходимость метрологического обслуживания (поверки – calibration of instrument). Центры интеллектуального обслуживания целесообразно строить как объединенные промышленно-университетские, использующие облачные кластеры суперкомпьютерного центра.

Искусственный интеллект в моделях цифровых двойников представлен традиционными методами обучения нейронных сетей RBF, MLP, NARX и глубокими методами обучения нейронных сетей CNN и RNN, специально предназначенных для обработки видео- и аудио- сигналов.

Традиционно для создания цифровых моделей физических объектов применяют методы математической физики. Эти методы используются при анализе прочности конструкций, в теплопередаче, в гидродинамике, в электроэнергетике и др. Сложная задача моделирования физического объекта представляется в виде набора краевых задач для обыкновенных дифференциальных уравнений и (или) дифференциальных уравнений в частных производных, например, для уравнений Эйлера – Лагранжа, Навье – Стокса.

К сожалению, традиционные подходы к решению таких задач – сеточные методы и методы конечных элементов [6-20] не обладают рядом важные свойств, необходимых для построения цифровых двойников:

- невозможность непрерывного обучения в течение всего жизненного цикла производственного оборудования;
- невозможность пред-обучения и пост-обучения модели;
- невозможность адаптации модели в результате пополнения данных от сенсоров;
- невозможность обработки текущей информации от всех видео- и аудио- данных;
- невозможность решения задач распознавания и диагностики;
- невозможность предсказания отказов для их предотвращения;
- не учитывают метрологические характеристики сенсоров;
- не способны, основываясь на наблюдениях, автоматически изменять структуру модели, изменять и подстраивать сетки и элементы.

Важно в области инженерии, обновлять модели «на лету», поскольку новые данные становятся доступными для лучшего контроля. Инженерные системы строго необходимы в персонализированной медицине, где все пациенты различны, а эксперименты in vivo невозможны. В этой области необходимо вывести наилучшую возможную модель для пациента из априорных знаний, полученных от других пациентов. Недавно были опубликованы успешные подходы, которые позволяют прогностическую науку в медицине, например, для лазерной обработки опухолей. Появляется область – «компьютерная предсказательная медицина» для приложений к персонализации модели опухоли головного мозга [21-24].

При моделировании сложных технических объектов обычно применяются пакеты компьютерного инженерного анализа, основанные на методе конечных элементов (МКЭ). Однако моделирование реального объекта с их помощью наталкивается на ряд принципиальных трудностей. Во-первых, точная информация о дифференциальных уравнениях, описывающих поведение объекта, обычно отсутствует в силу сложности описания происходящих в нём процессов. Во-вторых, для применения МКЭ нужно знать начальные и граничные условия, информация о которых обычно является ещё менее полной и точной. В-третьих, при работе реального объекта его свойства и характеристики, параметры протекающих в нём процессов могут меняться. Это требует соответствующей адаптации модели, что затруднительно провести с моделью, построенной на основе МКЭ.

Более перспективным представляется другой подход, когда для каждого элемента целевого объекта строится адаптивная модель, которая может уточняться и перестраиваться в соответствии с данными наблюдений над объектом. Такую модель удобнее всего строить в виде нейронной сети. Поэтому актуальной задачей является развитие технологий нейросетевого моделирования, более полный учет исторических и вновь поступающих данных, совершенствование методов автоматической

подстройки архитектуры и параметров модели, методов классификации и предсказания. Совокупность задач, стоящих перед цифровой моделью-двойником, может быть решена с использованием коллектива нейронных сетей, каждая из которых отображает некоторый фрагмент (элемент, процесс) объекта.

Методы

Поясним процедуру построения нейросетевой модели некоторого элемента (процесса) в объекте, для которого мы строим модель-двойник. Пусть соответствующий элемент (процесс) описан в виде начальной или краевой задачи для дифференциального, интегро-дифференциального или диференциально-алгебраического уравнения (обыкновенного или в частных производных)

$$A(u) = g, \quad u = u(\mathbf{x}), \quad \mathbf{x} \in \Omega \subset R^p, \; B(u)\Big|_\Gamma = h, \tag{1}$$

здесь u – функция, характеризующая состояние интересующего нас элемента или процесса, $A(u)$ – дифференциальный, интегро-дифференциальный или дифференциально-алгебраический оператор, т.е. алгебраическое выражение, содержащее производные, интегралы или алгебраические соотношения от функции u, $B(u)$ – оператор, определяемый граничными условиями, Γ – граница области Ω.

Ищем приближённое решение задачи (1) в виде выхода искусственной нейронной сети (ИНС) заданной архитектуры

$$u(\mathbf{x}, \mathbf{w}) = \sum_{i=1}^{N} c_i v(\mathbf{x}, \mathbf{a}_i) \tag{2}$$

Здесь $\mathbf{w} = (\mathbf{w}_1, ..., \mathbf{w}_N)$ – вектор весов $\mathbf{w}_i = (c_i, \mathbf{a}_i)$, агрегирующий линейно входящие параметры c_i и нелинейно-входящие параметры $\mathbf{a}_i$. Базисный нейроэлемент – функция v задается выбором типом нейронной сети и функции активации.

Иногда целесообразно применение ИНС с разными базисными элементами (гетерогенных ИНС). Обычно подобные сети применяются тогда, когда искомое решение ведёт себя принципиально различно в разных подобластях, например, в задачах с фазовыми переходами, скачками и т.д. В такой ситуации часто имеет смысл строить две или несколько нейронных сетей меньшего размера, каждая из которых отвечает своей подобласти.

Вектор весов ИНС $\mathbf{w}$ находится в процессе поэтапного обучения сети, построенном в общем случае на минимизации некоторого функционала ошибки $J(\mathbf{w})$. Для задачи (1) $J(\mathbf{w}) = J_1 + \delta J_2$, где J_1 выражает оценку удовлетворения уравнения, J_2 - краевого условия. В компонент J_2 могут входить слагаемые, которые отвечают и за иные условия постановки: симметрия, соотношения на границах раздела, уравнение состояния и т.д. Так как цифровой двойник должен учитывать данные наблюдений за целевым объектом, в функционал $J(\mathbf{w})$ добавляется слагаемое $\tilde{\delta} J_3$, отражающее точность, с которой модель (2) соответствует этим условиям.

Хотелось бы взять слагаемые функционала $J_1 = \int_\Omega (A(u) - g)^2 \, d\Omega$, $J_2 = \int_\Gamma (B(u) - f)^2 \, d\Gamma$,

но вычислить аналитически соответствующие интегралы удаётся только в исключительных случаях, поэтому удобнее использовать дискретную форму для функционалов

$$J_1 = \sum_{j=1}^{M} (A(u(\xi_j)) - g(\xi_j))^2, \quad J_2 = \sum_{k=1}^{K} (B(u(\xi'_k)) - h(\xi'_k))^2,$$

$$J_3 = \sum_{k=1}^{L} (u(\xi''_k) - u''_k))^2 \tag{3}$$

Здесь u''_k - результаты наблюдений за объектом, соответствующие точкам $J(\mathbf{w})$. В связи с тем, что исходные уравнения (1) соответствуют моделируемому объекту неточно, мы ищем не точный минимум функционала $J(\mathbf{w})$, а точку $\mathbf{w}_\eta$ в пространстве весов $J(\mathbf{w}_\eta) < \eta$, задающее так называемое η -решение $u_\eta = u(\mathbf{x}, \mathbf{w}_\eta)$. Число $\eta > 0$ выбирается таким, чтобы считать построенную модель достаточно точной. Оно может быть выбрано тем меньше, чем точнее исходная модель (1).

Вычислительные эксперименты показали, что использование фиксированного набора тестовых (пробных) точек $\{\xi_j\}_{j=1}^{M}$ нецелесообразно, так как в этом случае малые ошибки в этих пробных точках могут сопровождаться большими ошибками в других точках области Ω. Увеличение числа пробных точек с целью достаточно плотного покрытия области Ω позволяет избегнуть подобной ситуации в случае относительно небольшого числа нейронов (слагаемых в выражении (2)). При этом резко возрастает время вычислений, что существенно в ситуации, когда мы хотим моделировать объект в режиме реального времени. Решением этой проблемы оказалось использование периодически перегенерируемых множеств пробных точек $\{\xi_t\}$ в области Ω и, при необходимости, перегенерируемых множеств пробных точек $\{\xi'_k\}$ на её границе Γ. Перегенерация тестовых точек после определённого числа шагов процесса обучения сети делает его более устойчивым. При этом мы организуем вычисления как процесс минимизации набора функционалов, каждый из которых получается конкретным выбором тестовых точек и минимизируется не до конца (между перегенерациями тестового множества производится только несколько шагов выбранного метода минимизации). Такой подход, в частности, позволяет обойти проблему попадания в локальный экстремум, характерную для большинства методов глобальной нелинейной оптимизации. В процессе оптимизации функционала ошибки мы можем включать новые наблюдения как дополнительные слагаемые в сумме J_3. Если необходимо учесть устаревание информации со временем, то можно выбрасывать устаревшие наблюдения или использовать формулу

$$J_3 = \sum_{k=1}^{L} \delta_k (u(\xi''_k) - u''_k))^2$$

с переменными δ_k.

В большинстве ситуаций целесообразным оказалось случайное распределение тестовых точек, генерируемое через определённое число эпох обучения (шагов оптимизации) с помощью постоянной (или иной) плотности вероятности, что обеспечивает более устойчивый ход обучения. В некоторых случаях целесообразно использовать неравномерный закон распределения тестовых точек – сгущение их вблизи особенностей (границ разрыва, углов и т.д.) или в зонах с большими ошибками. Возможна также перегенерация лишь подмножества множества пробных точек. Число тестовых точек является компромиссом между требованием точности и устойчивости вычислительного процесса (чем больше точек, тем лучше эти показатели) и ограниченным временем на построение и адаптацию модели (для цифрового двойника эти процессы должны идти в реальном времени в процессе его эксплуатации).

Нестационарные задачи (в частности, начально-краевые задачи) могут быть рассмотрены в рамках данного подхода из-

менением размерности пространства – включением времени в число переменных. Но есть и другие ИНС подходы, в частности, применение динамических нейронных сетей [25].

Часто цифровой двойник должен содержать параметры, изменяющиеся в некоторой области. Например, плотность, коэффициенты теплопроводности, упругости и т.д. реальных сред могут не быть известны точно. В процессе функционирования целевого объекта могут варьироваться его размеры, температура окружающей среды и т.д. Часто требуется исследовать поведение цифрового двойника в зависимости от некоторого параметра, идентифицировать значение параметра по данным измерений или когда определяющие моделируемую систему характеристики заданы значениями, распределёнными в некоторых интервалах, – интервальными параметрами.

Классический подход предполагает в подобной ситуации численное решение задачи для достаточно представительного набора параметров. Это часто требует слишком больших вычислительных ресурсов и не может быть проведено в режиме реального времени. Предлагаемый нами подход позволяет искать решение в виде нейросетевой функции, для которой указанные параметры входят в число аргументов. Такую функцию можно построить заранее, для конкретного двойника реального объекта останется идентифицировать параметры по данным измерений.

Поясним суть нашего подхода на краевой задаче (1). Модификация формулировки задачи состоит в том, что в её постановку входят параметры $\mathbf{r} = (r_1, \ldots, r_k)$, меняющиеся на некоторых интервалах $r_i \in (r_i^-; r_i^+),\ i = 1, \ldots, k$:

$$A(u, \mathbf{r}) = g(\mathbf{r}),\ \ u = u(\mathbf{x}, \mathbf{r}),\ \ \mathbf{x} \in \Omega(\mathbf{r}) \subset R^p,\ B(u, \mathbf{r})\Big|_{\Gamma(\mathbf{r})} = f(\mathbf{r}). \tag{4}$$

Соответственно мы изменяем и представление для приближённого решения задачи. Ищем приближённое решение задачи (4) в виде выхода искусственной нейронной сети заданной архитектуры:

$$u(\mathbf{x}, \mathbf{r}) = \sum_{i=1}^{N} c_i v_i(\mathbf{x}, \mathbf{r}, \mathbf{a}_i), \tag{5}$$

веса которой определяются в процессе поэтапного обучения сети на основе минимизации функционала ошибки $J(\mathbf{w}) = J_1 + \delta J_2 + \tilde{\delta} J_3$, где

$$J_1 = \sum_{j=1}^{M} (A(u(\xi_j, \mathbf{r}_j)) - g(\xi_j, \mathbf{r}_j))^2$$

$$J_2 = \sum_{j=1}^{M'} (B(u(\xi'_j, \mathbf{r}'_j)) - f(\xi'_j, \mathbf{r}'_j))^2$$

$$J_3 = \sum_{k=1}^{L} (u(\xi''_k, r''_k) - u''_k))^2 \tag{6}$$

Здесь, как и ранее, $\{\xi_j, \mathbf{r}_j\}_{j=1}^{M}$ – периодически перегенерируемые пробные точки в области $\Omega(\mathbf{r}_j) \times \prod_{i=1}^{k} (r_i^-; r_i^+)$; $\{\mathbf{x}'_j, \mathbf{r}'_j\}_{j=1}^{M'}$ – пробные точки на её границе $\Gamma(\mathbf{r}'_j)$; $\{\xi''_j, \mathbf{r}''_j\}_{k=1}^{L}$ - точки и значения параметров, для которых производятся наблюдения за объектом.

На основе указанной методологии разработан *унифицированный процесс* построения и поддержания в актуальном состоянии двойников реальных объектов. Данный процесс может применяться и при построении моделей, не являющихся нейросетевыми. Его основные этапы:

1. *Декомпозиция объекта* на элементы, поведение которых может быть описано достаточно простыми физическими моделями или наборами таких моделей. Предполагается, что основная часть таких моделей может быть формализована в виде дифференциальных, интегро-дифференциальных или дифференциально-алгебраических уравнений и систем.

2. Характеристика качества модели в виде функционала (набора функционалов). Данный этап основан на формализации всей имеющейся информации о изучаемых объектах и проходящих в них процессах (модели должны допускать возможность уточнения в процессе построения решения, конструирования и функционирования объекта).

3. *Выбор функционального базиса (базисов).* Данный этап должен допускать выполнение как специалистом в предметной области на основе информации об объекте, для которого строится цифровой двойник, так и автоматически, с помощью эволюционных алгоритмов (см. [26]) . Мы предлагаем использовать базисы функций, характерные для нейронных сетей [25]. Наш опыт решения различных задач математического моделирования [27-35] показал, что нейросетевые функции одних и тех же типов применимы при решении разнообразных задач, устойчивы по отношению к ошибкам в исходных данных и допускают эффективную программную и аппаратную реализацию.

4. *Выбор и реализация методов подбора параметров и структуры модели.* Данный этап может быть полностью автоматизирован на основе приведённых далее эволюционных алгоритмов. При этом имеющаяся приблизительная информация о поведении объекта, для которого строится цифровой двойник, может быть легко учтена при построении модели.

5. *Реализация методов уточнения цифровых двойников объектов в процессе их функционирования и соответствующей подстройки алгоритмов управления ими.* Важнейшим свойством цифрового двойника является способность адаптации под новую информацию, поступающую об объекте и условиях его функционирования. Изложенные далее методы позволяют строить и модифицировать такую адаптивную модель.

6. *Пополнение базы данных моделей, алгоритмов и программ создания цифровых двойников.* В процессе создания и адаптации цифрового двойника мы получаем математическую и программную модель, которая в дальнейшем может быть использована в качестве первого приближения к цифровому двойнику другого похожего объекта. Изложенные далее методы адаптивны в том смысле, что содержат настраевыемые параметры, позволяющие улучшить работу программы в процессе построения и адаптации двойника. Такая программа, настроенная на некоторый тип цифровых двойников, далее может быть использована при построении и адаптации двойников определённого типа.

Поставленные выше задачи построения элемента цифрового двойника (1) и (4) допускают следующее обобщение. Пусть задан некоторый класс параметризованных моделей $\mathbf{y} = \mathbf{f}(\mathbf{x}, \mathbf{a})$, где подбору подлежат как конкретная структура S функции $\mathbf{f}$, выбираемая из некоторого множества структур, так и векторный параметр $\mathbf{a}$, выбрав который мы получаем конкретную модель. Заметим, что выбор структуры $\mathbf{f}$ тоже можно параметризовать, но обычно удобнее подбирать её напрямую. Следует учесть, что $\mathbf{f}$ может быть и отображением самого общего вида. Будем искать модель, наилучшим образом удовлетворяющую условиям $\{A_q(\mathbf{x}, \mathbf{y}, \Omega_q) = 0\}$, где Ω_q – некоторое множество $\mathbf{x}$,

на котором соответствующее условие должно быть выполнено. При этом указанные условия могут меняться в процессе построения и адаптации цифрового двойника. В упомянутых выше постановках задач в качестве множеств Ω_q выступают точки $\mathbf{x}_i$, но возможны задачи, для которых условие должно быть выполнено для каждой отдельной точки из множества Ω_q или в интегральном смысле, т.е. для всего множества в целом.

Сформулируем соответствующий алгоритм одновременного подбора параметров и структуры совокупности моделей, составляющих цифрового двойника, более формально. Начинаем с генерации некоторого начального множества моделей $F_0 = \{\mu_i = \mu(S_i, \mathbf{a}_i, G_i)\}$, определяемых структурой S_i, весами $\mathbf{a}_i$ и множеством переменных G_i, на котором данная модель работает. В множество G_i могут входить область изменения пространственных переменных, конечный или бесконечный интервал изменения времени и множество изменения параметров, характеризующих объект, для которого строится данный элемент цифрового двойника. Кроме того, необходимо ввести множества функционалов J_j, в процессе минимизации которых и строится цифровой двойник. Эти функционалы описывают, насколько точно модели соответствуют описывающих их физическим и иным законам, условия стыка элементов, величины, которые мы хотим оптимизировать при управлении объектом. При этом функционалы могут меняться при появлении новых данных об объекте или желаемых параметров его функционирования. Алгоритм определяется зависимостью $F_j(F_{j-1}, J_j[\{A_{i,j}, \Omega_{i,j}\}])$, где $J_{j+1} = J_{j+1}(F_j, J_j, \{A_{i,j}, \Omega_{i,j}\})$, $A_{i,j+1} = A_{i,j+1}(F_j, J_j, \{A_{i,j}, \Omega_{i,j}\})$, $\Omega_{i,j+1} = \Omega_{i,j+1}(F_j, J_j, \{A_{i,j}, \Omega_{i,j}\})$. Таким образом, в процессе работы алгоритма меняется не только совокупность моделей F_j, но и перестраиваются множества функционалов J_j, условий $A_{i,j}$ и множеств $\Omega_{i,j}$, на которых эти условия заданы. Для построения реального алгоритма построения цифрового двойника реального объекта надо все эти зависимости конкретизировать. Рассмотренные далее алгоритмы являются примерами алгоритмов данного класса.

Пусть реальное и желаемое функционирование цифрового двойника определяет набор условий

$$\{A_q(u_1, u_2, ..., u_r)\big|_{\Omega_q} = 0\}_{q=1}^{Q}, \quad (7)$$

где Ω_q – некоторое множество, на котором соответствующее условие должно быть выполнено, u_s – неизвестные функции, описывающие отдельные аспекты работы моделируемого объекта или его элементов. Операторы A_q задают уравнения, а также граничные и иные условия – например, законы сохранения, уравнения состояния, условия стыковки элементов или данные, полученные из опыта, а также иные требования к решению задачи, включая параметры, которые требуется оптимизировать. В процессе обучения операторы A_q могут изменяться, например, включая в рассмотрение вновь поступающую информацию или новые требования к работе объекта.

Будем искать каждую неизвестную функцию в виде разложения:

$$u_s(\mathbf{x}) = \sum_{i=1}^{N_s} c_{i,s}\varphi_s(\mathbf{x}; \mathbf{a}_{i,s}), \; s = 1...r, \quad (8)$$

подбирая веса – параметры $\mathbf{a}_{i,s}$ и $c_{i,s}$ – путём минимизации функционала ошибки, составленного из слагаемых вида

$$\sum_{j=1}^{M_q} \left| A_q(u_1, u_2, ..., u_r)(\mathbf{x}_{j,q})\big|_{\Omega_q} \right|^2$$

каждое из которых входит в сумму с весовым множителем $\delta_q > 0$, обычно фиксируемым заранее или пересчитываемым время от времени по определённой процедуре. Возможные процедуры подбора весовых множителей оставим в стороне, упомянем только самую простую – подбирать их так, чтобы слагаемые в функционале соответствовали точности выполнения условий (7), по которым они сформированы или важностью их выполнения для объекта, для которого строится цифровой двойник. Целесообразно время от времени осуществлять пересчёт δ_q, если указанная точность или значимость условий меняется во время работы алгоритма. Задачи с параметрами (4,5) также входят в данный класс задач. Если мы строим цифрового двойника для некоторого множества параметров $\mathbf{r} = (r_1, ..., r_k)$, описывающих работу объекта, то компоненты вектора $\mathbf{r}$ включаются в число компонент вектора $\mathbf{x}$. Если параметры $\mathbf{r} = (r_1, ..., r_k)$ подлежат определению на данном этапе построения или адаптации цифрового двойника, то компоненты вектора $\mathbf{r}$ включаются в число компонент соответствующих векторов $\mathbf{a}_{i,s}$. Возможен и смешанный вариант, при котором часть параметров включаются в вектор $\mathbf{x}$, а часть – в векторы $\mathbf{a}_{i,s}$.

Для *конечных* множеств Ω_q в качестве пробных точек $\mathbf{x}_{j,q}$ могут использоваться все точки множества, а если их слишком много – на каждом шаге может выделяться их конечное подмножество (например, случайно). Пробные точки $\mathbf{x}_{j,q}$, соответствующие *бесконечным* множествам Ω_q, выбираются регулярно или случайно. Соображения, приведённые ранее, позволяют рекомендовать повторять такой выбор через определённое число шагов алгоритма оптимизации. Слагаемые в функционале не обязаны быть квадратичными, они могут быть взяты и в другой форме.

Для решения сформулированной выше оптимизационной задачи можно применить несколько принципиально различных вариантов организации алгоритма.

Во-первых, можно заранее задать структуру цифрового двойника и составить единый функционал, используя сразу все условия, и искать сразу все параметры, минимизируя этот функционал. Этот вариант весьма требователен к вычислительным ресурсам. Этот вариант заведомо нерационален для построения двойника распределённого социально-экономического или технического объекта – региона, завода и т.д.

Во-вторых, можно также заранее задать структуру цифрового двойника и составить несколько функционалов, основанных на различных наборах условий. Вычислительный процесс будет состоять из этапов, на каждом из которых подбирается часть весов, минимизируя попеременно каждый функционал. При рациональной организации расчётов такой вариант позволяет ускорить вычисления, но остаётся проблема разумного выбора структуры используемых моделей.

В-третьих, для построения цифрового двойника можно применить один из эволюционных алгоритмов, сочетающий подбор его параметров модели и структуры. Такой вариант позволяет получить наиболее точное и адекватное решение поставленной задачи. Далее обсуждаются пять подходов такого рода.

I. *Обобщённый алгоритм кластеризации ошибок*

1. Ищем набор функций вида

$$u_s(\mathbf{x}) = \sum_{i=1}^{N_s} c_{i,s}\varphi_s(\mathbf{x}; \mathbf{a}_{i,s}),$$

совершая несколько шагов минимизации функционала

$$J(u_1,u_2,...,u_r)=\sum_{q=1}^{Q}\delta_q\sum_{j=1}^{M_q}\left|A_q(u_1,u_2,...,u_r)(\mathbf{x}_{j,q})\right|_{\Omega_q}\Big|^2 .$$

В этом случае можно взять число N_s не слишком большим.

2. Для каждого набора вычисляем ошибки $z_{j,q}=A_q(u_1,u_2,...,u_r)(\mathbf{x}_j)$, при этом набор тестовых точек $\mathbf{x}_j$ из $\Omega=\bigcup_{q=1}^{Q}\Omega_q$ можно поменять по сравнению с предыдущим пунктом, главное условие – он должен быть достаточно представительным. Эти тестовые множества перегенерируются после завершения каждого этапа обучения (прохождения определенного числа шагов в процессе минимизации функционала). Если значение A_q не определено в некоторой точке $\mathbf{x}_j$, то полагаем $z_{j,q}=0$.

3. Проводим кластеризацию точек $\{(\mathbf{x}_j,z_{j,1},z_{j,2},...,z_{j,Q})\}$ в соответствующем пространстве.

4. Берём кластеры (они могут и пересекаться), строим для каждого соответствующее приближение, дающее минимальную ошибку для сужения функционала $J(u_1,u_2,...,u_r)$ на множество точек из рассматриваемого кластера. Следует заметить, что максимальные ошибки могут соответствовать стыкам элементов, оптимизация соответствующих функционалов может проводиться на разных компьютерах. В данном случае целесообразно переслать информацию о данном кластере на один компьютер и строить приближение на нём, после чего разослать соответствующие данные по всем задействованным вычислительным узлам и продолжить процедуру построения или адаптации цифрового двойника, как и ранее.

5. Добавляем к искомым функциям построенные в предыдущем пункте слагаемые и повторяем шаг 1 с получившимся набором.

6. Если функционал недостаточно мал, то пополняем популяцию наборов, приближая кластеры и применяя шаги 1-5 к новой выборке.

Если возникают проблемы с адаптацией цифрового двойника под функционирующий объект в режиме реального времени, то в п.5 можно подбор коэффициентов всей модели не проводить. Если такая модификация алгоритма оказывается недостаточной, можно отбросить часть слагаемых в суммах (8). Выбор отбрасываемых слагаемых производится из соображений наименьшего роста соответствующего функционала. Следует заметить, что предложенный метод не предъявляет особых требований ни к форме области (односвязность, возможность декомпозиции), ни к уравнению (линейность, вещественность коэффициентов). Однако усложнение формы области Ω и уравнения приводит к трудности выбора начальных значений параметров моделей, к увеличению требуемого для достижения заданной точности решения числа функций и к соответствующему замедлению процесса нелинейной оптимизации.

II. *Обобщённый метод Шварца*

Данный метод используется в ситуации, когда существует возможность декомпозиции исходного объекта на существенно более простые элементы. Это условие можно формализовать, предполагая, что область Ω распадается на подобласти $\Omega=\bigcup_{p=1}^{P}\Omega'_p$, пересекающиеся только по части границ или по множествам ненулевой меры.

1. Подобно тому, как это было реализовано при подходе I для всей области Ω, в каждой из подобластей Ω'_p строим свою аппроксимацию для решения $u_{1,p},u_{2,p},...,u_{r_p,p}$, используя при задании функционала ошибки J_p соответствующую Ω'_p часть условий

$$J_p=\sum_{q=1}^{Q_p}\delta_{q,p}\sum_{j=1}^{M_p}\left|A_q(u_{1,p},u_{2,p},...,u_{r_p,p})(\mathbf{x}_{j,q,p})\right|_{\Omega_q\cap\Omega'_p}\Big|^2 ,$$

при этом учитываются краевые условия лишь там, где они известны.

2. После определённого числа этапов обучения каждого набора нейронных сетей возникают приближения для неизвестной части краевых условий на границах каждой подобласти и на стыке подобластей;

3. Происходит обмен данными – в каждый из функционалов ошибки вводится информация о рассогласовании решений

$$\sum_{q\neq p}\varepsilon_{q,p}\sum_{j=1}^{M'_{q,p}}\sum_{i=1}^{r_p}\left|u_{i,p}-u_{i,q}\right|^2(\mathbf{x}_{j,q,p}),$$

где пробные точки $\mathbf{x}_{j,q,p}$ берутся в пересечении $\Omega'_q\cap\Omega'_p$ (что даёт более гладкую стыковку), этой информацией является решение, построенное на другой подобласти.

4. Процедура вычислений повторяется заданное число раз или до достижения уровня требуемой точности.

Построение моделей для подобластей и обмен данными при построении цифрового двойника в целом может быть реализован в рамках *grid*-технологий. При этом решение для каждой подобласти .. подбирается на своём компьютере с учётом аппроксимаций решений на пересечении с соседними областями, информация о которых пересылается время от времени с соответствующих компьютеров. Эффективность такой процедуры сильно зависит от того, насколько удачно удалось провести декомпозицию, т.е. насколько сильно на процесс моделирования или адаптации элемента сказывается информация о связанных с ним элементах. Для адаптации критичной также является скорость изменения объекта, для которого строится цифровой двойник.

III. *Подход, использующий идеологию МГУА* [36]

1. Для каждого условия $A_q(u_1,u_2,...,u_r)\big|_{\Omega_q}=0$ выбираем некоторое множество наборов функций $u_1,u_2,...,u_r$.

2. Рассматриваем линейные комбинации таких наборов (только парные или не только, в любом случае объём перебора следует жёстко ограничить) и подбираем их коэффициенты (возможно, не только c_i), минимизируя функционал $\sum_{j=1}^{M_p}\left|A_p(u_1,u_2,...,u_r)(\mathbf{x}_{j,p})\right|_{\Omega_p}\Big|^2$.

3. Выбираем лучшие из получившихся функций в смысле лучшего значения функционалов вида

$$\sum_{q\in Q_p}\delta_{q,p}\sum_{j=1}^{M_p}\left|A_q(u_1,u_2,...,u_r)(\mathbf{x}_{j,q})\right|_{\Omega_q}\Big|^2 ,$$

где Q_p – номера областей пересекающихся (близких) с Ω_p. Для выбора можно использовать и несколько таких функционалов.

4. Рассматриваем линейные комбинации получившихся функций и повторяем предыдущие шаги, подбирая параметры моделей по одним функционалам, а выбор лучших моделей осуществляя по другим функционалам, пока ошибка не становится

достаточно малой.

Заметим, что возможна такая модификация алгоритма: при составлении парных линейных комбинаций в п.2 выбираются функции, отвечающие разным подобластям, а при отборе пар в п.3 лучшими считаются, например, пары, дающие минимальное рассогласование в пересечении подобластей. В любом случае, на первых реализациях п. 2 мы берём парные комбинации наборов, отвечающие одному и тому же элементу объекта, для которого строится цифровой двойник, далее рассматриваются комбинации моделей элементов одного вычислительного узла и уже затем комбинации моделей различных узлов.

IV. *Использующий декомпозицию области генетический алгоритм построения набора нейронных сетей, задающего решение*

1. Для каждого элемента целевого объекта и каждой из подобластей Ω'_p строим популяцию из K аппроксимаций для решения $u_{1,p}, u_{2,p}, \ldots, u_{r_p,p}$, используя при задании соответствующего функционала ошибки J_p соответствующую Ω'_p часть условий

$$J_p = \sum_{q=1}^{Q_p} \delta_{q,p} \sum_{j=1}^{M_p} \left| A_q(u_{1,p}, u_{2,p}, \ldots, u_{r_p,p})(\mathbf{x}_{j,q,p}) \right|_{\Omega_q \cap \Omega'_p} \Bigg|^2,$$

при этом, как и во втором подходе, контрольные множества точек берутся на границе там, где известны краевые условия.

2. Выбираем для дальнейшей работы из каждого коллектива лучшие модели в числе $K_1 < K$, исходя из минимума функционала ошибки по области $\Omega \setminus \Omega'_p$. Другой вариант этого шага – выбирается K_1 моделей, дающих минимальное значение функционалу, который использовался для их обучения, а из них $K_2 < K_1$ моделей, на которых минимален другой функционал аналогичного вида. Ещё один вариант – ранжировать все модели по ошибкам на каждом из подмножеств Ω'_p и для дальнейшей работы выбирать модели, имеющие минимальный суммарный ранг. Если модель попала в число худших с точки зрения своего функционала, но относительно другого функционала попала в число лучших, тогда её можно переместить в соответствующий коллектив.

3. Производим случайные мутации моделей, вероятность которых тем больше, чем больше ошибки по своей и по чужой области (например, сумма ошибок). Эти мутации могут быть разного типа: удаление слагаемого в сумме

$$u_s(\mathbf{x}) = \sum_{i=1}^{N} c_{i,s} \varphi_s(\mathbf{x}; \mathbf{a}_{i,s})$$

для данной модели с минимальным коэффициентом c_i или удаление случайно выбранного слагаемого; добавление функции φ_i со случайным векторным параметром $\mathbf{a}_i$; случайное изменение параметров $\mathbf{a}_i$ на некоторое значение и т.д. При этом функции φ_i могут быть разными, что приводит к построению гетерогенных моделей.

4. Делаем случайные транслокации – например, обменивая коэффициенты c_i у двух поднаборов функций. Этот пункт не обязателен, и его можно опустить.

5. Проводим скрещивание – берем лучшие $K_2 < K_1$ моделей (в смысле минимума соответствующего функционала), выбираем две из них и часть слагаемых берем от одной модели, часть – от другой, в результате получается новая модель, которая пополняет множество подбираемых моделей и называется потомком. Эта операция повторяется с некоторым множеством пар таких моделей. При этом часть потомков производится от моделей одной популяции (моделей, обучавшихся по одному и тому же множеству Ω'_p), часть – от моделей разных популяций. Получившимися потомками дополняем каждую популяцию до прежнего числа K или некоторого другого, если рассматривается популяция переменного размера.

6. Повторяем шаги 2-5 определённое число раз или до тех пор, пока ошибка

$$J(u_1, u_2, \ldots, u_r) = \sum_{q=1}^{Q} \delta_q \sum_{j=1}^{M} \left| A_q(u_1, u_2, \ldots, u_r)(\mathbf{x}_{j,q}) \right|_{\Omega_q} \Bigg|^2$$

для какого-либо набора моделей не станет достаточно малой.

Можно модифицировать данный алгоритм, добавив к нему процедуру кластеризации ошибок (подход I). Получившиеся функции добавляем к основному набору в качестве одного из видов мутаций на шаге 3.

Данный алгоритм легко адаптировать к распределённым вычислениям. Наиболее естественный вариант возникает, если каждый коллектив моделей, соответствующий элементу объекта, для которого строится цифровой двойник, подбирать в своём узле. При этом пересылаться между узлами должны только отдельные модели (набор параметров и информация о структуре) или части моделей, предназначенные для скрещивания. Кроме того, пересылаться может некоторая информация, например, значения оптимизируемых функционалов. Если узлов мало, то в одном узле может строится или адаптироваться несколько популяций моделей, соответствующих различным элементам, между ними разумно реализовать более интенсивное скрещивание, чем между популяциями из разных узлов. Если узлов много, то на каждом из них можно размещать часть популяции или даже отдельные элементы. Это не слишком сильно скажется на скорости вычислений, в особенности, если модели большие, так как наиболее трудоёмкая операция – подбор параметров осуществляется локально. При этом следует учитывать, что этап построения цифрового двойника мы рекомендуем проводить не в режиме реального времени для целого класса объектов, что позволяет рассматривать большие популяции с целью поиска более глубокого оптимума. Этап адаптации цифрового двойника приходится проводить при работающем объекте в режиме реального времени, поэтому на данном этапе приходится рассматривать маленькие популяции и более редко проводить генетические операции – шаги 3-5. Совсем отказаться от структурной адаптации на данном этапе мы не рекомендуем, так как состояние объекта может резко измениться, что потребует существенной перестройки цифрового двойника.

V. *Обучение коллектива моделей-экспертов* [37]

1. Подбираем K наборов функций вида

$$u_s(\mathbf{x}) = \sum_{i=1}^{N_s} c_{i,s} \varphi_s(\mathbf{x}; \mathbf{a}_{i,s}),$$

минимизируя единый функционал

$$J(u_1, u_2, \ldots, u_r) = \sum_{q=1}^{Q} \delta_q \sum_{j=1}^{M_q} \left| A_q(u_1, u_2, \ldots, u_r)(\mathbf{x}_{j,q}) \right|_{\Omega_q} \Bigg|^2.$$

В этом случае можно взять число N_s не слишком большим.

2. Для каждой каждого элемента объекта, для которого строится цифровой двойник, и подобласти Ω_q выбираем набор моделей, для которых минимальна ошибка

$$J_q(u_1, u_2, \ldots, u_r) = \sum_{j=1}^{M_q} \left| A_q(u_1, u_2, \ldots, u_r)(\mathbf{x}_{j,q}) \right|_{\Omega_q} \Bigg|^2,$$

соответствующая данной подобласти.

3. Подбираем каждый набор на своём подмножестве Ω_q,

включая в минимизируемый функционал слагаемые, отвечающие за рассогласование на стыках (см. Подход II).

В результате обучения возникает приближённое решение, которое в каждой подобласти задаётся соответствующей моделью.

Данный алгоритм допускает модификацию, при которой декомпозиция области не задаётся априорно, а производится естественным образом в процессе работы алгоритма.

1. Подбираем K наборов функций вида $u_s(\mathbf{x}) = \sum_{i=1}^{N_s} c_{i,s}\varphi_s(\mathbf{x};\mathbf{a}_{i,s})$, минимизируя функционал

$$J(u_1,u_2,...,u_r) = \sum_{q=1}^{Q}\delta_q \sum_{j=1}^{M_q}\left|A_q(u_1,u_2,...,u_r)(\mathbf{x}_{j,q})\big|_{\Omega_q}\right|^2.$$

В этом случае можно взять число N_s не слишком большим.

2. Для каждого набора вычисляем ошибки

$$z_{j,q} = A_q(u_1,u_2,...,u_r)(\mathbf{x}_j),$$

при этом набор тестовых точек $\mathbf{x}_j$ из $\Omega = \bigcup_{q=1}^{Q}\Omega_q$ можно поменять по сравнению с предыдущим пунктом, главное условие – он должен быть достаточно представительным. Если значение A_q не определено в некоторой точке $\mathbf{x}_j$, то полагаем $z_{j,q} = 0$.

3. Проводим кластеризацию точек

$$\{(\mathbf{x}_j, z_{j,1}, z_{j,2}, ..., z_{j,Q})\}.$$

4. Берём кластеры, (они могут и пересекаться), строим соответствующее покрытие множества Ω и выбираем для каждого получившегося подмножества и каждого элемента моделируемого объекта набор функций, дающий минимальную ошибку.

5. Модифицируем каждый набор на своём подмножестве, включая в минимизируемый функционал слагаемые, отвечающие за рассогласование на стыках (см. Подход II). Сами подмножества могут меняться в процессе работы алгоритма.

Так же, как и предыдущий, Подход V допускает достаточно простую и эффективную распределённую реализацию. Наиболее простой вариант такой реализации состоит в том, чтобы обучать на каждом узле свой набор функций, который оказался лучшим на некотором подмножестве параметров, соответствующих определённому элементу объекта, для которого строится цифровой двойник. При этом пересылать необходимо только информацию о поведении аппроксимации на стыках и только на те узлы, которым соответствуют элементам, стыкующимся с элементами для данного узла. Возможен также вариант, при котором одному узлу соответствует несколько наборов функций. Если узлов много, тогда можно обучать один набор на нескольких узлах, реализуя какой-либо распределённый алгоритм.

Нейросетевые модели вида (2) и (5) имеют существенные недостатки, такие как длительная процедура обучения и однослойный характер нейронной сети. Нами разработан новый подход [38-44], позволяющий быстро сформировать достаточно точное многослойное нейросетевое решение дифференциального уравнения. В отличие от предыдущих подходов, метод опирается не на МКЭ, а на метод конечных разностей. Сильной стороной метода является автоматическое включение в итоговую формулу параметров задачи, что позволяет обойтись без её многократного повторного решения при необходимости исследовать влияние параметров на результат. Это особенно важно для построения цифрового двойника конкретного объекта, учитывающего его уникальные особенности. Поясним сказанное на примере обыкновенных дифференциальных уравнений.

Рассмотрим задачу Коши для системы обыкновенных дифференциальных уравнений

$$\begin{cases}\mathbf{y}'(x) = \mathbf{f}(x,\mathbf{y}(x)),\\ \mathbf{y}(x_0) = \mathbf{y}_0\end{cases} \quad (9)$$

на промежутке $D = [x_0; x_0 + a]$. Здесь $x \in D \subset \mathbb{R}$, $\mathbf{y} \in \mathbb{R}^d$, $\mathbf{f}: \mathbb{R}^{d+1} \to \mathbb{R}^d$.

Классический метод Эйлера состоит в разбиении промежутка D на n частей: $x_0 < x_1 < ... < x_k < x_{k+1} < ... < x_n = x_0 + a$, и применении итерационной формулы

$$\mathbf{y}_{k+1} = \mathbf{y}_k + h_k\mathbf{f}(x_k, \mathbf{y}_k), \quad (10)$$

где $h_k = x_{k+1} - x_k$; $\mathbf{y}_k$ – приближение к точному значению искомого решения $\mathbf{y}(x_k)$.

Мы предлагаем с помощью формулы (10) строить приближённое решение задачи (9) на интервале $\tilde{D} = [x_0, x]$ с переменным верхним пределом $x \in [x_0, x_0 + a]$. При этом $h_k = h_k(x)$, $\mathbf{y}_k = \mathbf{y}_k(x)$, определяется начальными условиями $\mathbf{y}_0(x) = \mathbf{y}_0$. В качестве приближённого решения задачи (9) предлагается использовать $\mathbf{y}_n(x)$.

Отметим, что начальные значения $\mathbf{y}_0$ входят в выражение для $\mathbf{y}_n(x)$ в качестве параметров. Кроме этого, если функция $\mathbf{f}$ зависит от некоторых параметров, эти параметры войдут в выражение для $\mathbf{y}_n(x)$.

Заметим, что если функция $\mathbf{f}$ в (9) является нейросетевой, то формула (10) позволяет получить многослойную аппроксимацию решения, то есть непредставимую в виде (2) со стандартными базисными функциями. Если $\mathbf{f}$ не является нейросетевой, то её можно приблизить нейронной сетью из соответствующего класса [25] и снова получить многослойное нейросетевое решение.

Известно, что метод Эйлера обладает невысокой точностью. Его можно заменить на более точные методы, например, на метод Рунге-Кутта. Обычные оценки точности исходных классических методов позволяют привести удобные оценки точности полученных приближений. Если точность полученного таким образом решения недостаточна, соответствующие нейронные сети можно обучить с помощью обычных методов [26-35], минимизируя соответствующий функционал со слагаемыми (3).

Ряд работ, выполненных в СПбПУ, направлен на создание нейросетевого подхода к созданию моделей-двойников. В работах [40-42] с помощью представленного выше многослойного подхода построены модели реальных объектов, лучше согласованные с наблюдениями, чем точные решения исходных дифференциальных уравнений.

Заключение

В статье нами предложены методы, позволяющие строить цифровые двойники реальных объектов. Мы представляем алгоритмы построения цифровых моделей-двойников, которые могут быть гибко адаптированы под конкретный тип реальных объектов, для которых требуется построить цифрового двойника. Для такой адаптации можно использовать конкретизацию этих алгоритмов, изложенную в соответствующих работах из списка литературы.

Благодарности

Работа выполнена при финансовой поддержке РНФ (код проекта 18-19-00474).

Список использованных источников

[1] *Uhlemann T.H.-J., Steinhilper C.L.R., Steinhilper R.* The Digital Twin: Realizing the Cyber-Physical Production System for Industry 4.0 // Procedia CIRP. 2017. Vol. 61. Part of special issue: The 24th CIRP Conference on Life Cycle Engineering. Ed. by S. Takata, Y. Umeda, S. Kondoh. Pp. 335-340. DOI: 10.1016/j.procir.2016.11.152

[2] *Uhlemann T.H.-J., Schock C., Lehmann C., Freiberger S., Steinhilper R.* The Digital Twin: Demonstrating the Potential of Real Time Data Acquisition in Production Systems // Procedia Manufacturing. 2017. Vol. 9. Pp. 113-120. DOI: 10.1016/j.promfg.2017.04.043

[3] *Boschert S., Rosen R.* Digital Twin – The Simulation Aspect / P. Hehenberger, D. Bradley (Eds.) // Mechatronic Futures. Springer International Publishing, 2016. Pp. 59-74. DOI: 10.1007/978-3-319-32156-1_5

[4] *Dong-Ki K.* et al. A Study of Resource Management for Fault-Tolerant and Energy Efficient Cloud Datacenter / Y. Zhang, L. Peng, C-H. Youn (Eds.) // Cloud Computing. 6th International Conference, CloudComp 2015, Daejeon, South Korea, October 28-29, 2015, Revised Selected Papers. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Vol. 167. Springer International Publishing, 2016. Pp. 22-29. DOI: 10.1007/978-3-319-38904-2_3

[5] *Chawla V., Sogani P.* Cloud Computing – The Future / A. Mantri, S. Nandi, G. Kumar, S. Kumar (Eds.) // High Performance Architecture and Grid Computing. HPAGC 2011. Communications in Computer and Information Science, Vol. 169. Springer, Berlin, Heidelberg, 2011. Pp. 113-118. DOI: 10.1007/978-3-642-22577-2_15

[6] *Strang G., Fix G.J.* An Analysis of the Finite Element Method. Vol. 212. Prentice-Hall, Englewood Cliffs, NJ, 1973. 306 p.

[7] *Hughes T.J.* The Finite Element Method: Linear Static and Dynamic Finite Element Analysis. Courier Corporation, North Chelmsford, 2012. 704 p.

[8] *Dhatt G., Lefrançois E., Touzot G.* Finite Element Method. London, Wiley-ISTE, 2012. 624 p.

[9] *Bathe K.J.* Finite Element Method // Wiley Encyclopedia of Computer Science and Engineering. London, Wiley Online Library, 2008. DOI: 10.1002/9780470050118.ecse159

[10] *Nguyen V.P., Rabczuk T., Bordas S., Duflot M.* Meshless methods: a review and computer implementation aspects // Mathematics and Computers in Simulation. 2008. Vol. 79, issue 3. Pp. 763-813. DOI: 10.1016/j.matcom.2008.01.003

[11] *Nguyen V.P., Anitescu C., Bordas S.P., Rabczuk T.* Isogeometric analysis: an overview and computer implementation aspect // Mathematics and Computers in Simulation. 2015. Vol. 117. Pp. 89-116. DOI: 10.1016/j.matcom.2015.05.008

[12] *Hughes T.J., Cottrell J.A., Bazilevs Y.* Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement // Computer Methods in Applied Mechanics and Engineering. 2005. Vol. 194, issue 39-41. Pp. 4135-4195. DOI: 10.1016/j.cma.2004.10.008

[13] *Marussig B., Zechner J., Beer G., Fries T.P.* Fast isogeometric boundary element method based on independent field approximation // Computer Methods in Applied Mechanics and Engineering. 2015. Vol. 284. Pp. 458-488. DOI: 10.1016/j.cma.2014.09.035

[14] *Atroshchenko E., Xu G., Tomar S., Bordas S.* Weakening the tight coupling between geometry and simulation in isogeometric analysis: from sub-and super-geometric analysis to Geometry Independent Field approximaTion (GIFT). 2017. arXiv:1706.06371

[15] *Saputra A., Talebi H., Tran D., Birk C., Song C.* Automatic image-based stress analysis by the scaled boundary finite element method // International Journal for Numerical Methods in Engineering. 2017. Vol. 109, issue 5. Pp. 697-738. DOI: 10.1002/nme.5304

[16] *Ooi E., Song C., Natarajan S.* A scaled boundary finite element formulation with bubble functions for elasto-static analyses of functionally graded materials // Computational Mechanics. 2017. Vol. 60, issue 6. Pp. 943–967. DOI: 10.1007/s00466-017-1443-y

[17] *Natarajan S., Ooi E.T., Saputra A., Song C.* A scaled boundary finite element formulation over arbitrary faceted star convex polyhedral // Engineering Analysis with Boundary Elements. 2017. Vol. 80. Pp. 218-229. DOI: 10.1016/j.enganabound.2017.03.007

[18] *Simpson R., Trevelyan J.* A partition of unity enriched dual boundary element method for accurate computations in fracture mechanics // Computer Methods in Applied Mechanics and Engineering. 2011. Vol. 200, issue 1-4. Pp. 1-10. DOI: 10.1016/j.cma.2010.06.015

[19] *Peng X., Atroshchenko E., Kerfriden P., Bordas S.* Isogeometric boundary element methods for three dimensional static fracture and fatigue crack growth // Computer Methods in Applied Mechanics and Engineering. 2017. Vol. 316. Pp. 151-185. DOI: 10.1016/j.cma.2016.05.038

[20] *Scott M.A., Simpson R.N., Evans J.A., Lipton S., Bordas S.P., Hughes T.J., Sederberg T.W.* Isogeometric boundary element analysis using unstructured T-splines // Computer Methods in Applied Mechanics and Engineering. 2013. Vol. 254. Pp. 197-221. DOI: 10.1016/j.cma.2012.11.001

[21] *Oden J.T., Lima E.A., Almeida R.C., Feng Y., Rylander M.N., Fuentes D., Faghihi D., Rahman M.M., DeWitt M., Gadde M., Cliff Z.J.* Toward predictive multiscale modeling of vascular tumor growth // Archives of Computational Methods in Engineering. 2016. Vol. 23, issue 4. Pp. 735-779. DOI: 10.1007/s11831-015-9156-x

[22] *Fuentes D., Oden J., Diller K., Hazle J., Elliott A., Shetty A., Stafford R.* Computational modeling and real-time control of patient-specific laser treatment of cancer // Annals of Biomedical Engineering. 2009. Vol. 37, issue 4. Pp. 763-782. DOI: 10.1007/s10439-008-9631-8

[23] *Lê M., Delingett, H., Kalpathy-Cramer J., Gerstner E.R., Batchelor T., Unkelbach J., Ayache N.* Bayesian Personalization of Brain Tumor Growth Model / A.F. Frangi, J. Hornegger, N. Navab, W.M. Wells (Eds.) // MICCAI - Medical Image Computing and Computer Assisted Intervention. 2015, Oct 2015, Munich, Germany. Lecture Notes in Computer Science - LNCS. Vol. 9350. Springer, New York, 2015. Pp. 424–432. DOI: 10.1007/978-3-319-24571-3_51

[24] *Le Folgoc L., Delingette H., Criminisi A., Ayache N.* Sparse Bayesian registration / P. Golland, N. Hata, C. Barillot, J. Hornegger, R. Howe (Eds.) // MICCAI - Medical Image Computing and Computer Assisted Intervention. Proceedings of the 7th International Conference, Boston, MA, USA, September 14-18, 2014. Part I. Lecture Notes in Computer Science - LNCS. Vol. 8673. Springer, Cham, 2014. Pp. 235-242. DOI: 10.1007/978-3-319-10404-1_30

[25] *Haykin S.* Neural Networks: A Comprehensive Foundation. Prentice Hall, 1999. 823 p.
[26] *Vasilyev A., Tarkhov D.* Mathematical Models of Complex Systems on the Basis of Artificial Neural Networks // Nonlinear Phenomena in Complex Systems, 2014. Vol. 17, issue 3. Pp. 327-335.
[27] *Tarkhov D.A., Vasilyev A.N.* New neural network technique to the numerical solution of mathematical physics problems. I: Simple problems // Optical Memory and Neural Networks (Information Optics). 2005. Vol. 14, no. 1. Pp. 59-72.
[28] *Tarkhov D.A., Vasilyev A.N.* New neural network technique to the numerical solution of mathematical physics problems. II: Complicated and nonstandard problems // Optical Memory and Neural Networks (Information Optics). 2005. Vol. 14, no. 2. Pp. 97-122.
[29] *Gorbachenko V.I., Lazovskaya T.V., Tarkhov D.A., Vasilyev A.N., Zhukov M.V.* Neural Network Technique in Some Inverse Problems of Mathematical Physics / L. Cheng, Q. Liu, A. Ronzhin (Eds.) // Advances in Neural Networks – ISNN 2016. ISNN 2016. Lecture Notes in Computer Science - LNCS. Vol. 9719. Springer, Cham, 2016. Pp. 310-316. DOI: 10.1007/978-3-319-40663-3_36
[30] *Shemyakina T.A., Tarkhov D.A., Vasilyev A.N.* Neural Network Technique for Processes Modeling in Porous Catalyst and Chemical Reactor / L. Cheng, Q. Liu, A. Ronzhin (Eds.) // Advances in Neural Networks – ISNN 2016. ISNN 2016. Lecture Notes in Computer Science - LNCS. Vol. 9719. Springer, Cham, 2016. Pp. 547-554. DOI: 10.1007/978-3-319-40663-3_63
[31] *Kainov N.U., Tarkhov D.A., Shemyakina T.A.* Application of Neural Network Modeling to Identification and Prediction Problems in Ecology Data Analysis for Metallurgy and Welding Industry // Nonlinear Phenomena in Complex Systems. 2014. Vol. 17, no 1. Pp. 57–63.
[32] *Kaverzneva T., Lazovskaya T., Tarkhov D., Vasilyev A.* Neural network modeling of air pollution in tunnels according to indirect measurements // Journal of Physics: Conference Series. 2016. Vol. 772, no. 1. P. 012035. DOI: 10.1088/1742-6596/772/1/012035
[33] *Lazovskaya T.N., Tarkhov D.A., Vasilyev A.N.* Parametric Neural Network Modeling in Engineering // Recent Patents on Engineering. 2017, Vol. 11, no. 1. Pp. 10-15. DOI: 10.2174/1872212111666161207155157
[34] *Budkina E.M., Kuznetsov E.B., Lazovskaya T.V., Tarkhov D.A., Shemyakina T.A., Vasilyev A.N.* Neural network approach to intricate problems solving for ordinary differential equations // Optical Memory and Neural Networks. 2017. Vol. 26, issue 2. Pp. 96-109. DOI: 10.3103/S1060992X17020011
[35] *Antonov V., Tarkhov D., Vasilyev A.* Unified approach to constructing the neural network models of real objects. Part 1 // Mathematical Methods in the Applied Sciences. 2018. Vol. 41, issue 18. Pp. 9244-9251. DOI: 10.1002/mma.5205
[36] *Ивахненко А.Г., Юрачковский Ю.П.* Моделирование сложных систем по экспериментальным данным. М.: Радио и связь, 1987. 120 с.
[37] *Растригин Л.А., Эренштейн Р.Х.* Метод коллективного распознавания. М.: Энергоиздат, 1981. 80 с.
[38] *Lazovskaya T., Tarkhov D.* Multilayer neural network models, based on grid methods // IOP Conference Series: Materials Science and Engineering. 2016. Vol. 158, no. 1. P. 012061. DOI: 10.1088/1757-899X/158/1/012061
[39] *Lazovskaya T., Tarkhov D., Vasilyev A.* Multi-Layer Solution of Heat Equation / B. Kryzhanovsky, W. Dunin-Barkowski, V. Redko (Eds.) // Advances in Neural Computation, Machine Learning, and Cognitive Research. Studies in Computational Intelligence. Vol. 736. Springer International Publishing, 2018. Pp. 17–22. DOI: 10.1007/978-3-319-66604-4_3
[40] *Vasilyev A.N., Tarkhov D.A., Tereshin V.A., Berminova M.S., Galyautdinova A.R.* Semi-empirical Neural Network Model of Real Thread Sagging / B. Kryzhanovsky, W. Dunin-Barkowski, V. Redko (Eds.) // Advances in Neural Computation, Machine Learning, and Cognitive Research. Studies in Computational Intelligence. Vol. 736. Springer International Publishing, 2018. Pp. 138–146. DOI: 10.1007/978-3-319-66604-4_21
[41] *Zulkarnay I.U., Kaverzneva T.T., Tarkhov D.A., Tereshin V.A., Vinokhodov T.V., Kapitsin D.R.* A Two-layer Semi-Empirical Model of Nonlinear Bending of the Cantilevered Beam // IOP Conference Series: Journal of Physics: Conference Series. 2018. Vol. 1044, conference 1. P. 012005. DOI: 10.1088/1742-6596/1044/1/012005
[42] *Bortkovskaya M.R., Vasilyev P.I., Zulkarnay I.U., Semenova D.A., Tarkhov D.A., Udalov P.P., Shishkina I.A.* Modeling of the membrane bending with multilayer semi-empirical models based on experimental data / V. Sukhomlin, E. Zubareva, M. Shneps-Shneppe (Eds.) // Proceedings of the 2nd International scientific conference "Convergent cognitive information technologies" (Convergent'2017). Moscow, Russia: November 24–26, 2017. CEUR Workshop Proceedings. Vol. 2064. Pp. 150-156. URL: http://ceur-ws.org/Vol-2064/paper18.pdf (дата обращения: 05.06.2018).
[43] *Васильев А.Н., Тархов Д.А., Шемякина Т.А.* Приближенные аналитические решения обыкновенных дифференциальных уравнений / В.А. Сухомлин, Е.В. Зубарева, М.А. Шнепс-Шнеппе // Избранные научные труды XI Международной научно-практической конференции «Современные информационные технологии и ИТ-образование» (SITITO 2016). Москва, Россия, 25-26 ноября 2016. CEUR Workshop Proceedings. Т. 1761. С. 393-400. URL: http://ceur-ws.org/Vol-1761/paper50.pdf (дата обращения: 05.06.2018).
[44] *Тархов Д.А., Шершнева Е.А.* Приближенные аналитические решения уравнения Матьё, построенные на основе классических численных методов / В.А. Сухомлин, Е.В. Зубарева, М.А. Шнепс-Шнеппе // Избранные научные труды XI Международной научно-практической конференции «Современные информационные технологии и ИТ-образование» (SITITO 2016). Москва, Россия, 25-26 ноября 2016. CEUR Workshop Proceedings. Т. 1761. С. 356-362. URL: http://ceur-ws.org/Vol-1761/paper46.pdf (дата обращения: 05.06.2018).

Поступила 05.06.2018; принята в печать 10.07.2018;
опубликована онлайн 30.09.2018.

REFERENCES

[1] Uhlemann T.H.-J., Steinhilper C.L.R., Steinhilper R. The Digital Twin: Realizing the Cyber-Physical Production System for Industry 4.0. *Procedia CIRP*. 2017. Vol. 61. Part of special issue: The 24th CIRP Conference on Life Cycle Engineering. Ed. by S. Takata, Y. Umeda, S. Kondoh. Pp. 335-340. DOI: 10.1016/j.procir.2016.11.152
[2] Uhlemann T.H.-J., Schock C., Lehmann C., Freiberger S., Steinhilper R. The Digital Twin: Demonstrating the Potential of Real Time Data Acquisition in Production Systems. *Procedia Manu-*

facturing. 2017; 9:113-120. DOI: 10.1016/j.promfg.2017.04.043

[3] Boschert S., Rosen R. Digital Twin – The Simulation Aspect. P. Hehenberger, D. Bradley (Eds.) *Mechatronic Futures*. Springer International Publishing, 2016. Pp. 59-74. DOI: 10.1007/978-3-319-32156-1_5

[4] Dong-Ki K. et al. A Study of Resource Management for Fault-Tolerant and Energy Efficient Cloud Datacenter. Y. Zhang, L. Peng, C-H. Youn (Eds.) *Cloud Computing*. 6th International Conference, CloudComp 2015, Daejeon, South Korea, October 28-29, 2015, Revised Selected Papers. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Vol. 167. Springer International Publishing, 2016. Pp. 22-29. DOI: 10.1007/978-3-319-38904-2_3

[5] Chawla V., Sogani P. Cloud Computing – The Future. A. Mantri, S. Nandi, G. Kumar, S. Kumar (Eds.) *High Performance Architecture and Grid Computing. HPAGC 2011*. Communications in Computer and Information Science, Vol. 169. Springer, Berlin, Heidelberg, 2011. Pp. 113-118. DOI: 10.1007/978-3-642-22577-2_15

[6] Strang G., Fix G.J. An Analysis of the Finite Element Method. Vol. 212. Prentice-Hall, Englewood Cliffs, NJ, 1973. 306 p.

[7] Hughes T.J. The Finite Element Method: Linear Static and Dynamic Finite Element Analysis. Courier Corporation, North Chelmsford, 2012. 704 p.

[8] Dhatt G., Lefrançois E., Touzot G. Finite Element Method. London, Wiley-ISTE, 2012. 624 p.

[9] Bathe K.J. Finite Element Method. *Wiley Encyclopedia of Computer Science and Engineering*. London, Wiley Online Library, 2008. DOI: 10.1002/9780470050118.ecse159

[10] Nguyen V.P., Rabczuk T., Bordas S., Duflot M. Meshless methods: a review and computer implementation aspects. *Mathematics and Computers in Simulation*. 2008; 79(3):763-813. DOI: 10.1016/j.matcom.2008.01.003

[11] Nguyen V.P., Anitescu C., Bordas S.P., Rabczuk T. Isogeometric analysis: an overview and computer implementation aspect. *Mathematics and Computers in Simulation*. 2015; 117:89-116. DOI: 10.1016/j.matcom.2015.05.008

[12] Hughes T.J., Cottrell J.A., Bazilevs Y. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*. 2005; 194(39-41):4135-4195. DOI: 10.1016/j.cma.2004.10.008

[13] Marussig B., Zechner J., Beer G., Fries T.P. Fast isogeometric boundary element method based on independent field approximation. *Computer Methods in Applied Mechanics and Engineering*. 2015; 284:458-488. DOI: 10.1016/j.cma.2014.09.035

[14] Atroshchenko E., Xu G., Tomar S., Bordas S. Weakening the tight coupling between geometry and simulation in isogeometric analysis: from sub-and super-geometric analysis to Geometry Independent Field approximaTion (GIFT). 2017. arXiv:1706.06371

[15] Saputra A., Talebi H., Tran D., Birk C., Song C. Automatic image-based stress analysis by the scaled boundary finite element method. *International Journal for Numerical Methods in Engineering*. 2017; 109(5):697-738. DOI: 10.1002/nme.5304

[16] Ooi E., Song C., Natarajan S. A scaled boundary finite element formulation with bubble functions for elasto-static analyses of functionally graded materials. *Computational Mechanics*. 2017; 60(6):943–967. DOI: 10.1007/s00466-017-1443-y

[17] Natarajan S., Ooi E.T., Saputra A., Song C. A scaled boundary finite element formulation over arbitrary faceted star convex polyhedral. *Engineering Analysis with Boundary Elements*. 2017; 80:218-229. DOI: 10.1016/j.enganabound.2017.03.007

[18] Simpson R., Trevelyan J. A partition of unity enriched dual boundary element method for accurate computations in fracture mechanics. *Computer Methods in Applied Mechanics and Engineering*. 2011; 200(1-4):1-10. DOI: 10.1016/j.cma.2010.06.015

[19] Peng X., Atroshchenko E., Kerfriden P., Bordas S. Isogeometric boundary element methods for three dimensional static fracture and fatigue crack growth. *Computer Methods in Applied Mechanics and Engineering*. 2017; 316:151-185. DOI: 10.1016/j.cma.2016.05.038

[20] Scott M.A., Simpson R.N., Evans J.A., Lipton S., Bordas S.P., Hughes T.J., Sederberg T.W. Isogeometric boundary element analysis using unstructured T-splines. *Computer Methods in Applied Mechanics and Engineering*. 2013; 254:197-221. DOI: 10.1016/j.cma.2012.11.001

[21] Oden J.T., Lima E.A., Almeida R.C., Feng Y., Rylander M.N., Fuentes D., Faghihi D., Rahman M.M., DeWitt M., Gadde M., Cliff Z.J. Toward predictive multiscale modeling of vascular tumor growth. *Archives of Computational Methods in Engineering*. 2016; 23(4):735-779. DOI: 10.1007/s11831-015-9156-x

[22] Fuentes D., Oden J., Diller K., Hazle J., Elliott A., Shetty A., Stafford R. Computational modeling and real-time control of patient-specific laser treatment of cancer. *Annals of Biomedical Engineering*. 2009; 37(4):763-782. DOI: 10.1007/s10439-008-9631-8

[23] Lê M., Delingett, H., Kalpathy-Cramer J., Gerstner E.R., Batchelor T., Unkelbach J., Ayache N. Bayesian Personalization of Brain Tumor Growth Model. A.F. Frangi, J. Hornegger, N. Navab, W.M. Wells (Eds.) *MICCAI - Medical Image Computing and Computer Assisted Intervention*. 2015, Oct 2015, Munich, Germany. Lecture Notes in Computer Science - LNCS. Vol. 9350. Springer, New York, 2015. Pp. 424–432. DOI: 10.1007/978-3-319-24571-3_51

[24] Le Folgoc L., Delingette H., Criminisi A., Ayache N. Sparse Bayesian registration. P. Golland, N. Hata, C. Barillot, J. Hornegger, R. Howe (Eds.) *MICCAI - Medical Image Computing and Computer Assisted Intervention*. Proceedings of the 7th International Conference, Boston, MA, USA, September 14-18, 2014. Part I. Lecture Notes in Computer Science - LNCS. Vol. 8673. Springer, Cham, 2014. Pp. 235-242. DOI: 10.1007/978-3-319-10404-1_30

[25] Haykin S. Neural Networks: A Comprehensive Foundation. Prentice Hall, 1999. 823 p.

[26] Vasilyev A., Tarkhov D. Mathematical Models of Complex Systems on the Basis of Artificial Neural Networks. *Nonlinear Phenomena in Complex Systems*, 2014; 17(3):327-335.

[27] Tarkhov D.A., Vasilyev A.N. New neural network technique to the numerical solution of mathematical physics problems. I: Simple problems. *Optical Memory and Neural Networks (Information Optics)*. 2005; 14(1):59-72.

[28] Tarkhov D.A., Vasilyev A.N. New neural network technique to the numerical solution of mathematical physics problems. II: Complicated and nonstandard problems. *Optical Memory and Neural Networks (Information Optics)*. 2005; 14(2):97-122.

[29] Gorbachenko V.I., Lazovskaya T.V., Tarkhov D.A., Vasilyev A.N., Zhukov M.V. Neural Network Technique in Some Inverse Problems of Mathematical Physics. L. Cheng, Q. Liu, A. Ronzhin

(Eds.) *Advances in Neural Networks – ISNN 2016*. ISNN 2016. Lecture Notes in Computer Science - LNCS. Vol. 9719. Springer, Cham, 2016. Pp. 310-316. DOI: 10.1007/978-3-319-40663-3_36

[30] Shemyakina T.A., Tarkhov D.A., Vasilyev A.N. Neural Network Technique for Processes Modeling in Porous Catalyst and Chemical Reactor. L. Cheng, Q. Liu, A. Ronzhin (Eds.). *Advances in Neural Networks – ISNN 2016.* ISNN 2016. Lecture Notes in Computer Science - LNCS. Vol. 9719. Springer, Cham, 2016. Pp. 547-554. DOI: 10.1007/978-3-319-40663-3_63

[31] Kainov N.U., Tarkhov D.A., Shemyakina T.A. Application of Neural Network Modeling to Identification and Prediction Problems in Ecology Data Analysis for Metallurgy and Welding Industry. *Nonlinear Phenomena in Complex Systems*. 2014; 17(1):57–63.

[32] Kaverzneva T., Lazovskaya T., Tarkhov D., Vasilyev A. Neural network modeling of air pollution in tunnels according to indirect measurements. *Journal of Physics: Conference Series*. 2016; 772(1):012035. DOI: 10.1088/1742-6596/772/1/012035

[33] Lazovskaya T.N., Tarkhov D.A., Vasilyev A.N. Parametric Neural Network Modeling in Engineering. *Recent Patents on Engineering*. 2017; 11(1):10-15. DOI: 10.2174/1872212111666161207155157

[34] Budkina E.M., Kuznetsov E.B., Lazovskaya T.V., Tarkhov D.A., Shemyakina T.A., Vasilyev A.N. Neural network approach to intricate problems solving for ordinary differential equation. *Optical Memory and Neural Networks*. 2017; 26(2):96-109. DOI: 10.3103/S1060992X17020011

[35] Antonov V., Tarkhov D., Vasilyev A. Unified approach to constructing the neural network models of real objects. Part 1. *Mathematical Methods in the Applied Sciences*. 2018; 41(18):9244-9251. DOI: 10.1002/mma.5205

[36] Ivakhnenko A.G., Yurachkovsky Yu.P. Simulation of complex systems from experimental data [Modelirovanie slozhnyh sistem po ehksperimental'nym dannym]. M.: Radio and Communication, 1987. 120 p. (In Russian)

[37] Rastrigin L.A., Erenshtein R.H. Method of collective recognition [Metod kollektivnogo raspoznavanija]. M.: Energoizdat, 1981. 80 p. (In Russian)

[38] Lazovskaya T., Tarkhov D. Multilayer neural network models, based on grid methods. *IOP Conference Series: Materials Science and Engineering*. 2016; 158(1):012061. DOI: 10.1088/1757-899X/158/1/012061

[39] Lazovskaya T., Tarkhov D., Vasilyev A. Multi-Layer Solution of Heat Equation. B. Kryzhanovsky, W. Dunin-Barkowski, V. Redko (Eds.) *Advances in Neural Computation, Machine Learning, and Cognitive Research.* Studies in Computational Intelligence. Vol. 736. Springer International Publishing, 2018. Pp. 17–22. DOI: 10.1007/978-3-319-66604-4_3

[40] Vasilyev A.N., Tarkhov D.A., Tereshin V.A., Berminova M.S., Galyautdinova A.R. Semi-empirical Neural Network Model of Real Thread Sagging. B. Kryzhanovsky, W. Dunin-Barkowski, V. Redko (Eds.) *Advances in Neural Computation, Machine Learning, and Cognitive Research*. Studies in Computational Intelligence. Vol. 736. Springer International Publishing, 2018. Pp. 138–146. DOI: 10.1007/978-3-319-66604-4_21

[41] Zulkarnay I.U., Kaverzneva T.T., Tarkhov D.A., Tereshin V.A., Vinokhodov T.V., Kapitsin D.R. A Two-layer Semi-Empirical Model of Nonlinear Bending of the Cantilevered Beam. *IOP Conference Series: Journal of Physics: Conference Series*. 2018; 1044(conf. 1):012005. DOI: 10.1088/1742-6596/1044/1/012005

[42] Bortkovskaya M.R., Vasilyev P.I., Zulkarnay I.U., Semenova D.A., Tarkhov D.A., Udalov P.P., Shishkina I.A. Modeling of the membrane bending with multilayer semi-empirical models based on experimental data. V. Sukhomlin, E. Zubareva, M. Shneps-Shneppe (Eds.) *Proceedings of the 2nd International scientific conference "Convergent cognitive information technologies" (Convergent'2017)*. Moscow, Russia: November 24–26, 2017. CEUR Workshop Proceedings. Vol. 2064. Pp. 150-156. Available at: http://ceur-ws.org/Vol-2064/paper18.pdf (accessed 05.06.2018).

[43] Vasilyev A., Tarkhov D., Shemyakina T. Approximate analytical solutions of ordinary differential equations. V. Sukhomlin, E. Zubareva, M. Shneps-Shneppe (Eds.) *Proceedings of the XI International Scientific-Practical Conference "Modern Information Technologies and IT-Education" (SITITO 2016).* Moscow, Russia, November 25-26. 2016. CEUR Workshop Proceedings. Vol. 1761. Pp. 393-400. Available at: http://ceur-ws.org/Vol-1761/paper50.pdf (accessed 05.06.2018). (In Russian)

[44] Tarkhov D., Shershneva E. Approximate analytical solutions of Mathieu's equations based on classical numerical methods. V. Sukhomlin, E. Zubareva, M. Shneps-Shneppe (Eds.) *Proceedings of the XI International Scientific-Practical Conference "Modern Information Technologies and IT-Education" (SITITO 2016).* Moscow, Russia, November 25-26. 2016. CEUR Workshop Proceedings. Vol. 1761. Pp. 356-362. Available at: http://ceur-ws.org/Vol-1761/paper46.pdf (accessed 05.06.2018). (In Russian)

Submitted 05.06.2018; revised 10.07.2018; published online 30.09.2018.

About the authors:

Alexander N. Vasilyev, D. Sc. (Engineering), professor, Department of Higher Mathematics, Institute of Applied Mathematics and Mechanics, Peter the Great St. Petersburg Polytechnic University (29 Polytechnic Str., 195251 St. Petersburg, Russia), ORCID: http://orcid.org/0000-0003-0227-0162, a.n.vasilyev@gmail.com

Dmitry A. Tarkhov, D. Sc. (Engineering), professor, Department of Higher Mathematics, Institute of Applied Mathematics and Mechanics, Peter the Great St. Petersburg Polytechnic University (29 Polytechnic Str., 195251 St. Petersburg, Russia), ORCID: http://orcid.org/0000-0002-9431-8241, dtarkhov@gmail.com

Galina F. Malykhina, D. Sc. (Engineering), professor, Department of Measurement and Information Technologies, Institute of Computer Science and Technology, Peter the Great St. Petersburg Polytechnic University (29 Polytechnic Str., 195251 St. Petersburg, Russia), ORCID: http://orcid.org/0000-0002-1026-8727, gfmalykhina@gmail.com