# Общее описание алгоритма моделирования электромеханических переходных процессов

## Аннотация

В документе приведено описание подхода к моделированию ЭМПП с применением неявных методов интегрирования. Описание выполнено по результатам разработки прототипа – программы, реализующей предложенный подход. Прототип предназначен для исследовательских целей, но расчетное ядро разработано с учетом требований к промышленному ПО и может быть использовано для реализации следующего поколения ПК.

Моделирование электромеханических переходных процессов (ЭМПП) использует математическую модель в виде системы дифференциально-алгебраических уравнений (СДАУ). Расчет ЭМПП предполагает реализацию алгоритма, включающего три основных компонента: метод интегрирования, схема интегрирования и процесс интегрирования. Метод интегрирования определяет математические свойства алгоритма в решении СДАУ. Схема интегрирования определяет вычислительный процесс метода интегрирования, который должен быть организован оптимально для учета специфики задачи моделирования ЭМПП. И наконец, процесс интегрирования должен организовывать работу схемы интегрирования в условиях, когда необходим учет дискретных изменений математической модели.

## Метод интегрирования

Для моделирования ЭМПП в том или ином виде требуется решение СДАУ вида

|  |  |
| --- | --- |
|  | (1) |

где:

– вектор дифференциальных переменных;

– вектор алгебраических переменных;

и – гладкие векторные функции;

– время;

– заданные значения переменных при (начальные условия)

Решение системы (1) формально состоит в нахождении функций и , определенных на интервале . В качестве может использоваться константа, задаваемая в исходных данных, но этот параметр также может определяться автоматически в зависимости от поставленной цели моделирования ЭМПП. Часто (1) приводят в компактном виде, опуская обозначение функциональных зависимостей:

|  |  |
| --- | --- |
|  | (2) |

Разделение переменных на дифференциальные и алгебраические обусловлено тем, что для дифференциальных переменных заданы аналитические выражения производных по времени . Для алгебраических переменных такие выражения неизвестны, однако значения переменных могут быть определены численно. Алгебраическая часть (2) может рассматриваться как функциональное ограничение при решении дифференциальной части, но возможны схемы решения (2) без разделения переменных по типам. Так как моделирование ЭМПП выполняется в пространстве состояний, для ряда операций удобно объединять дифференциальные и алгебраические переменные в общий вектор переменных состояния . Данный вектор однозначно определяет состояние моделируемой системы в любой момент времени, и кроме того, при некотором может рассматриваться как вектор начальных условий для интегрирования (2) на интервале .

За редким исключением аналитическое решение (2) невозможно. Вместо аналитического решения используется численное, с нормированной точностью . Численное решение предусматривает замену дифференциальной части (2) конечно-разностными уравнениями и их пошаговое решение с помощью некоторого метода численного интегрирования. Метод интегрирования строит последовательность аппроксимаций удовлетворяющих условию:

|  |  |
| --- | --- |
|  | (3) |

в дискретных точках времени , , Интервал между соседними точками времени называется шагом интегрирования . Для принимается . Таким образом решаемая СДАУ преобразуется к виду:

|  |  |
| --- | --- |
|  | (4) |

Система (4) сохраняет свойства (1) при выполнении условия (3). Одним из важнейших свойств системы дифференциальных уравнений, а значит и СДАУ, является так называемая жесткость. Это термин имеет несколько вариантов математического определения, но качественно сильная жесткость означает наличие в СДАУ компонент со скоростями изменения различающимся на несколько порядков. В задаче моделирования ЭМПП примером компонент с высокой скоростью изменения могут служить дифференциальные контуры моделей АРВ, с низкой скоростью – контуры моделей тепломеханического оборудования. Степень жесткости СДАУ диктует выбор метода интегрирования. Шаг метода интегрирования должен выбираться таким образом, чтобы обеспечивалось выполнение (3). При этом метод интегрирования можно рассматривать как некоторую векторную функцию , с помощью которой по известному можно определить . Безусловно, для алгебраических переменных должна быть выполнена аналогичная операция , но она может быть реализована различными способами по отношению к , и пока не будет приниматься во внимание. Функция метода интегрирования может быть задана в явном виде

|  |  |
| --- | --- |
|  | (5) |

или неявном:

|  |  |
| --- | --- |
|  | (6) |

Очевидно, что (5) выглядит предпочтительно по отношению к (6), так как позволяет вычислить напрямую, без решения системы. Однако рассмотрим пример интегрирования с помощью метода Эйлера в явной

|  |  |
| --- | --- |
|  | (7) |

и неявной

|  |  |
| --- | --- |
|  | (8) |

формах. Для определения по выражению (8) потребуется решение уравнения относительно . Предположим, что так называемое модельное уравнение, используемое для анализа свойств устойчивости методов интегрирования. В соответствии с (7) и (8) можно записать следующие схемы решения для явного метода:

|  |  |
| --- | --- |
|  | (9) |

И для неявного

|  |  |
| --- | --- |
|  | (10) |

Пусть начальные условия , тогда для явного метода

|  |  |
| --- | --- |
|  | (11) |

и для неявного

|  |  |
| --- | --- |
|  | (12) |

Решение модельного уравнения известно: . При , . Это означает, что для явного метода должно выполняться условие:

|  |  |
| --- | --- |
|  | (13) |

и для неявного метода:

|  |  |
| --- | --- |
|  | (14) |

из чего следует, что для обеспечения численно устойчивого решения модельного уравнения шаг явного метода интегрирования должен быть ограничен сверху параметром решаемой задачи, в то время как шаг неявного метода не требует ограничения. При этом ограничение шага явного метода не может быть отброшено до конца интегрирования, так как асимптотически стремится к нулю.

В случае, если требуется решить систему дифференциальных уравнений, и в данной системе хотя бы небольшая часть компонент изменяется существенно быстрее остальных (то есть система является жесткой), явный метод интегрирования должен работать с шагом, который ограничивается параметрами уравнений быстрых компонент. Попытка увеличения шага может привести не только к нарушению условия (3) в локальных , но и к потере устойчивости всего решения. Для неявного метода интегрирования локальная ошибка, как правило, не означает потери устойчивости и возможность получить правдоподобное решение, пусть и с погрешностью, сохраняется. Данным свойством неявных методов можно воспользоваться на практике для приближенного, но ускоренного интегрирования процесса с компонентами, время изменения которых существенно меньше заданного шага интегрирования. Неявный метод не сможет адекватно воспроизвести поведение данных компонент на интервалах быстрых изменений, но после их затухания позволит продолжить интегрирование без риска неконтролируемого роста погрешности и нарушения численной устойчивости. Необходимо отметить, что данное свойство можно использовать только при условии, что решаемая система уравнений неизменна на интервале интегрирования, в противном случае отказ от точного воспроизведения быстрых компонент может привести тому, что будут проигнорированы изменения структуры модели, ограничения, срабатывания устройств автоматики и т.д.

## Схема интегрирования

Для решения исходной СДАУ (2) дифференциальные уравнения должны быть преобразованы в алгебраические с использованием функций, зависящих от метода интегрирования. Применение явного метода интегрирования не требует ввода эквивалентной системы алгебраических уравнений, позволяет решить дифференциальные уравнения (2) отдельно от алгебраических, использовать найденные решения в алгебраической системе и согласовать общее решение СДАУ итерационным путем. Применение неявного метода таким способом также возможно, но так как его использование в любом случае означает ввод новой системы алгебраических уравнений (6), то, по сути, система дифференциальных уравнений в (2) заменяется на алгебраическую систему. При этом исходная СДАУ преобразуется в эквивалентную чисто алгебраическую систему нелинейных уравнений, которую необходимо решать итерационным методом. Таким образом необходимость решения системы нелинейных алгебраических уравнений возникает вне зависимости от используемого метода интегрирования, применяемого к СДАУ. В связи с этим целесообразно использовать неявные методы интегрирования, так как их преимущества очевидны, а трудоемкость реализации, связанная с необходимостью итерационного решения системы уравнений, возникает в любом случае. Для решения результирующей системы алгебраических уравнений могут применяться различные методы. Возможен упрощенный в части реализации подход на базе метода простой итерации. Практика показывает, что данный метод работоспособен, но в определенных ситуациях в принципе не позволяет получить решения из-за присущих ему ограничений, связанных с жесткими требованиями к свойствам системы. Для использования метода Ньютона требуется формирование матрицы Якоби алгебраической системы, что многократно повышает трудоемкость разработки моделей: дополнительно к набору уравнений для каждой модели потребуется формировать блок матрицы Якоби. Однако метод Ньютона позволяет надежно получать решение, обладает хорошей сходимостью и, кроме того, может использоваться не только для решения (2), но и для определения начальных условий. Слабой стороной метода Ньютона является чувствительность к качеству начального приближения. Отчасти эта проблема решается путем использования начального приближения от предыдущего шага интегрирования. В качестве дополнительного фактора, улучшающего условия сходимости метода Ньютона, может выступать схема метода интегрирования. Распространенной схемой интегрирования является схема по методу прогноза-коррекции. Располагая данными о выполненном решении в точке метод с помощью экстраполяции может построить прогноз значений . Полученный прогноз используется в качестве начального приближения для решения эквивалентной (2) системы, в которой дифференциальные уравнения дискретизированы в виде, зависящем от выбранного метода интегрирования. В процессе решения прогноз корректируется так, чтобы значения удовлетворили заданной системе уравнений. Рассматриваемая схема эффективно реализуется при использовании многошаговых методов интегрирования, которые предусматривают сохранение данных выполненных шагов и их использование для выполнения очередного шага. Для решения СДАУ этот подход реализован в методе Гира [1]. Данный метод предполагает использование метода интегрирования BDF как для дифференциальных, так и для алгебраических уравнений, и обеспечивает совместное решение (2). В Eurostag [2] метод модифицирован так, что для дифференциальных переменных используется метод Адамса, а для алгебраических – BDF. Модификация применена ввиду того, что методы семейства BDF обладают свойством гиперустойчивости – демпфирования даже неустойчивых компонент решения при работе с шагом интегрирования, превышающим время изменения данных компонент. Применение гибридного метода для решения (2) позволяет исключить эффект гиперустойчивости. Разработанный прототип наглядно демонстрирует как данный эффект, так и результат его устранения за счет разделения методов интегрирования по типам переменных.

Все многошаговые методы любого порядка , несмотря на существенные различия их свойств, могут быть формализованы в виде, использующем представление с вектором Нордсика. Данный вектор, имеет размерность . Для реализованного в прототипе метода интегрирования , поэтому для вектора Нордиска зарезервирована размерность 1×3.

На шаге интегрирования для дифференциальных и алгебраических переменных векторы Нордсика имеют вид:

|  |  |
| --- | --- |
|  | (15) |

Векторы содержат информацию для представления и в виде ряда Тейлора до второго порядка в точке . Если известны векторы Нордсика для , то прогноз в точке может быть выполнен по выражениям:

|  |  |
| --- | --- |
|  | (16) |

где:

– нижне-треугольная матрица Паскаля;

– экстраполированные в векторы Нордсика по дифференциальным и алгебраическим переменным.

Коррекция прогноза для получения вектора Нордсика в точке выполняется по выражениям:

|  |  |
| --- | --- |
|  | (17) |

где:

– вектор-строка коэффициентов метода интегрирования для дифференциальных переменных;

– вектор-строка коэффициентов метода интегрирования для алгебраических переменных;

– векторы отклонений значений прогноза от решения в точке .

Векторы и нормированы так, что

|  |  |
| --- | --- |
|  | (18) |

Покомпонентно (17) можно записать в виде:

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  | | --- | --- | |  |  | | (19) |

Для дифференциальных переменных (2), используя уравнение для из (19), можно записать:

|  |  |
| --- | --- |
|  | (20) |

С учетом (17) и (18) получаем систему нелинейных алгебраических уравнений

|  |  |
| --- | --- |
|  | (21) |

Для решения которой относительно и необходимо сформировать матрицу Якоби со следующей структурой:

|  |  |
| --- | --- |
|  | (22) |

С учетом (21) можно переписать (22) в виде:

|  |  |
| --- | --- |
|  | (23) |

Следует заметить, что матрица Якоби при не становится вырожденной, что позволяет её использовать для решения СДАУ при поиске промежуточных начальных условий , если (2) потребуется изменить в некоторой точке . Кроме того, уменьшение шага не вызывает численных проблем с округлением, которым подвержена, например, схема интегрирования [3].

Решение (21) выполняется итерационно. Для итерации переменные в точке можно представить в виде:

|  |  |
| --- | --- |
|  | (24) |

Итерационный процесс решения (21) относительно и выполняется по рекуррентному выражению:

|  |  |
| --- | --- |
|  | (25) |

После того как итерационный процесс сойдется, векторы Нордиска и в точке рассчитываются по выражению (17).

Полученные в результате решения (25) по дифференциальным и алгебраическим переменным представляют собой разности между спрогнозированными и скорректированными значениями. По этим разностям можно оценивать погрешность локального решения и рассчитывать допустимый шаг и оптимальный порядок метода. Локальная ошибка на шаге определяется по выражению:

|  |  |
| --- | --- |
|  | (26) |

где:

– константа, зависящая от метода интегрирования;

– порядок метода интегрирования;

– элемент вектора коэффициентов метода интегрирования порядка .

При оценке погрешности принято использовать индивидуальные характеристики для каждой из компонент решения. Для этого вводится взвешивающий вектор:

|  |  |
| --- | --- |
|  | (27) |

где:

– относительная погрешность -ой компоненты вектора состояния;

– абсолютная погрешность -ой компоненты вектора состояния.

Погрешность решения на шаге является допустимой если выполняется:

|  |  |
| --- | --- |
|  | (28) |

Использование -нормы для оценки погрешности в отличие, например, от -нормы, позволяет отказаться от излишнего снижения шага интегрирования для повышения точности расчета быстрых компонент, которые сами по себе не столь важны для анализа ЭМПП. Квадратичная норма дает возможность контроля интегральной погрешности решения всей системы (2).

Данная оценка погрешности дополнительно используется для контроля сходимости (25), а также для выбора шага и порядка метода на следующем шаге интегрирования. Возможность изменения порядка метода используется в случае, если оценка следующего шага методом иного порядка окажется больше, чем оценка следующего шага текущего метода. Операции изменения шага и порядка многошаговых методов требуют модификации векторов Нордиска, в частности масштабирования, увеличения или уменьшения размерности, поэтому частота изменения параметров метода ограничивается. Управление шагом и порядком интегрирования реализовано на базе работы [4], в которой данные вопросы рассматриваются применительно к чисто дифференциальным системам уравнений. Так как решаемая задача является дифференциально-алгебраической и жесткой, то дополнительные меры контроля сходимости корректора и подавления спорадических изменений шага реализованы на базе [5].

На рисунке приведено сравнение шага интегрирования ПК RUSTab и прототипа, использующего описанную выше схему интегрирования. Сравнение выполнено на тестовой схеме из комплекта RUSTab с абсолютной точностью . На рисунке представлен график мощности одного из генераторов, рассчитанный в двух программах. Генераторы в тестовой схеме представлены моделями в форме ЭДС с одним контуром и работают под управлением АРВ сильного действия с активными каналами по напряжению, производной напряжения, частоте, производной частоты и производной тока ротора.

На следующем рисунке изображен график изменения шага интегрирования.

В быстрой фазе процесса 0.5-2.5с оба метода интегрирования работают с близкими значениями шага. Неявный метод не может иметь преимущества при интенсивной динамике по всем компонентам. Кроме того, схема интегрирования RUSTab не контролирует локальную ошибку по алгебраическим переменным напрямую (только решение дифференциальных уравнений), в отличие от схемы прототипа, который одинаково контролирует все переменные состояния.

По мере затухания метод прототипа многократно увеличивает шаг несмотря на то, что быстрые компоненты сохраняют небольшую активность, которая не дает увеличивать шаг методу RUSTab. Явный метод интегрирования не позволяет применить выражение типа (28) для контроля погрешности, и требует использования нормы . В противном случае с высокой вероятностью метод потеряет устойчивость при интегрировании быстрых компонент. Так как затраты на выбор шага метода RUSTab практически отсутствуют, шаг следует за скоростью изменения самой активной компоненты (производная частоты в АРВ), которая не затухает до конца расчета.

Резкое снижение шага прототипа после 7с расчета обусловлено накоплением ошибки, вызванной фиксацией матрицы Якоби, которую потребовалось обновить и сократить шаг для рестарта.

## Стабилизация метода Адамса 2-го порядка

Неявный метод Адамса 2-го порядка, известный также как метод трапеций, подвержен явлению «дребезга» (ringing), возникающего при увеличении шага. Дребезг наблюдается, обычно, при затухании ЭМПП и не приводит к искажению решения, но препятствует росту шага интегрирования несмотря на то, что физический процесс стремится к установившемуся. Методу Адамса 2-го порядка соответствует выражение:

|  |  |
| --- | --- |
|  | (29) |

Решение модельного уравнения методом (29) имеет вид:

|  |  |
| --- | --- |
|  | (30) |

При увеличении шага:

|  |  |
| --- | --- |
|  | (31) |

что и приводит к изменению знака решения на каждом шаге интегрирования. Для подавления этого эффекта необходимо исключить влияние на решение свойства (31). Существует несколько решений этой задачи. Для выбранной схемы интегрирования хорошие результаты дает предложенная в [6] замена , после получения решения по (29) на значение, :

|  |  |
| --- | --- |
|  | (32) |

На шаге выражение (29) будет использовать вместо сохраненного на шаге значения

|  |  |
| --- | --- |
|  | (33) |

Замена изменяет свойства устойчивости метода, поэтому выполняется только при обнаружении дребезга через определенное количество шагов интегрирования и при условии, что шаг интегрирования превышает 0.1с.

Эффект от применения подавления дребезга представлен на рисунке

|  |  |
| --- | --- |
|  |  |

## Процесс интегрирования

Рассмотренные выше методы и схема интегрирования предполагают, что СДАУ постоянна. При расчете ЭМПП это условие невыполнимо, так как значительная часть функций не являются гладкими на всей области определения из-за ограничений или имеют дискретный характер. Кроме того, для моделирования возмущений часть уравнений системы приходится модифицировать. В некоторых случаях меняется и размерность системы. Задачи, в которых система (2) не является постоянной выделены в отдельных класс DSAR – Differential Switched Algebraic and State Reset equations. Аналогом термина для нашей практики, по-видимому, может служить «дискретная СДАУ». Очевидный метод решения таких задач: интегрирование последовательности непрерывных СДАУ. Проблемы, которые возникают при решении таких задач, состоят в организации перехода от одной непрерывной системы к другой в строго определенные моменты времени с расчетом совместных начальных условий интегрирования следующей системы. Последняя проблема не является тривиальной, так как нелинейность и наличие дискретных элементов делают возможным множество решений при переходе от одной системы к другой. Гарантировать единственность решения во всех ситуациях невозможно, но за счет дополнительной информации, доступной в ретроспективе расчета и задаваемой в виде правил в моделях, неоднозначность удается разрешить путем формализованного выбора наиболее правдоподобного сочетания переменных состояния.

Таким образом, полноценный алгоритм расчета ЭМПП должен реализовывать так называемый процесс интегрирования, который является надстройкой над схемой интегрирования и решает следующие задачи:

1. Управление событиями.
2. Обработка разрывов.
3. Определение начальных условий и возобновление интегрирования после разрывов.
4. Поиск моментов времени условных событий.

Событие в процессе интегрирования является командой на исполнение некоторых действий в момент времени . Для исполнения действий, как минимум, требуется останов работы текущей схемы интегрирования. В общем случае исполнение действия может потребовать ввода изменений в (2). После исполнения действий, как правило, необходимо возобновить работу схемы интегрирования с новыми начальными условиями.

События можно разделить на две группы [7] – безусловные события (Time Events) и условные события (State Events). События первой группы вводятся в моменты времени, которые известны до начала расчета ЭМПП. К ним относятся события сценария расчета: ввод возмущений и завершение расчета в заданное время. Условные события возникают при определенном сочетании параметров модели и время их ввода (а часто и саму необходимость ввода) требуется определять в процессе расчета. Примерами условных событий являются срабатывания пороговых элементов: ограничений, компараторов, реле, таймеров и т.п.

Обработка события в процессе интегрирования вне зависимости от принадлежности к группам условных или безусловных выполняется одинаково, при условии, что определен момент времени ввода события . Для безусловных событий он известен заранее, для условных – определяется с помощью метода, описанного ниже. Пусть процесс интегрирования выполняет шаг из в , при этом Для корректной обработки событий процесс интегрирования должен выполнить следующие операции:

1. В точке скорректировать и выполнить шаг в точку . Далее считать, что .
2. Проверить локальную ошибку шага. Если она не допустима – отказаться от обработки события и вернуться в для выбора шага по условию допустимости локальной ошибки.
3. В точке зафиксировать результаты расчета до ввода события. Так как событие вводится «мгновенно», время до ввода события обозначается , время после ввода события – . Строго но для интерпретации результатов и устойчивой работы процесса интегрирования принято , где – минимально допустимый шаг метода интегрирования.
4. Ввести необходимые изменения в и (2) и решить новую систему (21) относительно при . Последнее условие означает, что . В некоторых экзотических моделях должно изменяться в точке разрыва (например, производные в АРВ Мустанг), но эти изменения описываются законами отличными от и . Решение (21) в точке разрыва методом Ньютона может представлять затруднения, связанные с выбором начального приближения, поэтому оно предварительно формируется линейным методом решения, а метод Ньютона работает в режиме контроля сходимости по параметру, в отличие от обычного режима работы в итерационном процессе (25).
5. В случае успешного решения (21) зафиксировать результаты расчета после ввода события в точке .
6. Выполнить рестарт схемы интегрирования с начальных условий и , удовлетворяющих решению. Рестарт предполагает переход к первому порядку метода интегрирования и использование среднего значения шага интегрирования, которое должно быть скорректировано на последующих шагах по критерию допустимости локальной ошибки.

Для определения моментов времени ввода условных событий необходим универсальный механизм контроля соотношений значений переменных состояния и пороговых значений в виде:

|  |  |
| --- | --- |
|  | (34) |

где:

– функция, ограничивающая переменную состояния;

Обычно такой механизм предусматривает ввод в алгебраическую часть (2) так называемых функций пересечения (zero-crossing functions):

|  |  |
| --- | --- |
|  | (35) |

По изменению знака можно определить . Контроль изменения знаков функций пересечения выполняется после завершения шага .

Для функций, знак которых изменился, необходимо с точностью до определить . Так как используемая схема основана на многошаговом методе интегрирования, после выполнения шага в доступен вектор Нордсика порядка до 2 для всех переменных, в том числе и для алгебраических:

|  |  |
| --- | --- |
|  | (36) |

используя который можно записать уравнение относительно :

|  |  |
| --- | --- |
|  | (37) |

и определить с погрешностью, не превышающей допустимую для метода интегрирования. После коррекции шага в точке контроль функций пересечения повторяется до тех пор, пока фиксируются изменения знака хотя бы одной из функций.

Нельзя исключать возможность некорректного определения времени условных событий в случае двойного изменения знака на интервале . Значительно снижает вероятность таких ошибок использование дополнительной функции пересечения , которая изменяет знак в точке экстремума . Для данной функции не требуется определять . Достаточно обнаружить факт изменения знака , и в этом случае выполнить поиск пересечения в интервале шага методом деления пополам.

Явное использование функций пересечения необходимо в программном обеспечении, которое использует внешние солверы для решения СДАУ. В этом случае функции пересечения являются частью интерфейса для представления событий в решаемой системе. В прототипе используется встроенный солвер, и поэтому использовать функции пересечения в явном виде нет необходимости. Так как для переменных (34) имеются индивидуальные векторы Нордсика, операцию (37) можно выполнить с разностью компонентов указанных векторов. За счет этого размерность решаемой системы можно не увеличивать для представления функций пересечения. Кроме того, около 80% ограничений в системе (2) имеют , что дополнительно упрощает определение моментов ввода условных событий.

К отдельному классу условных событий можно отнести события с выдержкой времени. Типичный пример – реле, которое срабатывает при выполнении определенного условия в течение заданного времени . Обработка моделей с выдержкой времени требует контроля двух связанных условных событий – момента пуска и момента возврата, если такой произойдет. При этом в момент пуска должно формироваться безусловное событие в . В случае фиксации возврата данное безусловное событие должно отменяться.

События такого класса могут использоваться для автоматического определения времени расчета ЭМПП – . Во многих сценариях симуляции переходных процессов, в число которых входят и ЭМПП, применяется расчет до «установившегося режима», критерием фиксации которого является затухание переходного процесса и отсутствие запланированных событий. В случае, если расчет ЭМПП выполняется до заданного , по событиям с , можно судить о вероятности дальнейшего развития ЭМПП. Отказ от учета некоторых событий, вязанных с защитой оборудования, может привести к ошибочной оценке устойчивости ЭМПП [8].

Для некоторых видов условных событий недостаточно контроля (34), так как функция ограничения может иметь более сложный вид. Например, для контроля положения точки годографа сопротивления требуется ограничение в виде функции двух переменных. Кроме того, некоторые ограничения задаются в виде кусочных функций, поэтому для контроля требуется вводить условное событие и корректировать (2) только для перехода на новый сегмент ограничения. Для таких сложных условных событий единый механизм пока не разработан и используется поиск путем деления шага пополам.

Процесс интегрирования, описание которого приведено выше, является классическим для решения DSAR. Современный подход к этой проблеме смещается к строгой теории решения систем дифференциальных уравнений с негладкой правой частью [9]. Есть положительный опыт использования данного подхода в задаче моделирования ЭМПП [10]. Инфраструктура прототипа, по оценке, не имеет особенностей, препятствующих применению такого подхода в перспективе.

## Особенности подготовки расчетных моделей

Необходимость решения системы нелинейных уравнений методом Ньютона диктует требование к использованию матрицы Якоби. Для всех уравнений, входящих в (2), необходим расчет правой части (по сути значений функций и для текущих , и ) и формирование блоков матрицы Якоби. Для распространённых солверов систем дифференциальных уравнений часто применяется опция аппроксимации матрицы Якоби с помощью конечных разностей, и для таких задач это даёт приемлемый результат [4]. Для жестких систем дифференциальных уравнений и тем более СДАУ такой приём практически неработоспособен, поэтому при разработке модели требуется поэлементное формирование блока матрицы Якоби. Это требование серьезно усложняет разработку моделей. Так, например, для трёхконтурной модели генератора в форме ЭДС, содержащей 7 алгебраических и 5 дифференциальных уравнений, потребуется 46 выражений для расчета частных производных. В ПК RUSTab, который для решения системы нелинейных уравнений использует метод простой итерации, для реализации этой модели было достаточно 12 выражений для расчета значений функций.

В прототипе создана программная инфраструктура, упрощающая ввод выражений для расчета невязок функций и частных производных, но сами выражения приходится формировать вручную. Неприятной особенностью матрицы Якоби в данной схеме интегрирования является слабая чувствительность к ошибкам в выражениях для частных производных. Именно это обстоятельство и позволяет использовать аппроксимацию матрицы Якоби для решения систем чисто дифференциальных уравнений. Применительно к СДАУ ошибка в одном или даже нескольких выражениях для расчета частных производных позволяет получить совершенно корректный результат решения, но при этом шаг интегрирования будет кратно меньше оптимального, а количество итераций метода Ньютона на каждом шаге будет на несколько единиц больше нормального. Таким образом, обнаружить такого рода ошибки при отладке модели, не располагая эталонной реализацией достаточно сложно.

Заметного упрощения разработки моделей удаётся достичь с помощью использования символьного дифференцирования. Эта технология позволяет автоматически преобразовать заданные алгебраические выражения уравнений модели в набор алгебраических выражений частных производных по собственным и по внешним переменным с корректным отнесением к строкам и столбцам матрицы Якоби. В основе технологии лежит синтаксический разбор алгебраических выражений в структуру абстрактного синтаксического дерева (AST – Abstract Syntax Tree). Данная структура ставит в соответствие операнду и оператору выражения узел графа. Ребра графа описывают связи операндов и операторов. Задав правила определения производных для каждого оператора, можно формировать набор частных производных по заранее известному множеству переменных. Используя дополнительную информацию о свойствах переменных в уравнениях, можно исключить частные производных от переменных, которые в модели являются константами. В дополнение AST даёт возможность оптимизировать исходные алгебраические выражения и результирующие выражения для частных производных по выбранному критерию. В качестве такого критерия выбран минимум операций для вычисления выражения. Для минимизации количества операций используются стандартные математические приемы упрощения выражений, включая подстановки основных алгебраических и тригонометрических тождеств. Кроме того, хорошие результаты даёт ввод промежуточных переменных, которые могут многократно использоваться при расчете результирующих выражений. Для полноценного использования этих возможностей AST должно строиться не для каждого отдельного уравнения модели, а для полной системы уравнений модели. В результате граф имеет вид не дерева, а «леса», с несколькими корневыми узлами.

Полученные в результате символьной обработки системы уравнений выражения могут быть использованы разработчиком для полуавтоматического формирования необходимой модели. Однако в прототипе результаты символьной обработки используются для полностью автоматического формирования исходного текста модели на языке C, который может быть скомпилирован в самостоятельную библиотеку. Такая библиотека может быть включена в общее ядро как встроенная модель, но также может быть вынесена в отдельную dll. В любом случае такая модель будет иметь характеристики скорости расчета идентичные модели корректно реализованной «вручную» разработчиком. Технология компиляции моделей отработана в ПК RUSTab, показала отличные результаты и получила развитие в прототипе. С помощью компиляции могут создаваться пользовательские модели и модели описаний автоматики и сценария. Последние также требуют ввода в (2) соответствующих уравнений. В RUSTab эти описания обрабатываются виртуальной машиной, которая, естественно, уступает в производительности машинному коду в несколько раз.

Кроме выражений для расчета блока частных производных и невязок уравнений модель должна содержать правила обработки разрывов функций и . Правила необходимы для однозначного решения (2) в точках разрыва при наличии в моделях дискретных или нелинейных элементов. Для описания такого рода моделей, представляемых DSAR, созданы специальные языки, например — Modelica и Dymola, в которых часть семантики ориентирована на описание обработки разрывов. Одной из целей, которая ставится при разработке таких языков, является универсальность подхода, так как системы моделирования, в которых они используются, предназначены для решения широкого круга инженерных задач. Поэтому такие языки позволяют в явном виде описывать дифференциальные и алгебраические уравнения. Для расчета ЭМПП, как правило, используются модели, состоящие из связанных стандартных блоков с возможностью параметризации. Использование стандартных блоков позволяет в значительной степени унифицировать обработку разрывов, так как соответствующие правила могут быть включены во встроенные в ядро блоки. Унификация позволяет применить при разработке моделей для расчета ЭМПП графическое представление, такое, например, как в SimuLink или Eurostag. Подавляющее большинство пользовательских моделей могут быть реализованы с помощью графического представления. Однако в случае необходимости задания специфических правил, например — описание логики state machine, потребуется либо ввод нового стандартного блока, либо поиск пути решения проблемы в рамках существующих возможностей [11].

В прототипе реализована гибридная система описания моделей, позволяющая использовать текстовый или графический форматы. Оба формата преобразуются в AST, поэтому предоставляют идентичные возможности при моделировании с помощью стандартных блоков. Текстовый формат имеет дополнительные возможности для описания правил обработки разрывов, которые позволяют создавать модели со сложной логикой поведения в точках разрывов и выходить за рамки доступного множества стандартных блоков.

Преобразование графического представления модели в AST не вызывает проблем, так как обе структуры данных описываются направленными графами. Остальная работа, необходимая для генерации модели, не связана с графическим представлением. Поэтому в качестве графической среды может быть применена универсальная программная подсистема, например – DevExpress Diagrams или даже Microsoft Visio. Для использования в целях моделирования устройств в прототипе в графическую подсистему достаточно ввести набор графических примитивов, представляющих стандартные блоки, правила их соединения и параметризации.

Для корректного ввода встроенной или пользовательской модели в единую расчетную модель ЭМПП необходима информация о привязке, включающая тип модели, набор типов моделей с которыми данная модель может взаимодействовать, а также описание переменных. В RUSTab привязка осуществляется пользователем, путем явного указания типов. В прототипе в каждую модель кроме математического описания включена метаинформация, на основании которой ядро может выполнить привязку автоматически. Понятие «тип модели», которое в RUSTab однозначно определяло тип, в прототипе заменено на множество встроенных типов, в качестве которых модель может быть интерпретирована. Множество строится по принципу наследования функционала моделей. Каждый из типов множества определяет набор переменных, которые необходимы модели для корректной работы и набор собственных переменных, доступ к которым модель может предоставить ядру. Так, например, единственная модель, которая может напрямую взаимодействовать с сетью имеет тип «источник мощности». Для нее обязательны переменные активной и реактивной мощности инъекции. На базе этой модели выполнена модель типа «источник напряжения», имеющая в своем составе описание ЭДС с внутренним сопротивлением. Модели такого типа, в частности, используются для определения состояния зон сети («под напряжением», «отключено», «КЗ», «заземлено»), которые могут образовываться в результате коммутаций. Тип «источник мощности» является базовым для всех моделей генераторов, поэтому уравнения сети строятся вне зависимости от детализации моделей генераторов, так как две переменные необходимые для уравнений сети – активная и реактивная мощности. В то же время информация о детализации необходима, например, для корректного связывания модели генератора и модели системы возбуждения. Простейшая модель генератора, которая может взаимодействовать с системой возбуждения в прототипе – модель типа «Генератор в форме ЭДС с одним контуром». Модель такого типа содержит описание обмотки возбуждения и соответствующие переменные. По мере усложнения моделей генераторов способ взаимодействия с моделью системы возбуждения не изменяется. Таким образом, каждая модель может быть интерпретирована различными способами в зависимости от требуемой детализации. В качестве примера модель генератора, соответствующая «Мустанг» имеет следующее множество типов:

* Источник мощности
* Источник напряжения
* Генератор в форме ЭДС с одним контуром
* Генератор «Мустанг»

Располагая расширенной информацией о типах моделей ядро может определить возможные связи всех заданных моделей и корректно сформировать общую систему уравнений. При разработке пользовательской модели достаточно указать верхний базовый тип, в результате чего множество типов будет построено автоматически по дереву наследования функционала встроенных моделей.

Кроме описания типов в метаинформации моделей также предусмотрено описание переменных. В её состав для каждой переменной входят название в короткой и подробной формах, единицы измерения и тип переменной (собственная, внешняя или константа). Такая метаинформация позволяет корректно связывать модели с неоднозначно определяемыми типами, отображать результаты с автоматическим масштабированием и детальным описанием параметров вне зависимости от того является ли модель встроенной или пользовательской.

# Библиографический список

1. *The Simultaneous Numerical Solution Of Differential - Algebraic Equations SLAC-PUB-0723.* **Gear, C.W.** 1971, IEEE Trans.Circuits Theor. 18, pp. 85-95.

2. *The mixed Adams-BDF variable step size algorithm to simulate transient and long term phenomena in power systems.* **Astic, J.Y., Bihain, A. and Jerosolimski, M.** 2, 1994, IEEE Transactions on Power Systems, Vol. 9.

3. **Petzold, Linda.** *A Description of DASSL: A Differential/Algebraic System Solver.* s.l. : Sandia National Laboratories, 1982. SAND82-8637.

4. **Radhakrishnan, Krishnan and Hindmarsh, Alan C.** *Description and Use of LSODE,the Livermore Solver for Ordinary Differential Equations.* s.l. : NASA reference publication, 1993.

5. **Moore, Peter and Petzold, Linda.** *Stepsize Control strategy For Stiff Systems Of Ordinary Differential Equations.* s.l. : Computer Science Department University of Minnesota, 1994.

6. *A New Stabilization Of Adaptive Step Trapezoid Rule Based On Finite Difference Interrupts.* **Lee, Alex J, Nam, Jaewook and Pasquali, Matteo.** 2, s.l. : SIAM, 2015, Vol. 37.

7. **Cellier, Francois and Kofman, Ernesto.** *Continuous System Simulation.* s.l. : Springer-Verlag, 2006. ISBN:0387261028.

8. **Van Cutsem, Thierry, Glavic, Mevludin and Rosehart, William.** *Test Systems for Voltage Stability Analysis and Security Assessment.* s.l. : IEEE Power & Energy Society, 2015. PES-TR19.

9. **Filippov, A.F.** *Differential Equations with Discontinuous Righthand Sides.* s.l. : Springer Netherlands, 1988. ISBN 978-94-015-7793-9.

10. *Application of Filippov Theory to the IEEE Standard 421.5-2016 Anti-windup PI Controller.* **M. A. Adib Murad, B. Hayes, F. Milano.** Milano : PowerTech, 2019.

11. *Semi-implicit formulation of proportional-integral controller block with non-windup limiter according to IEEE Standard 421.5-2016.* **Fabozzi, D., et al.** Bulk Power Systems Dynamics and Control Symposium (IREP) : s.n., 2017.

12. *Implementation and validation of the Nordic test system in DIgSILENT PowerFactory.* **Ospina, Luis David Pabón, Correa, Andrés Felipe and Lammert, Gustav.** Mahchester : 2017 IEEE Manchester PowerTech, 2017. 978-1-5090-4237-1.