УДК 517.925
DOI: 10.20310/1810-0198-2017-22-6-1268-1276

АЛГОРИТМЫ РЕШЕНИЯ ПРОСТЫХ ТИПОВ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ В MATH PARTNER

© **С. А. Глазков, М. А. Рыбаков**

Тамбовский государственный университет имени Г.Р. Державина
392000, Российская Федерация, г. Тамбов, ул. Интернациональная, 33
E-mail: sergey.glazkov96@yandex.ru, mixail08101987@mail.ru

Мы даем описание алгоритмов символьного решения простых типов обыкновенных дифференциальных уравнений в системе компьютерной алгебры Math Partner. Сюда относятся дифференциальные уравнения с разделяющимися переменными, однородные дифференциальные уравнения и уравнения в полных дифференциалах.
Ключевые слова: дифференциальные уравнения; система компьютерной алгебры; система Math Partner

1. Введение

Системы символьных вычислений с каждым годом находят все больше приложений в науке, технике и образовании. Другое название для этих систем — системы компьютерной алгебры (СКА). Они позволяет решать многие математические задачи, оперировать с функциями и функциональными матрицами, получать как точные численно-аналитические решения, так и решения, в которых числовые коэффициенты получаются с требуемой степенью точности.

В последние годы интерес у прикладных специалистов вызывают облачные системы символьных вычислений. Это свободные системы, которые можно использовать, не устанавливая на свой компьютер, т. к. все вычисления производятся на сервере. Самыми известными из них сегодня являются SAGE и Math Partner.

Мы описываем алгоритмы символьного решения простых типов обыкновенных дифференциальных уравнений в системе компьютерной алгебры Math Partner. Сюда относятся дифференциальные уравнения с разделяющимися переменными, однородные дифференциальные уравнения и уравнения в полных дифференциалах.

В основе языка Mathpar [1-3] лежит широко используемый математиками и физиками язык TeX, который обычно используют для набора математических текстов.

Чтобы пользователь решил обыкновенное дифференциальное уравнение в Math Partner, ему необходимо (1) перейти по адресу http://mathpar.cloud.unihub.ru, (2) открыть электронную тетрадь по ссылке «Тетрадь», (3) задать окружение для рациональных чисел и двух переменных `Q[x,y]` командой `SPACE = Q[x,y]`, (4) ввести оператор для решения уравнения `solveDE` с записанным в нем уравнением и (5) получить решение, нажав на кнопку в виде треугольника над рабочей областью (или с помощью комбинации клавиш Ctrl+Enter).

Пример:

```
SPACE = Q[x,y];
\solveDE(1+y+(1+x)\d(y,x) = 0);
```

В результате получим такой ответ: $abs(y+1) = e^{-(C+ln(|x+1|))}$.

Все разработанные методы собраны в четырех java-классах [4]. Их описание приведено в следующих четырех параграфах. В последних двух параграфах мы приводим примеры решения дифференциальных уравнений и описываем учебную страницу в разделе «Помощь» системы Math Partner.

2. Основной класс решения ОДУ

Прежде чем решать ОДУ, необходимо определить его вид для выбора нужного алгоритма решения. Этим занимается метод `solve` в классе `SolveDiffEq`, который является главным классом. В методе реализованы проверки на тип уравнения и вызов класса решения определенного типа уравнения. Уравнение, заданное пользователем, поступает на вход метода `solve` в качестве двух аргументов — левой и правой частей. Если уравнение было решено, то выдается ответ, иначе — ошибка.

Кроме метода `solve` класс содержит в себе еще 6 методов: `containsDifferential`, `split`, `isETD`, `isHDE`, `HDE_replace`, `toNormalForm`.

Опишем алгоритм работы класса подробнее.

Получаем заданное уравнение в качестве аргументов `solve`. Методом `containsDifferencial` ищем в разности левой и правой частей дифференциалы. Если они присутствуют в уравнении, то оно представимо в виде `Pdx + Qdy = 0`. Тогда с помощью метода `split` получаем из уравнения выражения `P` и `Q`. На данном этапе тип уравнения неизвестен.

Чтобы достоверно определить тип уравнения, вызываем метод `isETD` с параметрами P и Q. В этом методе проверяется равенство выражений P'_y и Q'_x. Если эти выражения совпадают, возвращается $true$ — уравнение в полных дифференциалах. Его решаем в классе `SolveEqInTotalDiff`. Если вернулось $false$ (т. е. выражения не равны), приводим уравнение к явному виду по формуле $Qy' + P = 0$.

Если уравнение не является уравнением в полных дифференциалах или изначально задано в явном виде, решаем его как уравнение с разделяющимися переменными в классе `SolveDiffEqWithSeparableVariables`. Если разделить переменные не удалось, проверяем уравнение на однородность с помощью метода `isHDE`. В нем проводится замена переменных x и y на zx и zy, соответственно (метод `HDE_replace`), после чего упрощаем получившееся выражение. Если получилось исходное уравнение, значит, оно является однородным. Тогда отправляем его на решение в класс `SolveHomogeneousDiffEq`.

Решение уравнений возвращаются в объекте VectorS с одним или двумя полями. Одно поле будет в случае ошибки (нерешаемое системой уравнение), два (левая и правая части решения) — в случае успеха. Если решить уравнение не удалось, программа выдаст строку «$ERROR$: $incorrect\ equation$».

Решение в общем случае получаем в виде неявно заданной функции $g(x,y) = h(x,y)$. Если возможно, оно будет приведено к явному виду $y = f(x)$ в итерационном методе `toNormalForm`. Он последовательно вызывает сам себя до того момента, пока в левой части не окажется y либо неупрощаемое выражение.

Пример:

Вход: $ln(abs(y+1)) = (1/2)x^2 + C$
Выход: $abs(y+1) = e^{(1/2)x^2+C}$

После этого отправляем полученное решение в систему функцией `F.EQUATION`.

3. Решение ДУ с разделяющимися переменными

Алгоритмы решения дифференциальных уравнений с разделяющимися переменными были реализованы в классе `SolveDiffEqWithSeparableVariables`.

В этом классе инициализированы основные глобальные переменные `leftPart` и `rightPart` типа Element, используемые во время подготовки уравнения к решению, и массивы `elemsY` и `elemsX` типа ArrayList, куда будут добавляться функции в процессе разделения переменных.

Методы класса: `solve, solve_left_part, solve_right_part, unconvert, varX, varY, factorRoot, findRoot, factor_exp, group_derivative, find, factor_d, isDerivative, revers`.

3.1. Метод `solve`

Уравнение поступает на вход метода `solve` из главного класса `SolveDiffEq` в виде левой f1 и правой f2 частей.

Подготовка к решению состоит из трех шагов.

1. Вычитаем из f1 выражение f2, после чего f2 приравниваем к нулю. Это требуется для последующей группировки производных, которые изначально могут быть в обеих частях. Далее, если f1 является функцией, с помощью метода `group_derivative` группируем слагаемые с производными. Затем присваиваем переменным `leftPart` и `rightPart` значения соответственно f1 и f2, и все дальнейшие действия будем производить с ними.

2. Выполняем поиск производной в левой части методом `find` с одновременным переносом всех свободных от производной членов в правую часть. Если производная не была найдена, то `find` вернет $false$, и программа выдаст ошибку, т. к. исходное уравнение не является ОДУ первого порядка.

3. После проверки и приведения уравнения к виду $y'f(x,y) = g(x,y)$ начинается процесс самого разделения переменных. Поочередно вызываются методы `solve_left_part()` и `solve_right_part()`. В процессе работы этих методов заполняются глобальные массивы `elemsX` и `elemsY`.

Когда разбор уравнения завершен, произведения элементов этих массивов интегрируются по x и y соответственно. Получаем левую и правую части решения типа VectorS.

3.2. Метод `group_derivative`

Ищет слагаемые с производными и группирует их. Если функция f является произведением или производной, то алгоритм возвращает ее же. Иначе поднимается сумма в f методом `ExpandForYourChoise` из класса `CanonicForms`, создается объект expand, которому присваивается значение f. Далее проверяем все аргументы функции. Если аргумент является функцией (а это может быть только произведение и деление), то проверяем методом `factor_d`, входит ли в число ее аргументов производная y'. Если входит, то тем же методом получаем множитель при y' и добавляем его в массив. Одновременно вычитаем эту функцию из f. После того, как просмотрели все аргументы f, собираем новую функцию из суммы произведения y' и суммы элементов из массива и оставшихся аргументов исходной функции. Получившуюся функцию и возвращаем.

Пример:

Вход: $1 + xy' + y + y'$

Выход: $(x+1)y' + y + 1$

3.3. Метод `isDerivative`

Определяет, является ли функция производной первого порядка. Если входное выражение — функция, ее имя в классе `F` совпадает с `F.d`, порядок производной равен 1, то возвращается $true$, иначе $false$.

3.4. Метод `factor_d`

Возвращает множитель при производной. Циклом проходит по аргументам функции. Если находится производная, то возвращается частное от деления функции на y'. Алгоритм вернет 0, если производная в функции не найдена, или 1, если функция сама является производной.

Пример:

Вход: $(x+1)y'$
Выход: $x+1$

3.5. Метод `find`

Ищет производную в левой части и переносит свободные слагаемые в правую часть. Если левая часть не является функцией, возвращает $false$, т. к. в этом случае она не содержит производную. Если левая часть является производной, то возвращает $true$. Затем проверяется тип функции (произведение, сумма) с раскрытием дерева функции. Если находится производная, то она и ее множители остаются в левой части, а все остальное с противоположным знаком переносится в правую часть.

Пример:

Вход: $(x+1)y'+y+1=0$
Выход: $(x+1)y'=-y-1$

3.6. Методы `varX, varY`

Метод `varX` проверяет, зависит ли полином/функция от переменной x и не зависит от y. Программа работает рекурсивным способом, проходя по всем элементам дерева функции. Если на входе метода появляется полином, проверяется наличие в нем той или иной переменной путем проверки значений массива степеней элементов. Если в результате работы метода встречается переменная y, программа вернет $false$. Иначе возвращается $true$.

Метод `varY` работает аналогично, с той лишь разницей, что выражение проверяется на наличие в нем переменной y и отсутствие x.

3.7. Методы `solve_right_part, solve_left_part`

Проводят разбор подготовленных частей `rightPart, leftPart`.

Правая часть может быть одним из трех следующих типов.

1) Число. В таком случае это число добавляется в массив `elemsX`, и обработка правой части на этом завершается возвратом $true$.

2) Полином. Сначала его нужно факторизовать (разложить на простые множители). После чего проверяем, от какой переменной зависит каждый множитель, и поочередно добавляем их в нужный массив. Если вдруг встретился множитель, зависящий одновременно и от x, и от y, программа вернет $false$.

3) Функция. Правая часть конвертируется в полином посредством метода `PolynomialConvert` класса `CanonicForms`. При этом в новом кольце появляются новые переменные, каждая из которых заменяет в полученном полиноме определенную функцию. Все последующие действия

такие же, как при работе с полиномом. Перед добавлением элементов в массивы `elemsX` или `elemsY` их нужно преобразовать из полиномов обратно в функции. Этим занимается метод `unconvert`.

Метод `solve_left_part` работает аналогично, только левая часть будет строго являться функцией, т. к. в ее состав входит производная. Саму производную при разделении переменных пропускаем.

Пример: если задано уравнение $(x+1)y' = y+1$, то после работы методов `solve_left_part` и `solve_right_part` массив `elemsY` будет содержать выражение $1/(-y-1)$, а `elemsX` - выражение $1/(x+1)$. После интегрирования этих выражений получим левую и правую части решения.

3.8. Метод `unconvert`

Восстанавливает функцию из полинома. Работает рекурсивно. Сначала осуществляется восстановление до функции входного элемента. Затем вызывается метод с входных параметром из аргумента полученной функции. С каждым вызовом метода в случае, если аргумент — полином или функция, возвращается функция с тем же именем, что и входная функция, и аргументами, принятыми из следующего вызова метода. Программа завершает работу, пройдя дерево функции до всех его листьев.

3.9. Метод `findRoot`

Выполняет поиск корней в выражении с последующим вызовом метода `factorRoot`.

3.10. Метод `factorRoot`

Выполняет факторизацию корней.

Пример:

Вход: $\sqrt{4x+3xy}$
Выход: $\sqrt{x}\sqrt{4+3y}$

3.11. Метод `factor_exp`

Раскладывает экспоненту со сложным показателем степени в произведение экспонент.

Пример:

Вход: $e^{4x+3y-5}$
Выход: $e^{4x}e^{3y}e^{-5}$

3.12. Метод `revers`

Возвращает выражение, обратное входному.

4. Решение однородных дифференциальных уравнений

Для решения однородных дифференциальных уравнений первого порядка был создан класс `SolveHDE`. В процессе решения уравнений этого вида будем использовать замены $y = yx$, $y' = y'x + y$.

Методы класса: `solve, replace1, replace2`.

4.1. Метод `solve`

Метод подготавливает к решению и решает входное уравнение.

Сначала преобразуем входное уравнение в уравнение с разделяющимися переменными с помощью метода `replace1`, который вернет функцию с замененными переменными. После этого отправляем полученное уравнение с разделяющимися переменными на решение в класс `SolveDiffEqWithSeparableVariables`. В результате получаем элемент класса `VectorS`. В случае успешного решения он будет содержать в себе 2 элемента (проинтегрированные левая и правая части), если же разделить переменные не удалось, то один элемент (строка ошибки). Последний случай означает, что исходное уравнение не являлось однородным. В случае успешного решения преобразованного уравнения необходимо провести обратную замену переменных в методе `replace2`.

4.2. Метод `replace1`

Метод преобразует однородное уравнение в уравнение с разделяющимися переменными путем замены переменных. Работает рекурсивно, пробегая по всему дереву функции. Встречая производную y', заменяет ее на $y'x+y$, а переменную y заменяет на yx.

4.3. Метод `replace2`

Метод выполняет обратную замену переменных. Работает рекурсивно, пробегая по всему дереву функции. Заменяет переменную y на $\frac{y}{x}$.

5. Решение уравнений в полных дифференциалах

Алгоритмы решения уравнений в полных дифференциалах находятся в классе `SolveEqInTotalDiff`.

Уравнение $Pdx+Qdy=0$ поступает на вход основного метода `solve` параметрами P и Q. Это уравнение гарантированно является уравнением в полных дифференциалах, т. к. оно прошло проверку в классе `SolveDiffEq`.

Методы класса: `solve, FPsimplify`.

5.1. Метод `solve`

Метод решает уравнение по стандартному алгоритму [5].

Рассмотрим пример. Решим уравнение $(2x-y+1)dx+(2y-x-1)dy=0$.

Здесь $P=2x-y+1$, $Q=2y-x-1$.

Обозначим $A=\int Pdx = -yx+x^2+x+\phi(y)$.

Найдем производную A'_y, обозначим ее буквой B:

$B=-x+\phi(y)'_y$.

Так как $B+\phi(y)'_y=Q$, то $\phi(y)'_y=Q-B$:

$\phi(y)'_y=Q-B=2y-1$.

Тогда

$\phi=\int\phi(y)'_ydy=y^2-y+C$.

Подставляем полученное значение ϕ в A и приравниваем все выражение к нулю — это и будет решение. В нашем случае получили $-yx+x^2+x+y^2-y+C=0$.

Решение возвращается в объекте `VectorS(new Element[]f1, 0)`, где f1 — полученное решение.

5.2. Метод `FPsimplify`

Раскрывает скобки в выражении типа FactorPol.

В результате выполнения метода `solve` получаем выражение, являющееся левой частью решения, а правую принимаем за ноль.

6. Примеры

Таблица 1: Примеры ОДУ с разделяющимися переменными

Уравнение	Решение
$(x^2+1)exp(y)dy-2x(exp(y)+1)dx=0$	$abs(1+exp(y))=exp(ln(abs(x^2+1))+C)$
$y'=x/cos(y)$	$y=arcsin(1/2x^2+C)$
$(x+2)\sqrt{y}-(3xy')=0$	$y=((1/6)x+(1/3)ln(abs(x))-(1/6)C)^2$
$exp(x)+y'=exp(x)$	$y=C$
$(yx^2+y)y'=y^2+1$	$abs(y^2+1)=exp(2C-2arctg(1/x))$
$x\cdot exp(-y+x)=y'$	$y=ln(x\cdot exp(x)-C+exp(x))$
$y^2y'=x$	$y=\sqrt[3]{3C+(3/2)x^2}$

Таблица 2: Примеры однородных дифференциальных уравнений

Уравнение	Решение
$2x^3y'=-y^3+2yx^2$	$y=\sqrt{(-x^2)/(C-ln(abs(x)))}$
$(x^2y'+y^2)=yxy'$	$ln(abs(y/x))-y/x=-ln(abs(x))+C$
$xy'=y-x\cdot exp(y/x)$	$y=-xln(ln(abs(x))-C)$
$xy'=\sqrt{yx}+y$	$y=((1/2)C+(1/2)ln(abs(x)))^2x$
$(yy'+4x)=xsin(cos(exp(y/x)))$	$\int(y/(sin(cos(exp(y)))-(4+y^2)))dy_{y=y/x}=ln(abs(x))+C$
$y'=arcsin((y^2+x^2)/2yx)$	$\int(1/(arcsin((y^2+1)/2y)-y))dy_{y=y/x}=ln(abs(x))+C$

Таблица 3: Примеры уравнений в полных дифференциалах

Уравнение	Решение
$(-y+2x+1)dx+(2y-x-1)dy=0$	$y^2-yx-y+x^2+x+C=0$
$(-3y^2+3x^2+4x)dx-(6yx+4y)dy=0$	$-3y^2x-2y^2+x^3+2x^2+C=0$
$(3y^2+6y-3x^2)dx+(6yx+6x)dy=0$	$3y^2x+6yx-x^3+C=0$
$(2x(1-exp(y))/(x^2+1)^2)dx+(exp(y)/(x^2+1))dy=0$	$(x^2+exp(y))/(x^2+1)+C=0$
$(6y^2x+3x^2)dx+(4y^3+6yx^2)dy=0$	$y^4+3y^2x^2+x^3+C=0$
$2yxdx+(3y^2+x^2)dy=0$	$y^3+yx^2+C=0$
$(y\cdot exp(x)+cos(y)sin(x)-y^6x^3)dx+$ $+(exp(x)+sin(y)cos(x)-3/2y^5x^4)dy=0$	$y\cdot exp(x)-(cos(y)cos(x)+$ $+1/4y^6x^4)+C=0$

7. Приложение

В раздел «Помощь» были добавлены примеры решения разных видов ОДУ. На рис. 1 представлена часть раздела на русском языке.

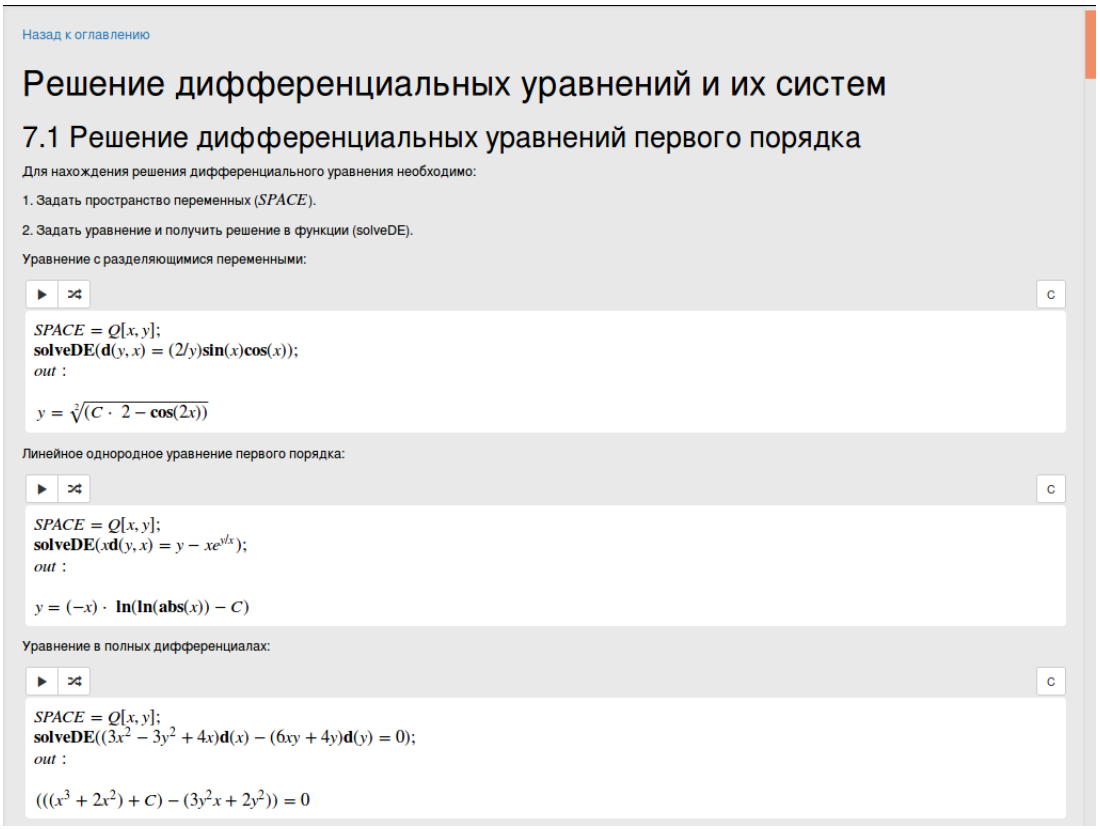

Рис. 1. Примеры решения ОДУ в разделе «Помощь»

8. Заключение

Были разработаны алгоритмы символьного решения некоторых типов обыкновенных дифференциальных уравнений. Эти алгоритмы были программой реализованы в системе компьютерной алгебры Math Partner.

СПИСОК ЛИТЕРАТУРЫ

1. Math partner URL: http://mathpar.cloud.unihub.ru (accessed: 16.10.2017)

2. *Малашонок Г.И.* Руководство по языку «MATHPAR». Тамбов: Издательский дом ТГУ им. Г.Р. Державина, 2013.

3. *Malaschonok G.I.* MathPartner Computer Algebra // Programming and Computer Software, 2017. V. 43. № 2. P. 112–118.

4. *Ноутон П., Шилдт Г.* Java 2. С.-Пб: БХВ-Петербург, 2008.

5. *Виленкин Н.Я., Доброхотова М.А., Сафонов А.Н.* Дифференциальные уравнения. М.: Просвещение, 1984.

БЛАГОДАРНОСТИ: Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект № 16-07-00420).

Поступила в редакцию 17 августа 2017.

Глазков Сергей Александрович, Тамбовский государственный университет им. Г.Р. Державина, г. Тамбов, Российская Федерация, магистрант, институт математики, естествознания и информационных технологий, e-mail: sergey.glazkov96@yandex.ru

Рыбаков Михаил Анатольевич, Тамбовский государственный университет им. Г.Р. Державина, г. Тамбов, Российская Федерация, старший преподаватель кафедры функционального анализа, e-mail: mixail08101987@mail.ru

UDC 517.925
DOI: 10.20310/1810-0198-2017-22-6-1268-1276

THE SYMBOLIC SOLUTION OF ORDINARY DIFFERENTIAL EQUATIONS IN THE COMPUTER ALGEBRA SYSTEM MATH PARTNER

© S. A. Glazkov, M. A. Rybakov

Tambov State University named after G.R. Derzhavin,
33 Internatsionalnaya st., Tambov, Russian Federation, 392000
E-mail: sergey.glazkov96@yandex.ru, mixail08101987@mail.ru

The article deals with algorithms of symbolic solution of ordinary differential equations, their software implementation in the computer algebra system Math Partner. Classes are described for solving differential equations with separable variables, homogeneous differential equations, and equations in complete differentials.

Keywords: differential equations; computer algebra system; MathPartner

REFERENCES

1. Math partner URL: http://mathpar.cloud.unihub.ru (accessed: 16.09.2017)
2. *Malaschonok G.I.* Mathpar Language Guide. Tambov: Publishing House of TSU, 2013.
3. *Malaschonok G.I.* MathPartner Computer Algebra // Programming and Computer Software, 2017. V. 43. № 2. P. 112–118.
4. *Nouton P., Schildt H.* Java 2. S.-Pb: BHV-Peterburg, 2008.
5. *Vilenkin N.Ya., Dobrohotova M.A., Safonov A.N.* Differential equations. M.: Prosveshchenie, 1984.

ACKNOWLEDGEMENTS: The work is partially supported by the Russian Fund for Basic Research (project № 16-07-00420).

Received 17 August 2017

Glazkov Sergey Aleksandrovich, Tambov State University named after G.R. Derzhavin, Tambov, Russian Federation, Master's student, Institute Mathematic, Natural Sciences and Information Technologies, e-mail: sergey.glazkov96@yandex.ru

Rybakov Michail Anatolevich, Tambov State University named after G.R. Derzhavin, Tambov, Russian Federation, Senior Lecturer of Functional Analysis Department, e-mail: mixail08101987@mail.ru

Для цитирования: *Глазков С.А., Рыбаков М.А.* Алгоритмы решения простых типов обыкновенных дифференциальных уравнений в Math Partner // Вестник Тамбовского университета. Серия Естественные и технические науки. Тамбов, 2017. Т. 22. Вып. 6. С. 1268–1276. DOI: 10.20310/1810-0198-2017-22-6-1268-1276.

For citation: Glazkov S.A., Rybakov M.A. Algoritmy resheniya prostyh tipov obyknovennyh differentsial'nyh uravneniy v Math Partner [The symbolic solution of ordinary differential equations in the computer algebra system Math Partner]. *Vestnik Tambovskogo universiteta. Seriya Estestvennye i tekhnicheskie nauki – Tambov University Reports. Series: Natural and Technical Sciences*, 2017, vol. 22, no. 6, pp. 1268–1276. DOI: 10.20310/1810-0198-2017-22-6-1268-1276 (In Russian, Abstr. in Engl.).