

ALGORITHM FOR INTEGRATING FACTOR FOR A NON-EXACT LINEAR FIRST ORDER ORDINARY DIFFERENTIAL EQUATION

Emmanuel Appoh Andam^{*}

William Obeng – Denteh^{**}

Lawrence Obiri – Apraku^{***}

Wallace Agyei^{****}

Evans Atteh^{*****}

Abstract

In this work we present a simple algorithm for computing integrating factors of certain classes of first order Ordinary Differential Equations (ODE), as well as the foundation of the algebraic basis for these methods. Matlab has been used to generate the algorithm for computing the integration factors. The aim of this paper is of a rather practical nature, i.e. the presentation of practical and efficient algorithm for the generation of integration factors for non – exact Ordinary Differential Equations, which can directly be implemented in the framework of a general purpose computer algebra system.

Keywords: Differential Equation, Integrating factor, Ordinary differential equation, Exact, Non-exact.

** Department of Mathematics, Mansoman Senior High School, Manso Atwere, Amansie West, Ashanti, Ghana.*

*** Department of Mathematics, Kwame Nkrumah University of Science and Tehnology, Kumasi, Ghana.*

**** Depatment of Maathematics, Boakye-Tromo Senior High School, Duayaw- Nkwanta, B/A, Ghana.*

***** Department of Mathematics, Tweneboa Kodua Senior High School, Kumawu, Ashanti, Ghana.*

****** Department of Mathematics, Esaase-Bontefufuo Senior High Tecnical School, Esaase, Amansie West, Ashanti, Ghana.*

1. Introduction

A differential equation, according to [1] is an equation involving an unknown function and its derivatives. Differential equations are a powerful tool in constructing mathematical models for the physical world. Many physical processes are modeled in terms of *differential equations*, i.e., equations for an unknown function in terms of its derivatives.

The history of Ordinary Differential Equations (ODEs) goes way back to the XVII century when two great scientists Isaac Newton and Gottfried Leibniz introduced calculus which came to place from the concept of functions [2].

Differential equations are essential tools in scientific modeling of physical problems which found their relevance in almost every human endeavor. Their use in industry is so widespread and they perform their task so well that they are clearly one of the most successful of modeling tools [3]. Examples of differential equations are found in many of the mathematical models that are derived from fields such as physics, engineering, chemistry, biology, and the earth sciences [4]. For example, consider modeling the acceleration of a ball falling through the air (considering only gravity and air resistance). The acceleration of the ball towards the ground is the acceleration due to gravity minus the acceleration due to air resistance. Gravity is a constant but air resistance is proportional to the ball's velocity. This means the ball's acceleration is dependent on its velocity. Because acceleration is the derivative of velocity, solving this problem requires a differential equation.

Many of the principles, or laws, underlying the behaviour of the natural world are statements or relations involving rates at which things happen. When expressed in mathematical terms the relations are equations and the rates are derivatives [5]. The attempt to solve physical problems led gradually to mathematical models involving an equation in which a function and its derivatives play important roles. Ordinary differential equations arise in many applications and numerous other fields and it describes relations between variables and their derivatives.

According to [6], the study of differential equations has three main purposes:

1. To discover the differential equation that describes a specified physical situation.
2. To find, either precisely or approximately, the appropriate solution of that equation.
3. To analyze and interpret the solution that is found.

In the development of the subject of differential equations, we distinguish two broad distinct streams, namely:

1. A procedure to obtain a definite or one of the definite types, either in closed forms, which are rarely possible or else by some process of approximation.
2. A process to abandon all attempts to reach an exact or approximate solution, and this will lead to obtaining information about the whole class of solution.

There are lots of aspects to learn about the solutions to any given differential equation without solving it. For example, considering the differential equation $X' = f(X)$; we can find the equilibrium points by finding the zeros X' of f .

The possibility of converting a differential equation that is not exact into an exact equation is by multiplying the equation by a suitable integrating factor. The basis of the integrating factor formulation is in integrating the linear part of the differential equation precisely. There are several studies ([7], [8]) on the integrating factor formulation for an ordinary differential equation (ODE). According to reference [9], both sides of the ordinary differential equation (ODE) are multiplied by an appropriate integrating factor, and a differential equation is obtained in which change variables are changed so that the linear part could be solved exactly [10].

In this paper, the authors seek to formulate an algorithm using Matlab software to generate the appropriate integrating factor for a non – exact differential equation to be converted to an exact differential equation, and hence solved exactly for its solution.

2. Problem Statement and Way Forward

The main problem that was considered in this paper was finding an integrating factor for a non – exact differential equation. A differential equation is called exact when it can be written as a total derivative of an appropriate function, called potential function. When the equation is written in that way it is simple to find implicit solutions. Given an exact equation, we just need to find a potential function, and a solution of the differential equation will be determined by any level surface of that potential function. There are differential equations that are not exact but they can be converted into exact equations when they are multiplied by an appropriate function, called an integrating factor. An integrating factor converts a non-exact equation into an exact equation. For

linear equations we computed integrating factors that transformed the equation into a derivative of a potential function.

3. Methodology

3.1. Exact Ordinary Differential Equation

Let the functions M , N , M_y and N_x , where the subscripts denote partial derivatives, be continuous in the rectangular region

$R: \alpha < x < \beta, \gamma < y < \delta$, then

$$M(x, y) + N(x, y)y' = 0 \quad (1)$$

is an exact differential equation in R if and only if, $M_y(x, y) = N_x(x, y)$ at each point of R , that is there exist a function φ satisfying,

$$\varphi_x(x, y) = M(x, y), \varphi_y(x, y) = N(x, y) \quad (2)$$

if and only if M and N satisfy

$$M_y(x, y) = N_x(x, y).$$

3.2. Non – Exact Ordinary Differential Equation

The differential equation: $M(x, y) + N(x, y)y' = 0 \quad (3)$

is termed non – exact if and only if

$$M_y(x, y) \neq N_x(x, y) \quad (4)$$

in the given region $R: \alpha < x < \beta, \gamma < y < \delta$. In this case solving the differential equation becomes difficult, if not possible. The question then is how do we solve a non – exact ordinary differential equation? That is where an integrating factor is needed. When the whole differential equation (both sides) is multiplied by an appropriate integrating factor it will make the non – exact differential equation an exact differential equation, hence making it solving easier to solve.

3.3. The Integrating factor Method

An integrating factor for a differential equation

$$M(x, y)dx + N(x, y)dy = 0 \quad (5)$$

is a non-vanishing function $\mu(x, y)$ such that the product

$$(\mu M)dx + (\mu N)dy = 0 \quad (6)$$

is an exact differential, that is

$$\partial M(x, y)/\partial x = \partial N(x, y)/\partial y \quad (7)$$

That is, it is sometimes possible to convert a differential equation that is not exact to an exact equation by multiplying the equation with a suitable integrating factor. An integrating factor is guaranteed to exist if the given differential equation actually has a solution.

Consider the general first order linear differential equation

$$y' + p(x)y = g(x) \quad (8)$$

We choose a function $\mu(x)$ such that we multiply equation (8) by the function $\mu(x)$. By so doing, the left hand side of equation (8) can be written as the derivative of the single function $\mu(x)y$.

We then obtain the expression

$$\mu(x)[y' + p(x)y] = [\mu(x)y]' = \mu(x)y' + \mu'(x)y \quad (9)$$

Thus $\mu(x)$ must satisfy the equation

$$\mu(x)p(x)y = \mu'(x)y \quad (10)$$

Assuming that $\mu(x) > 0$, we then obtain

$$\mu'(x)/\mu(x) = p(x) \quad (11)$$

Since $\mu'(x)/\mu(x)$ is the derivative of $\ln\mu(x)$,

we then have

$$\ln\mu(x) = \int p(t)dt \quad (12)$$

We finally arrive at the expression for the integrating factor and is given as

$$\mu(x) = \exp^{\int p(t)dt} \quad (13)$$

4. Results and Discussion

The theoretical and manual generation of the integration factor is explained above. Matlab was then used to develop codes for the generation of the integration factor and it found in the **Appendix**. The algorithm provides a convenient way of not going through all theory of the integrating factor generation.

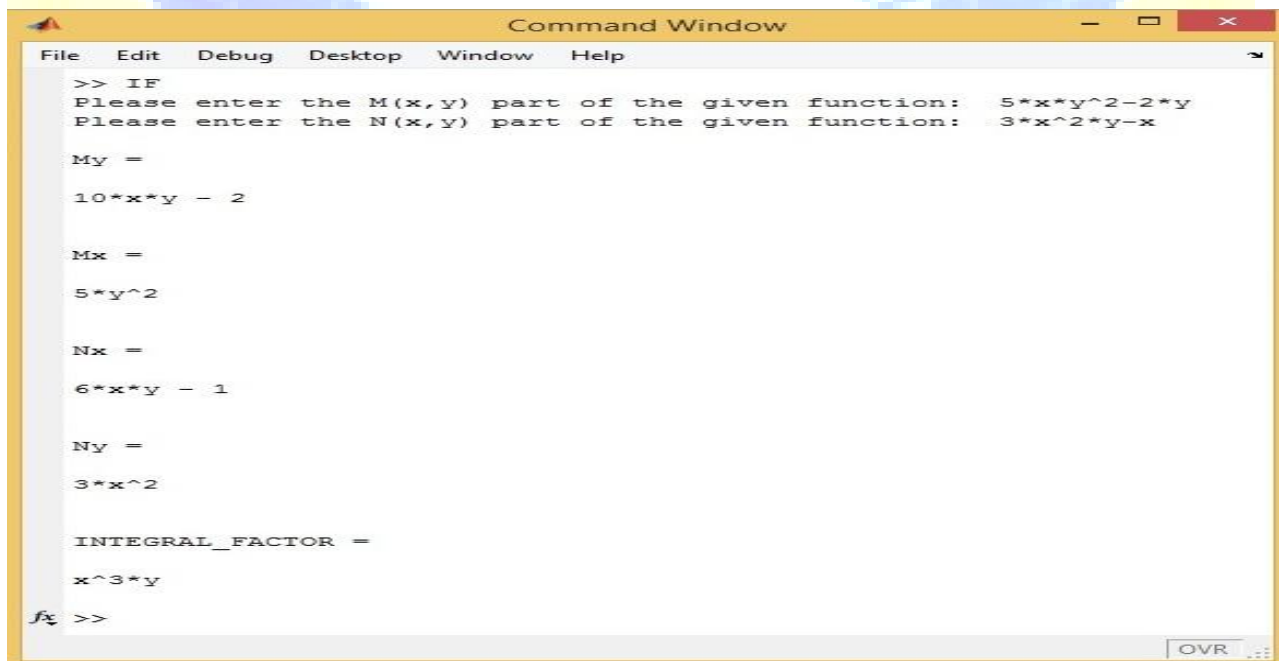
The authors explored the algorithm with the following examples, with each of them being non – exact differential equations;

$$\text{Example 1: } (5xy^2 - 2y)dx + (3x^2 - x)dy = 0$$

$$\text{Example 2: } (3e^x y + x)dx + e^x dy = 0$$

$$\text{Example 3: } (x + y)\sin y dx + (x \sin y + \cos y)dy = 0$$

See **Figures' 1, 2 and 3** for the operations of the Matlab code. In each of the examples, the Matlab software generated the suitable integrating factor for non – exact differential equations. With this, the integrating factor can be multiplied with their respective equations and converted to exact differential equations and their solutions can be easily solved for.



```
Command Window
File Edit Debug Desktop Window Help
>> IF
Please enter the M(x,y) part of the given function: 5*x*y^2-2*y
Please enter the N(x,y) part of the given function: 3*x^2*y-x

My =
10*x*y - 2

Mx =
5*y^2

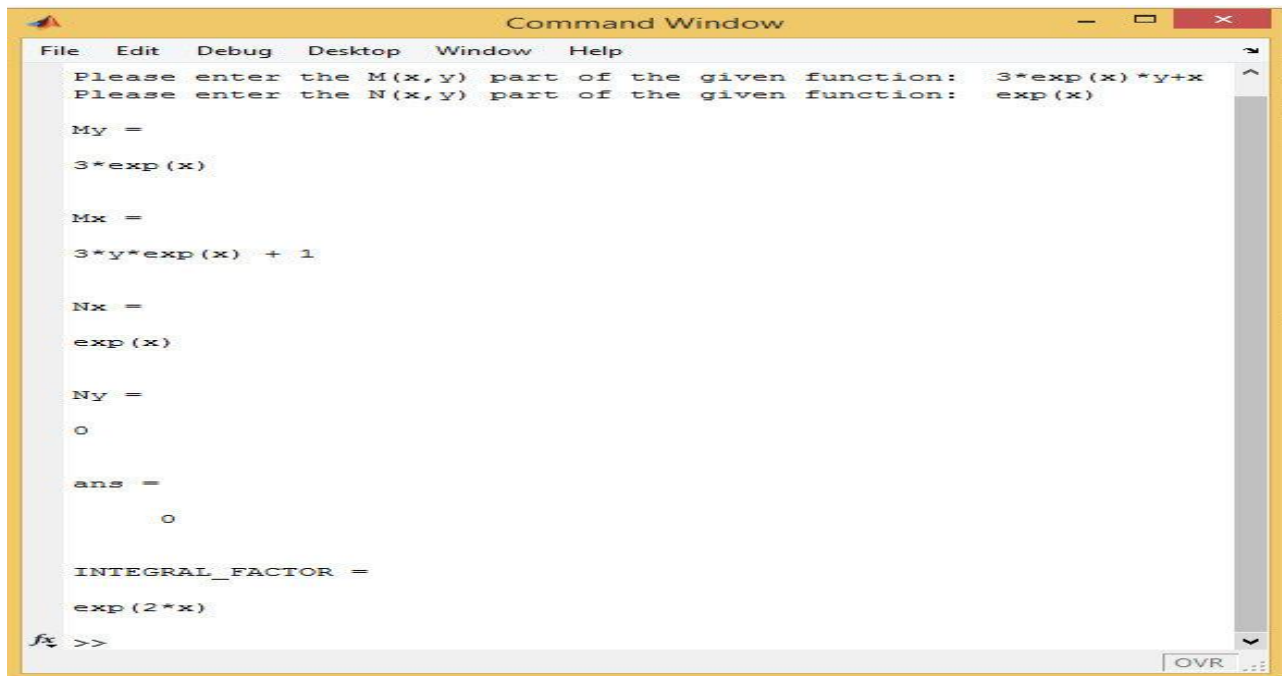
Nx =
6*x*y - 1

Ny =
3*x^2

INTEGRAL_FACTOR =
x^3*y

fx >>
```

Figure 1. Example 1 depicted in the Matlab command window



```

Command Window
File Edit Debug Desktop Window Help
Please enter the M(x,y) part of the given function: 3*exp(x)*y+x
Please enter the N(x,y) part of the given function: exp(x)

My =
3*exp(x)

Mx =
3*y*exp(x) + 1

Nx =
exp(x)

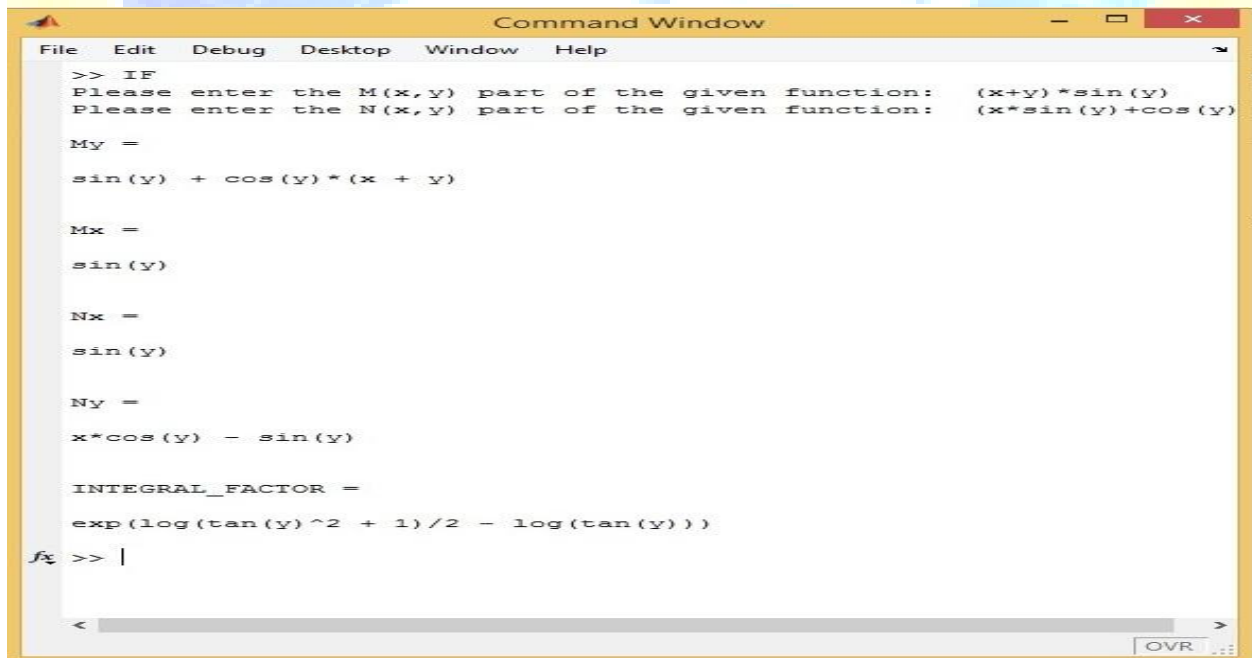
Ny =
0

ans =
0

INTEGRAL_FACTOR =
exp(2*x)

fx >>
    
```

Figure 2. Example 2 depicted in the Matlab command window



```

Command Window
File Edit Debug Desktop Window Help
>> IF
Please enter the M(x,y) part of the given function: (x+y)*sin(y)
Please enter the N(x,y) part of the given function: (x*sin(y)+cos(y))

My =
sin(y) + cos(y)*(x+y)

Mx =
sin(y)

Nx =
sin(y)

Ny =
x*cos(y) - sin(y)

INTEGRAL_FACTOR =
exp(log(tan(y)^2 + 1)/2 - log(tan(y)))

fx >> |
    
```

Figure 3. Example 3 depicted in the Matlab command window

It is worthy to note that from the example being displayed, the algorithm has generated the respective integrating factors for the individual non – exact differential equation given. The

algorithm has provided a much convenient process of not going through the manual and theoretical framework of generating the integrating factors. The algorithm can be used for the examples for at your perusal.

5. Conclusion

The main aim of this work is to provide an algorithm to generate the integrating factor for a non – exact differential equation. Integrating factors are difficult to uncover and sometime the search for it is not successful. Generally, solutions for exact differential equations are determined by any level surface of that potential function. Matlab software has been used to generate the integration factors explained above. Non – exact differential equations need to be converted into exact differential equations before their solutions can be found, and this can be achieved when we multiple that equation (non – exact differential equation) with the integration factor. The integration factor converts non – exact differential equation to exact differential equation.

More often than not, much attention is not given to integrating factors despite the indispensable role it plays in differential equations both in its teaching and in books. It is either discussed briefly or left once students are able to work one or two examples.

Hopefully this should help add to the emphasis needed to place on this all important aspect of differential equations.

Acknowledgements

Authors express their gratitude to unknown reviewers for their useful comments

Competing Interest

Authors have declared that no competing interests exist.

References

- [1] Polking, J., Boggess, A. and Arnold, D. (2005). *Differential equations with boundary value problems* (2nd Ed.). In S. Yagan (Ed.). New Jersey, USA: Pearson Education, Inc.
- [2] Kline, M. (1972). *Mathematical Thought from Ancient and Modern Times*, Oxford University Press, Oxford.
- [3] Borrelli, R. L. and Colema, C. S. (2004). *Differential equations: a mathematical modeling in*

- perspective*. USA: John Wiley and Sons Inc.
- [4] Hundsdorfer, W. and Verwer, J. G. (2003). Numerical solution of time - dependent advection-diffusion-reaction equations. Springer - Verlag, Berlin.
- [5] Boyce, W. E. and DiPrima, R. C. (2001). *Elementary differential equations and boundary value problems*. Toronto: John Wiley and Sons Inc.
- [6] Edwards C. H. and Penny D. E. (2004). *Differential Equations and Boundary Value Problems, Computing and Modeling*. Third Edition. Pearson Education, Inc., New Jersey, USA.
- [7] Lawson, J.D. (1967). Generalized Runge-Kutta Processes for Stable Systems with Large Lipschitz Constants, *SIAM J. Numer Anal*, 372-380.
- [8] Berland, H. and Skaestad, B. (2006). Solving the Nonlinear Schrödinger Equation Using Exponential Integrators, *Norwegian Society of Automatic Control*, 27, 201- 217.
- [9] Kassam, A. K. High Order Time stepping for Stiff Semi-Linear Partial Differential Equations, PhD thesis, Oxford University, (2004).
- [10] Berland, H., Skaestad, B. and Wright, W. M. (2007). EXPINT - A Matlab Package for Exponential Integrators, *ACM Transactions on Mathematical Software*, 33, Article Number4.

Appendix

% a programme written in matlab to find the integral factor of a non-exact ordinary differential equation (ode)

%NOTE

%this programme works for problems with variables not exceeding two

%this programme was written to accept only x and y as input variables and as such if the problem given is in differnt variables, it must be converted to x

%and y

% inputs required are: $M(x,y)dx$ and $N(x,y)dy$

% where $M(x,y)$ = the part of the o.d.e with dx attached to it

% $N(x,y)$ = the part of the o.d.e with dy attached to it

% M_y = partial differential of $M(x,y)$ with respect to y

% M_x = partial differential of $M(x,y)$ with respect to x

% N_x = partial differential of $N(x,y)$ with respect to X

% N_y = partial differential of $N(x,y)$ with respect to y

symsxyuv

M = input('Please enter the M(x,y) part of the given function:');

N = input('Please enter the N(x,y) part of the given function:');

M_y = diff (M,y)

M_x = diff (M,x)

N_x = diff (N,x)

N_y = diff (N,y)

a = simple(($M_y - N_x$)/N);

b = simple($M_y - N_x$)/(-M);

c = simple($M_x + N_y$);

```

d = x^u*y^v;

sa = findsym(a);      %find the variables in the expression (My-
Nx)/N
sb = findsym(b);      %find the variables in the expression (My-
Nx)/(-M)
facty = simple(M/y);
factx = simple(N/x);

if My == Nx
disp ('IT IS AN EXACT DIFFERENTIAL EQUATION AND HENCE THERE IS
NO NEED FOR A INTEGRAL FACTOR')
elseif isempty(sa)...
&findsym(My) == findsym(Nx)...
&findsym(Nx) == findsym(N)...
&findsym(N) == x

INTEGRAL_FACTOR = exp(int(a,x))
elseif sa == x

INTEGRAL_FACTOR = exp(int(a,x))
elseif isempty(sb)...
&findsym(My) == findsym(Nx)...
&findsym(Nx) == findsym(M)...
&findsym(M) == y

INTEGRAL_FACTOR = exp(int(b,y))
elseif sb == y

INTEGRAL_FACTOR = exp(int(b,y))
elseif y*facty + x*factx == 0
facty ~= factx

INTEGRAL_FACTOR = 1/(x*M-y*N)

```

```

else
d1 = M*d;
d2 = N*d;
d3 = diff(d1,y,1);
d4 = diff(d2,x,1);
A1 = solve(...
subs(d3,{x,y},{0.1,0.2})- ...
subs(d4,{x,y},{0.1,0.2}) ...
,u);
B1 = solve(...
subs(d3,{x,y},{0.1,0.2})- ...
subs(d4,{x,y},{0.1,0.2}) ...
,v);

%unknown coefficient u
Const = [ solve(subs(d3- ...
                d4,{v},{B1}),u)
solve(subs(d3- ...
                d4,{u},{A1}),v) ];

%unknown coefficient v
INTEGRAL_FACTOR = x^(Const(1))*y^(Const(2))
end

```