{MERAA} Mathematics in Education, Research and Applications ISSN 2453-6881

Received: 2021-01-10
Accepted: 2021-02-15
Online published: 2021-09-30
DOI: https://doi.org/10.15414/meraa.2021.07.01.01-09

***Original Paper***

# An algorithm for solving of Euler parameters differential equations system

**Jozef Rédl** [*]

Slovak University of Agriculture in Nitra, Faculty of Engineering, Institute of Design and Engineering Technologies, Slovak Republic

**ABSTRACT**

The design an optimal numerical method for solving a system of ordinary differential equations simultaneously is described in this paper. System of differential equations was represented by a system of linear ordinary differential equations of Euler's parameters called quaternions. The components of angular velocity were obtained by the experimental way. The angular velocity of the centre of gravity was determined from sensors of acceleration located in the plane of the centre of gravity of the machine. The used numerical method for solving was a fourth-order Runge-Kutta method. The stability of solving was based on the orthogonality of a direct cosine matrix. The numerical process was controlled on every step in numerical integration. The algorithm was designed in the C# programming language.

**KEYWORDS**: spatial dislocation, quaternion, numerical integration

**JEL CLASSIFICATION: C63**

## INTRODUCTION

The goal of this contribution is that the published research supplement the missing part of many scientific books and scientific articles dealing with spacecraft attitude dynamics of rigid body movement in three-dimensional space. Determination of the dislocation of a certain moving frame of the system is a very important part of applied dynamics. One of these cases refers to the robotic arms movement as defined in [5]. In even a specific case is the problem to determine the dislocation of the whole moving body in three-dimensional spaces. This case refers to a moving vehicle on the ground with respect to an inertial coordinate system. There are many methods how to determine body dislocation, for example, using GPS. But using GPS giving only coordinates of moving objects with no acceptable precision. On the other hand, the data obtained from sensors of acceleration or gyroscopic gauge, giving the more

[*] Corresponding author: Assoc. prof. Jozef Rédl, PhD., Institute of Design and Engineering Technologies, Faculty of Engineering, Slovak University of Agriculture in Nitra, Tr. A. Hlinku 2, 949 76 Nitra, Slovak Republic, e-mail: jozef.redl@uniag.sk

usable data set. Angular velocities can be utilized by the system of quaternion differential equations (SQDE) [10]. Derivation of SQDE was published by [2, 3, 6]. The analytical method of solving the system of simultaneous linear differential equations (SLDE) was published by [4]. The application of the numerical integration method to solve ordinary differential equations was published by [8]. The numerical solution of SLDE was published by [9, 11]. Matrix notation of the quaternion vector space was analysed by [1, 7]. The exact and clear description of numerical solution SQDE is presented in this contribution with the utilization of matrix formalism and C# programming language.

## MATERIAL AND METHODS

### Measurement system and object

An object for measurement was a municipal services tool carrier Reform Metrac H6X. The machine on duty is depicted in Figure 1. The basic parameters and centre of gravity dislocation of the machine were published in [10].

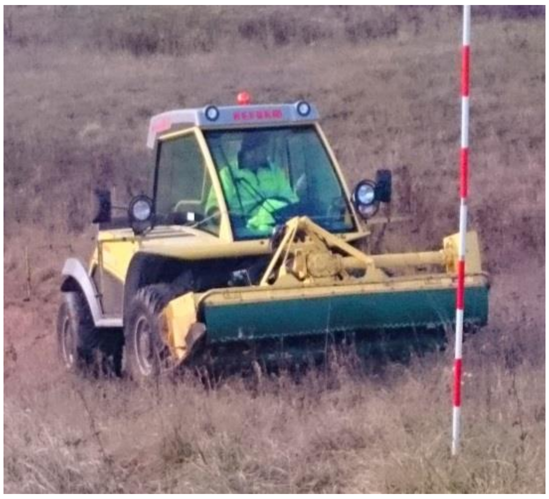
Figure 1. Metrac H6X on terrain

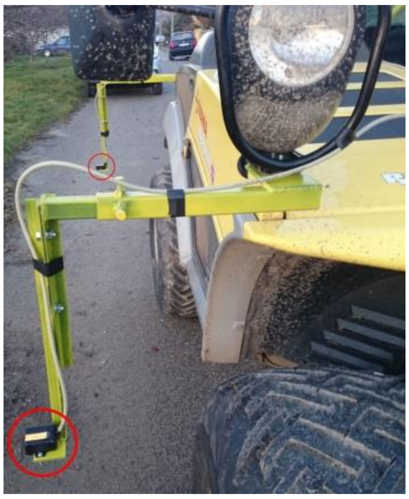
Figure 2. Dislocations of acceleration sensors on machine

For the measurement of accelerations of the machine were used the ADXL 345 sensors. The sensor measured accelerations in the XYZ axes. The sensors were dislocated in the plane of the center of gravity of the machine where the z-coordinate dimension was zero with respect to the center of gravity. Mounted sensors are depicted in Figure 2. We were provided an experimental ride with a machine with a defined trajectory as depicted in Figure 1. For application, we used the data from the ride down direction along the downhill with turning back to uphill with braking. The relevant accelerations were recorded in real-time [8]. The processed angular velocities of the center of gravity of the machine are depicted in Figure 3.

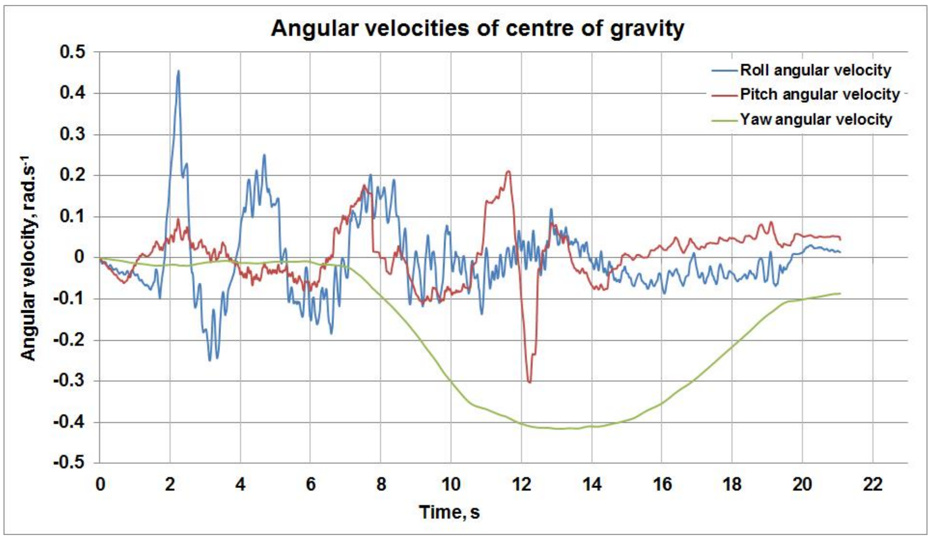

Figure 3. Angular velocities of centre of gravity of the machine

## Numerical method

For stability of numerical solving of system of differential equations, we chose the fourth order Runge-Kutta numerical method. The systems of simultaneous differential equations are in the form (01).

$$\frac{d\,{}^1y}{dx} = {}^1f\left(x,{}^1y,\ldots,{}^ny\right)$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \qquad (01)$$
$$\frac{d\,{}^ny}{dx} = {}^nf\left(x,{}^1y,\ldots,{}^ny\right)$$

Written in the shorter form of system (01) is:

$$\frac{d\,{}^iy}{dx} = {}^if\left(x,{}^iy,\ldots,{}^ny\right), i = 1,2,3\ldots n, \qquad (02)$$

with initial conditions:

$${}^1y\left(x_0\right) = {}^1y_0,\ldots,{}^ny\left(x_0\right) = {}^ny_0 . \qquad (03)$$

Rewrite:

$${}^iy\left(x_0\right) = {}^iy_0 , i = 1,2,3\ldots n . \qquad (04)$$

For solving system (01) was used the fourth-order Runge-Kutta method, with constant step size in the next form:

$${}^iY_{(j)} = {}^iY_{(j-1)} + \frac{\Delta t}{6}\left({}^ik_1 + 2\,.{}^ik_2 + 2\,.{}^ik_3 + {}^ik_4\right), \qquad (05)$$

where the coefficients are:

$$\begin{aligned}
{}^{i}k_1 &= {}^{i}f\left(x_{(j-1)},\ {}^{1}Y_{(j-1)},\dots,{}^{n}Y_k\right), \\
{}^{i}k_2 &= {}^{i}f\left(x_{(j-1)}+\frac{\Delta t}{2},\ {}^{1}Y_{(j-1)}+\frac{\Delta t}{2}.{}^{1}k_1,\dots,{}^{n}Y_k+\frac{\Delta t}{2}.{}^{n}k_1\right), \\
{}^{i}k_3 &= {}^{i}f\left(x_{(j-1)}+\frac{\Delta t}{2},\ {}^{1}Y_{(j-1)}+\frac{\Delta t}{2}.{}^{1}k_2,\dots,{}^{n}Y_k+\frac{\Delta t}{2}.{}^{n}k_2\right), \\
{}^{i}k_4 &= {}^{i}f\left(x_{(j-1)}+\Delta t,\ {}^{1}Y_{(j-1)}+\Delta t.{}^{1}k_3,\dots,{}^{n}Y_k+\Delta t\,.{}^{n}k_3\right)\ .
\end{aligned} \tag{06}$$

Where the variable $n$ is the count of the differential equations in system and $k$ is the count of discrete points of technical functions in the interval $\langle 1, k\rangle$. In the solutions we assume that the function $f$, respectively functions ${}^{1}f.....{}^{n}f$, are the continuous in the interval $\langle j-1, j\rangle$ and satisfied with the solution on all points. Variable $\Delta t$ is the step size of the method.

## Simultaneous quaternion differential equations

In most spacecraft applications occur the SQDE in the next form:

$$\begin{bmatrix} \frac{dq_0}{dt} \\ \frac{dq_1}{dt} \\ \frac{dq_2}{dt} \\ \frac{dq_3}{dt} \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix}.\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}, \tag{07}$$

The equations can be rewritten in matrix form:

$$\left[\frac{dq_i}{dt}\right]_{i=0,1,2,3} = \frac{1}{2}\left[\omega_j\right]_{j=x,y,z}.\left[q_i\right]_{i=0,1,2,3}, \tag{08}$$

and without indexation in general form:

$$\left[\frac{dQ}{dt}\right] = \frac{1}{2}[\Omega].[Q], \tag{09}$$

where:

$\left[\frac{dQ}{dt}\right]$ - matrix of quaternion differentials,

$[\Omega]$ - matrix of angular velocities,

$[Q]$ - matrix of quaternions.

## RESULTS AND DISCUSSION

To design an efficient algorithm we have to rewrite the matrix form (07) to separate single equations system to the form with the derivatives on the left side and other members placed on the right side of the equation. By these steps, we get the system of simultaneous equations (10) as follows:

$$\frac{dq_0}{dt}=\frac{1}{2}.\left(-\omega_x.q_1-\omega_y.q_2-\omega_z.q_3\right);\frac{dq_2}{dt}=\frac{1}{2}.\left(\omega_y.q_0-\omega_z.q_1+\omega_x.q_3\right),$$
$$\frac{dq_1}{dt}=\frac{1}{2}.\left(\omega_x.q_0+\omega_z.q_2-\omega_y.q_3\right);\frac{dq_3}{dt}=\frac{1}{2}.\left(\omega_z.q_0+\omega_y.q_1-\omega_x.q_2\right). \tag{10}$$

The algorithm has the next structure.

```
$Initial\ condition\ for\ quaternion\ values \rightarrow q_0^*=1, q_1^*=0, q_2^*=0, q_3^*=0$ ;
$Begin\ cycle \rightarrow i=1;$
$j=1;$
$p=0..3;$
$\omega_{(x,y,z)0}=\omega_{(x,y,z)i-1}$; $\omega_{(x,y,z)1}=\omega_{(x,y,z)i}$; ( $\omega_{(x,y,z)0}\rightarrow\omega_{(x)0},\omega_{(y)0},\omega_{(z)0}$, convention is same for all used variables)
$q_p=q_p^*$; $\omega_{x,y,z}=\omega_{(x,y,z)i-1}$;
$solve \rightarrow f\left(q_p\right)_j$; (function defined after end of cycle)
  $j=2, p=0..3;$
  $\omega_{x,y,z}=\omega_{(x,y,z)0}+\frac{1}{2}.\left[\omega_{(x,y,z)1}-\omega_{(x,y,z)0}\right];$
  $q_p=q_p^*+\frac{\Delta t}{2}.f\left(q_p\right)_1;$
  $solve \rightarrow f\left(q_p\right)_j;$
   $j=3, p=0..3$
   $q_p=q_p^*+\frac{\Delta t}{2}.f\left(q_p\right)_2;$
   $solve \rightarrow f\left(q_p\right)_j;$
   $j=4, p=0..3;$
   $\omega_{x,y,z}=\omega_{(x,y,z)1}$; $q_p=q_p^*+\Delta t.f\left(q_p\right)_3;$
   $solve \rightarrow f\left(q_p\right)_j;$
   $q_p=q_p^*+\left\{\frac{\Delta t}{6}.\left[f\left(q_p\right)_1+2.f\left(q_p\right)_2+2.f\left(q_p\right)_3+f\left(q_p\right)_4\right]\right\};$
```

$$\textit{Creating transoformation matrix} \rightarrow \left[M_q\right]_i = \prod_i^k \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}^T ,$$

where:

$$a_{11} = q_0^2 + q_1^2 - q_2^2 - q_3^2,\ a_{12} = 2.(q_1.q_2 + q_0.q_3),\ a_{13} = 2.(q_1.q_3 - q_0.q_2),$$

$$a_{21} = 2.(q_1.q_2 - q_0.q_3),\ a_{22} = q_0^2 + q_2^2 - q_3^2 - q_1^2,\ a_{23} = 2.(q_2.q_3 + q_0.q_1),$$

$$a_{31} = 2.(q_3.q_1 + q_0.q_2),\ a_{32} = 2.(q_2.q_3 - q_0.q_1),\ a_{33} = q_0^2 + q_3^2 - q_1^2 - q_2^2,$$

and $[\ ]^T$ mean the matrix transpose on each $i$ to $k$.

We define the new variable array $\delta_i = \det\left[M_q\right]_i$, where $\delta_i$ is a determinant of the transformation matrix. The transformation matrix $\left[M_q\right]_i$ is strongly orthogonal. The precision of numerical integration is defined through the orthogonal matrix determinant value. We define the precision variable or error of numerical integration $E_r(q)$, where:

```
$i = 0$ to $k$
  $E_r(q)_i = 1 - \delta_i$
    $increment\ i;$
    $return;$
```

```
$function\ solve\ f(q_p)$
  $p = 0..3;$
$(q_0)_j = \frac{1}{2}.(-q_1.\omega_x - q_2.\omega_y - q_3.\omega_z); (q_1)_j = \frac{1}{2}.(q_0.\omega_x - q_3.\omega_y + q_2.\omega_z);$
$(q_2)_j = \frac{1}{2}.(q_3.\omega_x - q_0.\omega_y - q_1.\omega_z); (q_3)_j = \frac{1}{2}.(q_2.\omega_x + q_1.\omega_y + q_0.\omega_z);$
$end\ \ function$
```

Through the designed algorithm we were solved the transformation matrix determinant values on each step and these values are depicted in the Figure 4.

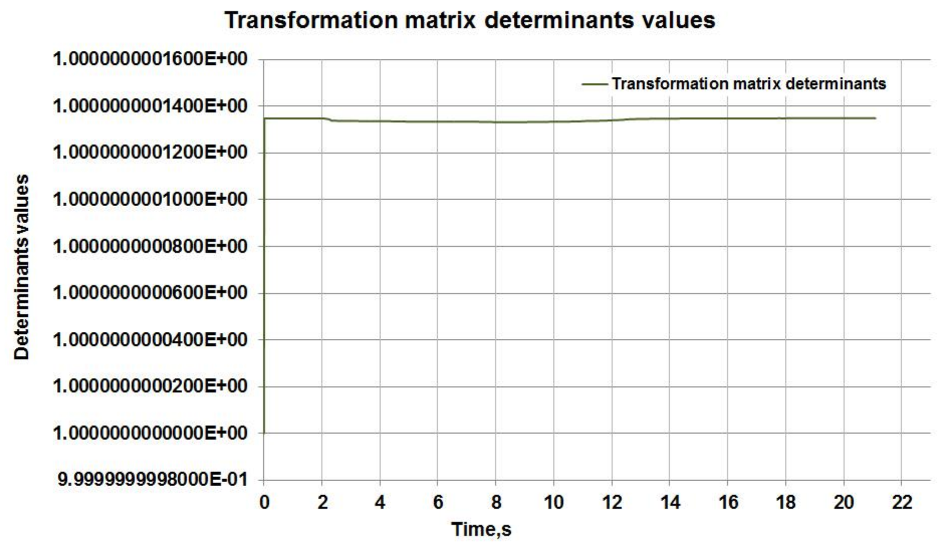

Figure 4. Transformation matrix determinants

For showing a viewable look of values of determinant we create a chart in y-axis interval $\langle 1.00000000013300, 1.00000000013500 \rangle$. The chart is depicted in Figure 5.

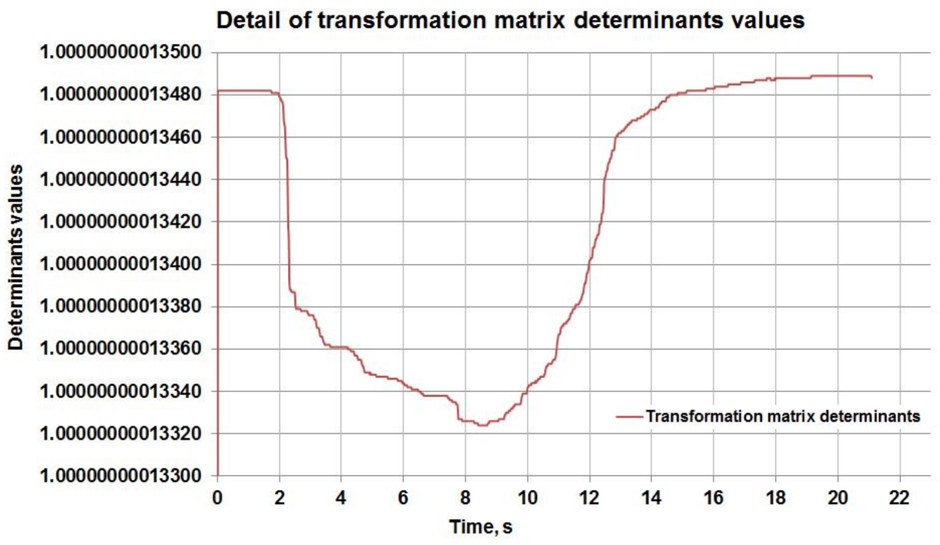

Figure 5. Transformation matrix determinants details

To control numerical integration error, we have solved the transformation matrix error on each step of counting cycles. This error is based on the orthogonality of the transformation matrix, where the determinant is equal to one. The error is depicted in Figure 6.

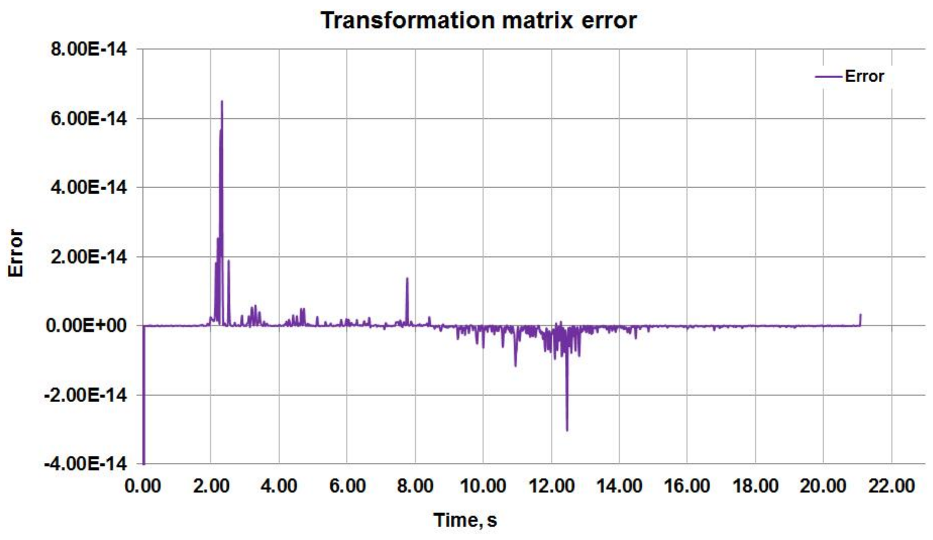

Figure 6. Numerical integration errors

## CONCLUSIONS

In this paper, we are dealing with design an algorithm for numerical solving a simultaneous system of differential equations (SSDE). For the model situation, we chose the quaternion differential equation, where the angular velocities we got from the real experiment with an agricultural machine. The aim of this work was to creating and testing the optimal algorithm for solving SSDE. As a programming language, we chose the language Visual C#, where we create the function *QTSolver* and the function *SolveFQP* as a subroutine of function *QTSolver*. The published algorithm was written here in the shortest form with using the indexation of many used variables. The published algorithm has a convention like Visual C# language. The goal of this article is showing the easy way to solve any system of linear differential equations with Runge-Kutta numerical integration method with constant step size. We were chosen the quaternion system of the differential equation as a very suitable example. The benefit of these types of equations is the orthogonality as well as the control of the stability of numerical solving. In the presented example the result of solving are quaternions from which we create the transformation matrix. The determinant of transformation matrix (see Fig.4.) in respect of orthogonality is equal to one. From this assumption, we are able to solve the error of numerical integration (see Fig. 6). The process of solving is very accurate, and the errors values are in the interval $\left\langle -4.10^{-14}, 6.10^{-14} \right\rangle$.

## REFERENCES

[1] Al-Zhour, Z. (2019). Some new linear representations of matrix quaternions with some applications. *Journal of King Saud University – Science*, Vol.31, (1), p. 42-47.
DOI: http://dx.doi.org/10.1016/j.jksus.2017.05.017

[2] Bong, W. (2008). *Space Vehicle Dynamics and Control*. 2nd ed. AIAA.
DOI: https://doi.org/10.2514/4.860119

[3] Goldstein, H. & Poole, Ch. P. & Safko, J. (2011). *Classical Mechanics*. 3rd. ed. Pearson Education. 664 p.

[4] Kou, K.I. & Xia, Y. (2018). Linear Quaternion Differential Equations: Basic Theory and Fundamental Results. *Studies in Applied Mathematics*, Vol. 141 (1). John Wiley & Sons, Inc. p. 3-45. DOI: https://doi.org/10.1111/sapm.1221

[5] Lenarcic, J. & Parenti-Castelli, V. (2018). *Advances in Robot Kinematics 2018*. Springer Proceedings in Advanced Robotics. DOI : https://doi.org/10.1007/978-3-319-93188-3

[6] Markley, F. L. & Crassidis, J. L. (2014). *Fundamentals of Spacecraft Attitude Determination and Control*. 2014 ed. Springer. 486 p.

[7] Nie, X. &Wang, Q. & Zhang, Y. (2017). *A System of Matrix Equations over the Quaternion Algebra with Applications.* Algebra Colloquium, 24, (2). pp. 233–253.
DOI: https://doi.org/10.1142/S100538671700013X

[8] Rédl, J. & Váliková, V. (2013). Application of differential geometry in agricultural vehicle dynamics. *Research in agricultural engineering*, 59, p. 34-41. DOI: https://doi.org/10.17221/49/2012-RAE

[9] Schaub, H. & Junkins, J. L. (2014). *Analytical Mechanics of Space Systems*. 3rd ed. AIAA.

[10] Váliková, V. (2015). *The analysis of the safe operating conditions of the agricultural terrain machines*. Dissertation thesis. Faculty of engineering, SUA Nitra. p. 139 (in Slovak).

[11] Wang, Q. & Wang, X. (2017). Runge–Kutta Methods for Systems of Differential Equation with Piecewise Continuous Arguments: Convergence and Stability. *Numerical Functional Analysis and Optimization*. DOI: https://doi.org/10.1080/01630563.2017.1421554

[12] Zingg, D. W. & Chisholm, T.T. (1999) Runge–Kutta methods for linear ordinary differential equations. *Applied Numerical Mathematics*, vol. 31, (1), pp. 227–238. DOI: https://doi.org/10.1016/S0168-9274(98)00129-9