Bulgarian Academy of Sciences. Space Research and Technology Institute.
Aerospace Research in Bulgaria. 32, 2020, Sofia

DOI: https://doi.org/10.3897/arb.v32.e08

STEADY FLOW ANALYSIS OF A SLENDER WING BY LIFTING SURFACE METHOD

Konstantin Metodiev

Space Research and Technology Institute – Bulgarian Academy of Sciences
e-mail: komet@space.bas.bg

Key words: *Panel Method, Lifting Surface, Iterative Scheme*

Abstract

In the paper hereby, steady flow around a thin-walled wing is analysed by means of the Lifting Surface Method. In order to carry out tests, the wing has been divided into a finite number of quadrilateral panels. All panel edges in turn are replaced by discrete straight vortex segments which induce velocities within the flow field. The problem boils down to working out velocity circulation distribution on the wing surface. For this purpose, numerical realization has been developed in C by Minimalist GNU for Windows compiler and Code::Blocks IDE. To work out a solution to the linear non-homogeneous algebraic system, the Gauss – Seidel stationary iterative method has been applied. The obtained results for various angle of attack values are depicted by means of ParaView.

Introduction

The proposed study aims at displaying a fast algorithm for three-dimensional smooth flow analysis. A straight slender wing with finite span has been chosen for the test purposes. It has an exact resemblance to HAWK-2M Unmanned aerial Vehicle (UAV) wing upper surface. The UAV is produced by Aviotehnika Ltd. in Bulgaria. Although, the utilized algorithm is relatively old (it is also widely known as the Lifting Surface Method), it is seldom used in conjunction with iterative schemes for working out a solution to the linear algebraic system. In addition, the lack of specific aerodynamic data motivates the proposed study, so does development of a nonproprietary source code.

Method

The utilized approach towards working out a numerical solution follows procedure thoroughly described in [1]. The wing is divided into finite number of quadrilateral panels. Each panel is replaced afterwards by a vortex ring consisting of four straight vortex segments with constant intensity Γ. The rings purpose is to replace the actual geometry by inducing velocities within the flow field. In this

way, discontinuities of the velocity field magnitude are introduced which is what the actual wing is designed for. Each panel centroid is considered a collocation point. The velocities induced by all panels are computed successively at each collocation point and a linear algebraic system is formed afterwards in terms of velocity circulation distribution on the wing surface. A solution to the system is worked out numerically employing the Gauss – Seidel's stationary iterative scheme. Validation of the implemented algorithm has also been carried out.

The Biot – Savart law applied to a straight vortex element

The velocity induced at point P, fig. 1, by a straight vortex element with finite length r_0 might be computed by means of following formula, [1]:

$$\mathbf{q}_{1,2} = \frac{\Gamma}{4\pi} \frac{\mathbf{r}_1 \times \mathbf{r}_2}{\left|\mathbf{r}_1 \times \mathbf{r}_2\right|^2} \cdot \mathbf{r}_0 \cdot \left(\frac{\mathbf{r}_1}{r_1} - \frac{\mathbf{r}_2}{r_2} \right) \tag{1}$$

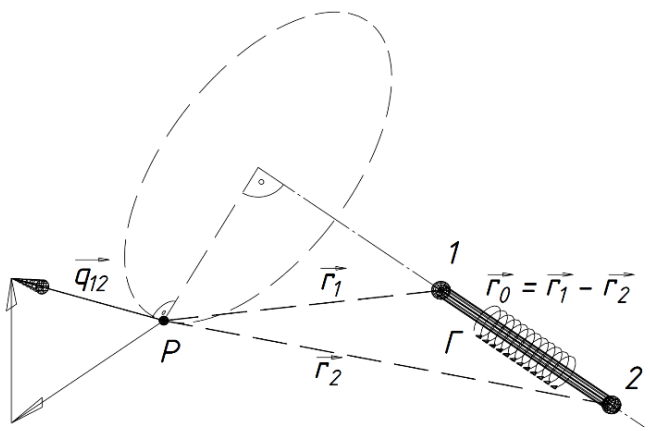

Fig. 1. Velocity induced by straight vortex element

Formula (1) is an alternative expression of the Biot-Savart law, which establishes a relationship between the induced velocity and the vortex element geometry. This formula is mainly used about further computations, which are to be made in the paper. It should also be noted that the velocity vector $\mathbf{q}_{12}$, Fig. 1, is orthogonal to the plane formed by vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ which statement intuitively follows taking into account the cross product in (1).

Computation of influence coefficients

The boundary condition imposed on the problem under consideration implies that normal flow component is not allowed through the wing surface. The

normal velocity component at each point of the wing could be divided into a self-induced and a free-stream part, [1], i.e.

$$(2) \qquad \left(\mathbf{u}+\mathbf{Q}_{\infty}\right)\mathbf{n}=0\,.$$

The self-induced part is a linear combination of the so-called influence coefficients, which must be computed at each collocation point. These coefficients are defined as velocity components normal to the surface due to singularity element (vortex ring in our case) with unit strength, [1]. What is more, the velocity components induced by a single vortex ring at given collocation point are defined as an algebraic sum of velocities induced by all straight vortex segments comprising the ring. For instance, the influence coefficient at collocation point 1 due to vortex ring *j* is defined as dot product (3) between induced velocity **u** and panel normal vector **n**,

$$(3) \qquad a_{1j}=\mathbf{u}_{1j}\,\mathbf{n}_1\,.$$

At given collocation point, velocities induced by all vortex rings that the wing contains must be added to each other. At that point, the left hand side of (2) is represented by the sum (4),

$$(4) \qquad \sum_{j=1}^{N} a_{1j}\Gamma_j=\sum_{j=1}^{N}\mathbf{u}_{1j}\mathbf{n}_1\Gamma_j=-\mathbf{Q}_{\infty}\mathbf{n}_1\,.$$

In formula (4) circulation value Γ_j is unknown. In addition, the induced velocity computed by means of (1) takes value of $\Gamma = 1$.

Having traced all vortex rings influence upon all collocation points, following linear non-homogenous algebraic system with constant coefficients is formed:

$$(5) \qquad \begin{Vmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1N} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2N} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & a_{N3} & \dots & a_{NN} \end{Vmatrix} \begin{Vmatrix} \Gamma_1 \\ \Gamma_2 \\ \Gamma_3 \\ \vdots \\ \Gamma_N \end{Vmatrix} = \begin{Vmatrix} RHS_1 \\ RHS_2 \\ RHS_3 \\ \vdots \\ RHS_N \end{Vmatrix}.$$

In system (5), the right-hand side (RHS) is computed at current collocation point as

$$(6) \qquad RHS_i=-\mathbf{Q}_{\infty}\,\mathbf{n}_i\,.$$

The approach described above is illustrated in Fig. 2 adopting double indexing notation.

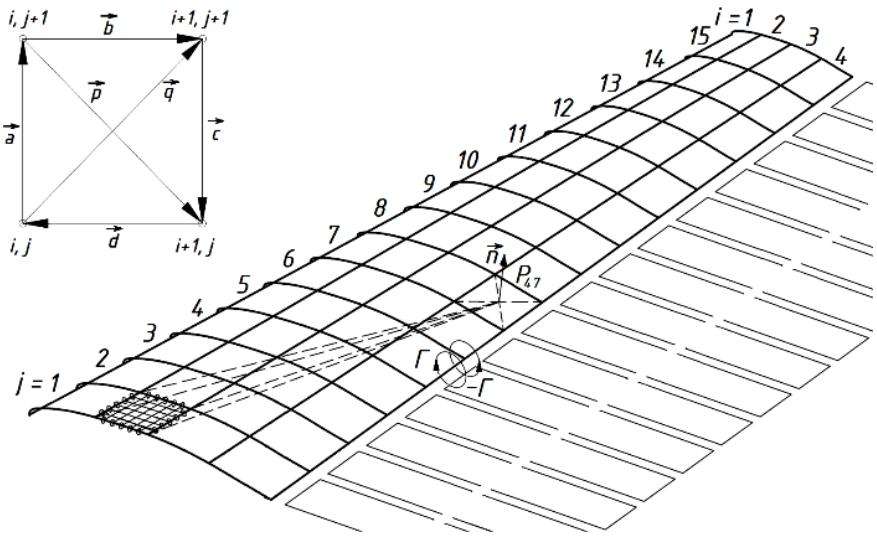

Fig. 2. Velocity induced at collocation point P47 due to panel i = 2, j = 1

In order to put the system (5) together, normal unit vectors at each panel must be computed in advance. This could be easily done recalling that the panel area equals half the cross product

$$\mathbf{S}_{ij} = 0.5(\mathbf{p} \times \mathbf{q}). \tag{7}$$

The vectors **p** and **q** form diagonals of the quadrilateral panel, Fig. 2, upper left corner. Hence, the unit vector is

$$\mathbf{n}_{ij} = \frac{\mathbf{S}_{ij}}{|\mathbf{S}_{ij}|}. \tag{8}$$

Solving a linear algebraic system

Having computed the influence coefficients, a non-homogenous linear algebraic system (5) is obtained in terms of circulation Γ distribution on the wing surface. The system is said to be strictly diagonal dominant if the absolute value of each main diagonal element is greater than sum of the absolute values of remaining entries in the current row respectively, i.e.

(9) $$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|$$

If the requirement (9) is met, then the following stationary iterative method, [2]

(10) $$x_i^k = \frac{1}{a_{ii}}\left(b_{ii} - \sum_{j=1}^{i-1} a_{ij} x_j^k - \sum_{j=i+1}^{n} a_{ij} x_j^{k-1} \right) \quad i = 1,2,\ldots,n \quad k = 1,2,3,\ldots$$

for working out a solution to the system (5) is said to converge unconditionally. Method (10) is named after Gauss and Seidel who used it as a modification of the widely known Jacobi method, [3].

(11) $$x_i^k = \frac{1}{a_{ii}}\left(b_{ii} - \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij} x_j^{k-1} \right) \quad i = 1,2,\ldots,n \quad k = 1,2,3,\ldots$$

The convergence criterion used in the algorithm is the relative difference

(12) $$\max \left| \mathbf{x}^k - \mathbf{x}^{k-1} \right| \Big/ \left| \mathbf{x}^k \right| < 10^{-3}$$

Both iterative schemes (10) and (11) require initial guess for the vector **x**.

Secondary quantities

After computing the velocity circulation distribution, it becomes possible to work out lift *L*, pressure distribution *p*, and drag due to lift *D* values. Following formulae are recommended, [1]:

- Lift on each bound vortex segment:

(13) $$\begin{aligned} \Delta L_{ij} &= \rho Q_\infty \left(\Gamma_{ij} - \Gamma_{i-1\,j} \right) \Delta y_{ij} \quad & i > 1 \\ \Delta L_{ij} &= \rho Q_\infty \Gamma_{ij} \Delta y_{ij} & i = 1 \end{aligned}$$

- Static pressure distribution:

(14) $$\Delta p_{ij} = \Delta L_{ij} \Big/ \Delta S_{ij}$$

- Induced drag due to trailing vortex segments of each panel:

$$\Delta D_{ij} = -\rho w_{ij}\left(\Gamma_{ij} - \Gamma_{i-1\,j}\right)\Delta y_{ij} \quad i > 1$$
$$\Delta D_{ij} = \rho w_{ij}\Gamma_{ij}\Delta y_{ij} \quad i = 1 \tag{15}$$

where w is the induced velocity on the wing surface. Eventually, for total values it yields:

- Total drag:

$$D = \sum_{i=1}^{M}\sum_{j=1}^{N}\Delta D_{ij} \tag{16}$$

- Total lift:

$$L = \sum_{i=1}^{M}\sum_{j=1}^{N}\Delta L_{ij} \tag{17}$$

Since the fluid is assumed ideal, any other kind of drag force is expected to be zero due to d'Alembert paradox.

Source code description and validation

The source code has been developed in C. It utilizes two main structures shown in Fig. 3. Firstly, a structure storing quantities due to one straight vortex segment is created. In addition to geometry and velocity data, it contains a pointer to a function working out the induced velocities $\mathbf{q}_{12}$ according to formula (1). Then, another structure is created to store quantities related to one vortex ring. Since one ring contains four segments, four pointers to the former structure are declared within the latter. In this way, an inheritance is implemented facilitating code development process and making it possible to clarify vortex segments pertaining to the current ring and their location on the mesh. According to author's experience, double indexing of the panels in both curvilinear directions is the best approach as it is shown in Fig. 2.

The source code reads a text file containing grid point coordinates. Having read the data, the code passes them to aforementioned structures for further processing. All data blocks such as structures and arrays are dynamically allocated and assigned to pointers thereafter, which lets developer make use of the program with arbitrary number of panels. Both the number of wake panels and their geometry are computed automatically. Having completed the calculations, all data blocks are set free.

The source code has been developed by means of Minimalist GNU for Windows v.4.9.2, [4] and Code::Blocks IDE v.17.12, [5]. The 3rd party software used is ParaView to visualize obtained data, [6].

```
#ifndef DEFS_H
#define DEFS_H
#define PI_180 4. * atan(1.) / 180.
#define I 10
#define J 32
#define AOA 10 //deg
#define IW 2   // wake points along I

typedef double real;
typedef int int_t;

typedef struct oneSegment {
        struct oneSegment *that;
        real u, v, w, x1, y1, z1, x2, y2, z2;
        int_t (*q12)(struct oneSegment*, real, real, real, real);
} mySegment;

typedef struct onePanel {
        struct onePanel *that;
        real S, nx, ny, nz;
        mySegment *East, *West, *North, *South;
} myPanel;

real **x, **y, **z, **xP, **yP, **zP, **xW, **yW, **zW, *A, *b, **Gamma;
myPanel ***panel, ***panelW;

real** make2DArray(int_t X, int_t Y);
int_t delete2DArray(real **foo, int_t X);
int_t q12Common(mySegment *foo, real xP, real yP, real zP, real G);
myPanel* makePanel(int_t i, int_t j, real **x, real **y, real **z);
int_t killPanel(myPanel *foo);
int_t LHS(real *pA, real **G, int_t knob);
int_t RHS(real *pb);
int_t solveLS_GS(real *a, real *b, real **G);

#endif // DEFS_H
```

Fig. 3. Main header file used in the developed software

In order to estimate the program ability to work out a solution in advance, a few validation cases were carried out. Firstly, the iterative scheme (10) was tested with exact solution of small-sized system of linear non-homogenous equations, precisely 3 and 4. The Gauss-Seidel method proved to be about three times faster than Jacobi's, as expected.

The proposed algorithm is tested further by means of a thin rectangular wing divided into a mesh of 4 times 26 panels. This problem has been solved in [1] by means of a FORTRAN code published in Appendix D.2. Unlike the presented study, Gauss eliminations were used in [1] to solve the linear system of equations.

Both results for circulation Γ distribution on the wing surface are shown in Fig. 4 and Table 1.

Numerical results

In Table 1 and Fig. 4, results from presented code validation are shown. In Fig. 4, the wing semi span is solely depicted due to symmetry.

Table 1. Exemplary serial computations

	Gamma, m^2/s, Katz and Plotkin, [1]				Gamma, m^2/s, presented code			
R/C	1	2	3	4	1	2	3	4
1	0.491	0.699	0.822	0.889	0.491	0.698	0.822	0.889
2	0.490	0.697	0.820	0.887	0.490	0.697	0.819	0.887
3	0.487	0.693	0.815	0.882	0.488	0.693	0.815	0.882
4	0.484	0.688	0.808	0.875	0.484	0.688	0.808	0.874
5	0.479	0.680	0.799	0.864	0.479	0.680	0.799	0.864
6	0.472	0.670	0.786	0.850	0.472	0.670	0.786	0.850
7	0.463	0.656	0.769	0.830	0.463	0.656	0.769	0.830
8	0.451	0.637	0.746	0.805	0.451	0.637	0.746	0.805
9	0.435	0.613	0.715	0.771	0.435	0.613	0.715	0.770
10	0.413	0.579	0.674	0.724	0.413	0.579	0.674	0.724
11	0.383	0.532	0.615	0.659	0.383	0.532	0.615	0.658
12	0.337	0.460	0.526	0.561	0.337	0.460	0.526	0.561
13	0.255	0.336	0.378	0.400	0.255	0.336	0.378	0.400

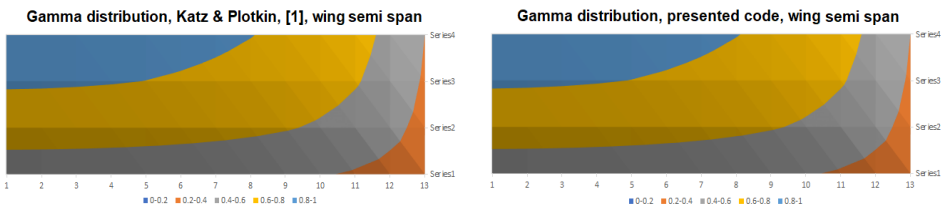

Fig. 4. Program validation, α = 5 deg, Q_∞ = 1 m/s

In Fig. 5, numerical results are shown for lift force distribution along upper surface of the Hawk-2M wing. In this case, the Gauss-Seidel numerical routine (10) has been used to work out a solution to system (5). In case of angle of attack α = 10 deg, Q_∞ = 16.7 m/s, ρ = 1.225 kg/m^3, the total lift is L = 147 N. In addition, the total lift coefficient is C_L = 1.358. In Fig. 6, a screenshot taken during numerical computations is depicted regarding same computational case. In the figure, upper left corner of the coefficient matrix taking part in system (5) is clearly visible. It is evident that the main diagonal elements are dominant with respect to

the order of magnitude. In Fig. 7, the root wing foil of Hawk-2M UAV is depicted. The wingfoil upper surface has been extruded along the semi span so as to obtain the 3D slender wing used in the present study.

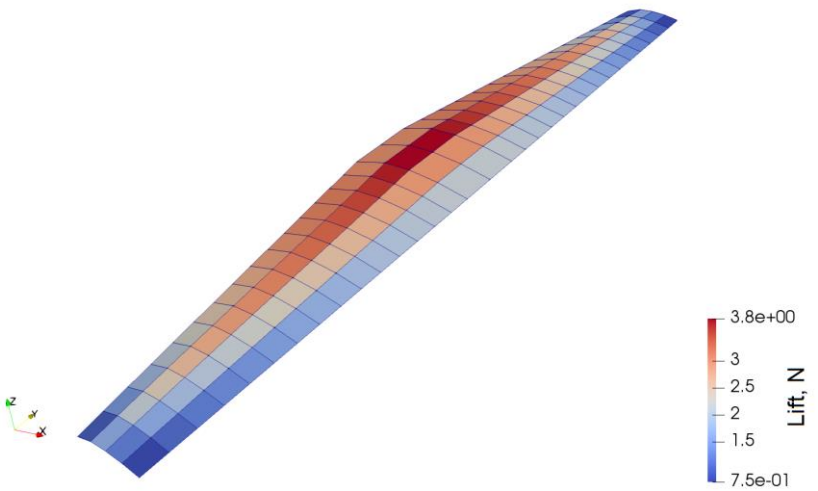

Fig. 5. Lift force distribution, N

"C:\Users\Prokopy\Desktop\Desktop\Lifting Surface Method\lift3DH3D\bin\Debug\lift3DH3D.exe"

-1.126e+001	8.969e-001	8.536e-002	2.545e-002	1.117e-002	5.985e-003
8.659e-001	-1.069e+001	9.490e-001	9.085e-002	2.698e-002	1.179e-002
7.532e-002	9.197e-001	-1.023e+001	9.978e-001	9.618e-002	2.845e-002
2.089e-002	8.111e-002	9.698e-001	-9.840e+000	1.044e+000	1.013e-001
8.606e-003	2.254e-002	8.669e-002	1.018e+000	-9.515e+000	1.086e+000
4.358e-003	9.292e-003	2.413e-002	9.217e-002	1.061e+000	-9.238e+000
2.506e-003	4.706e-003	9.953e-003	2.570e-002	9.740e-002	1.102e+000
1.572e-003	2.706e-003	5.042e-003	1.061e-002	2.721e-002	1.025e-001
1.050e-003	1.698e-003	2.900e-003	5.375e-003	1.124e-002	2.869e-002
7.362e-004	1.134e-003	1.819e-003	3.092e-003	5.697e-003	1.186e-002
5.359e-004	7.951e-004	1.216e-003	1.940e-003	3.279e-003	6.013e-003
4.022e-004	5.789e-004	8.524e-004	1.297e-003	2.058e-003	3.461e-003
3.096e-004	4.345e-004	6.206e-004	9.093e-004	1.375e-003	2.173e-003
2.434e-004	3.345e-004	4.659e-004	6.622e-004	9.647e-004	1.453e-003
1.948e-004	2.630e-004	3.588e-004	4.972e-004	7.026e-004	1.019e-003
1.510e-004	2.001e-004	2.671e-004	3.608e-004	4.945e-004	6.912e-004
1.145e-004	1.489e-004	1.945e-004	2.564e-004	3.415e-004	4.614e-004
9.251e-005	1.187e-004	1.528e-004	1.981e-004	2.587e-004	3.418e-004
7.902e-005	1.005e-004	1.280e-004	1.640e-004	2.114e-004	2.751e-004
6.800e-005	8.576e-005	1.083e-004	1.372e-004	1.749e-004	2.244e-004
5.891e-005	7.374e-005	9.231e-005	1.159e-004	1.462e-004	1.854e-004
5.135e-005	6.384e-005	7.932e-005	9.879e-005	1.234e-004	1.548e-004
4.503e-005	5.563e-005	6.864e-005	8.484e-005	1.051e-004	1.306e-004

Fig. 6. Matrix A upper left corner

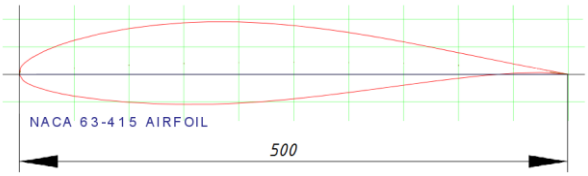

Fig. 7. NACA 63-415, 15%

4. Discussion

It is somewhat appropriate to note that velocities induced at given collocation point by an arbitrary vortex ring are highly dependent of the distances $\mathbf{r}_1$ and $\mathbf{r}_2$, Fig. 1, dashed lines, Fig. 2. The longer the distance the least the influence. Therefore, the biggest velocities are induced by a ring, which overlaps the collocation point. This in turn implies that the user is expected to get a diagonally dominant system of linear equations because diagonal elements with same indices all in (5) are obtained whenever the collocation point and vortex ring coincide.

In Fig. 2, a set of vortex rings (wake) is depicted right behind the wing trailing edge. Essentially, this implements the Kutta condition implying that the rear stagnation point should be held at the trailing edge. At that point, the flow velocity takes zero value. The vortex straight segments in Fig. 2 along the trailing edge replace the initial vortex. They would remain uncompensated if the vortex wake were absent. It is well known fact that the velocity vector circulation makes up for the initial vortex to preserve angular momentum of the mechanical system. Having encountered the trailing edge panels, the algorithm appends wake rings influences in order to impose the Kutta condition. For all that, the wake is a major source of numerical errors, which is why it is recommended for its length to be no less than 20 chords, [1].

In Table 1, Fig. 4, slight differences might be observed between results. The explanation might be found the digit precision. In book [1], single precision variables were used whilst in presented study results are quoted using double digit precision. In Fig. 5, the numerical results are shown for angle of attack $\alpha = 10$ *deg*. It may be figured out that the lift force decreases dramatically in the vicinity of ailerons thus reducing their efficiency. This is notably the case of take-off and landing. Efficiency of Hawk-2M aileron has been already thoroughly discussed in paper [7].

According to Fig. 6, the coefficient matrix is expected to be noninvertible. The matrix determinant has been computed in addition by means of GNU Scientific Library, [8], “gsl_linalg_LU_det” routine. The obtained order of result was quite small which makes the iterative scheme preferable to some direct methods such as the Cramer’s rule.

A possible algorithm extension would be simulating an unsteady flow around the wing. In this case, the wake shape evolution should be computed in addition. Also, it is advisable to study the topic of winglets design for this particular wing. Unfortunately, it is not possible to dive into all details of the source code due to the limited paper size. However, the source code is extensively commented and distributed by the author on demand.

Some additional tests carried out by means of Jacobi's and Gauss-Seidel's methods might be found in [9].

References

1. Katz, J. and A. Plotkin, Low-Speed Aerodynamics, Cambridge University Press, NY, 2010.
2. Fadugba, S. E., On Some Iterative Methods for Solving Systems of Linear Equations, Computational and Applied Mathematics Journal, 2015, 1, 2, 21–28. AASCIT
3. Larson, R., Elementary Linear Algebra, 7th edition, Brooks/Cole, CENGAGE Learning, MA, 2013.
4. MinGW. http://www.mingw.org/ (Accessed on 05 November 2020)
5. Codeblocks. http://www.codeblocks.org/ (Accessed on 05 November 2020)
6. Paraview. https://www.paraview.org/ (Accessed on 05 November 2020)
7. Metodiev, K., A New Aileron Designed for Hawk–2M Unmanned Aerial Vehicle, Journal of the Technical University–Sofia, Plovdiv branch, "Fundamental Sciences and Applications", 2015, 21.
8. GSL - GNU Scientific Library. https://www.gnu.org/software/gsl/ (Accessed on 05.11.2020)
9. Solution of Linear Equations by Stationary Iterative Methods. http://www.space.bas.bg /acsu/Ax_b/ (Accessed on 05 November 2020)

АНАЛИЗ НА СТАЦИОНАРНО ТЕЧЕНИЕ ОКОЛО ТЪНКО КРИЛО ПО МЕТОДА НА НОСЕЩАТА ПОВЪРХНОСТ

К. Методиев

Резюме

В настоящата статия стационарно течение около тънко крило е анализирано по Метода на носещата повърхност. За да се проведат числени експерименти, крилото бе разделено на краен брой квадратични панели. Всички страни на панелите на свой ред са заменени от дискретни прави вихрови сегменти, които индуцират скорости в полето на течението. Задачата се свежда до намиране на разпределението на циркулацията по повърхността на крилото. За целта бе разработена числена реализация на език С посредством компилатор Minimalist GNU for Windows и развойна среда Code::Blocks. За да се намери решение на линейната нехомогенна алгебрична система бе приложен итеративен метод на Гаус – Зайдел. Получените резултати за различни ъгли на атака са визуализирани посредством ParaView.