Nationaal Lucht- en Ruimtevaartlaboratorium
National Aerospace Laboratory NLR

provisional issue

NLR-TP-2002-

The B2000 Doublet Lattice Processor: B2DL

P. Arendsen

This report is based on a presentation held on 4th biannual B2000 Workshop, at Ligerz Switzerland on 24/25 October 2002.

Customer:	National Aerospace Laboratory NLR
Working Plan number:	
Owner:	National Aerospace Laboratory NLR
Division:	Structures and Materials
Distribution:	Unlimited
Classification title:	Unclassified
	October 2002

Summary

A method called “Doublet Lattice” is described for calculating the aerodynamic loading on infinitely thin, harmonically oscillating airfoils in subsonic flow. The generalised aerodynamic force coefficients are also determined from the pressure distributions. The method is flexible and fast, but relatively simple. It can be applied to a wide range of aircraft configurations. Albano, Rodden, Giesling and Kalman have developed the theory around 1965. At NLR a computer program based in this theory has been developed around 1974 by Bennekers and Labrujere, which forms the basis of the B2000 Doublet Lattice processor B2DL.
This old code has been ported to the B2000 environment, such that memory management, disk usage and pre- and postprocessing capabilities are greatly improved.

Contents

(24 pages in total)

List of Symbols

C_l	Steady lift coefficient (equation 9)
C_m	Steady moment coefficient (equation 10)
ΔC_p	Time independent amplitude of the pressure jump across the lifting surface divided by the dynamic pressure q of the undisturbed flow.
ΔC_p^j	ΔC_p on panel j
$\Delta C_p^{(j)}$	ΔC_p going with deflection mode j
D_{ij}	i, j-th element of the matrix of total downwash factors (defined by equation 7)
$f^{(i)}$	Deflection going with mode i
$f_k^{(i)}$	Deflection in ¼-chord point going with mode i of panel k
h	Deflection mode
k	Reduced frequency, $k = \omega l / U$
$k(y)$	Lift coefficient in section y (defined by equation 9)
K	Kernel function
l	Reference length
$m(y)$	Moment coefficient in section y (defined by equation 10)
M	Mach number
M_j	Local Mach number at panel j
M_∞	Freestream Mach number
$n(y)$	Hinge moment coefficient in section y (defined by equation 12)
O	Total wing area
O_k	Area of panel k
$p(y)$	Local control surface lift coefficient in section y (equation defined by 11)
$\overline{p}$	Column vector of ΔC_p
q	Dynamic pressure: $q = \frac{1}{2}\rho U^2$
Q_{ij}	Generalised force coefficients with deflection mode i and pressure mode j (defined by equation 13)
t	Time
U	Freestream velocity
w	Normalwash, time independent amplitude of velocity, normal to the surface, divided by the freestream velocity U
$\overline{w}$	Column vector of w's
W_j	Scaling factor of panel j
Δx_j	Local panel chord of panel j
x	Cartesian x-co-ordinate
x_h	Cartesian x-co-ordinate of hinge line

x_l	Cartesian x-co-ordinate of leading edge
x_t	Cartesian x-co-ordinate of trailing edge
$x_{c/4}$	$x_l + (x_t - x_l)/4$
y	Cartesian y-co-ordinate
z	Cartesian z-co-ordinate
λ_j	Sweep angle of ¼-chord line of panel j
μ	Integration variable along ¼-chord line of a panel
ρ	Atmospheric density
ω	Circular frequency

1 Introduction

The Doublet Lattice (DL) method to calculate unsteady aerodynamic loading is a relatively fast, but simple, method compared to more modern methods like unsteady wing body panel methods of even Navier-Stokes analyses. Its applicability lies in the fast calculation of aerodynamic loading in early phases of the design of an aircraft. In those phases the accuracy of the geometry, mass distributions and stiffness distributions is such that the use of Doublet Lattice methods is justified.

Albano, Rodden, Giesling and Kalman have developed the Doublet Lattice theory around 1965 [1],[2],[3],[4]. The method divides the lifting surface in a number of panels. On each panel the force is generated by lifting line elements along the ¼ chord line. The pressure is obtained by dividing the lifting force through the panel area. In this method any planform can be represented by choosing an appropriate panel distribution. As the vibration and/or displacement modes are prescribed only in a number of discrete points, any mode can be treated without causing a modification of the computer program, as would be the case when using kernel functions [5].

At NLR a computer program based in the Doublet Lattice theory has been developed around 1974 by Bennekers and Labrujere [6],[7], which forms the basis of the B2000 Doublet Lattice processor B2DL. This old code, called VARDOB, has been ported to the B2000 environment, such that memory management, disk usage and pre- and postprocessing capabilities are greatly improved.

2 Theoretical aspects about the Doublet Lattice method

The DL-method is a lifting element method in which the infinitely thin lifting surfaces are divided into trapezoidal elements (panels). These elements are arranged in strips aligned with the direction of freestream (figure 1). Each element contains a distribution of acceleration potential doublets, which is equivalent to a pressure jump across the surface. Each potential (pressure jump) is of oscillating yet unknown strength, concentrated at its ¼-chord line (lifting line). In addition each element possesses a control point (collocation point) in the middle of its ¾-chord line. The normalwash introduced by all lifting lines is summed for each control point. Equalising this to a prescribed normalwash, as derived from the oscillatory behaviour of the lifting surface, leads to a set of algebraic equations. From these equations the strength of the lifting line and thus the pressure jump across the surface can be computed. Integration over the surface gives local and total aerodynamic force coefficients.

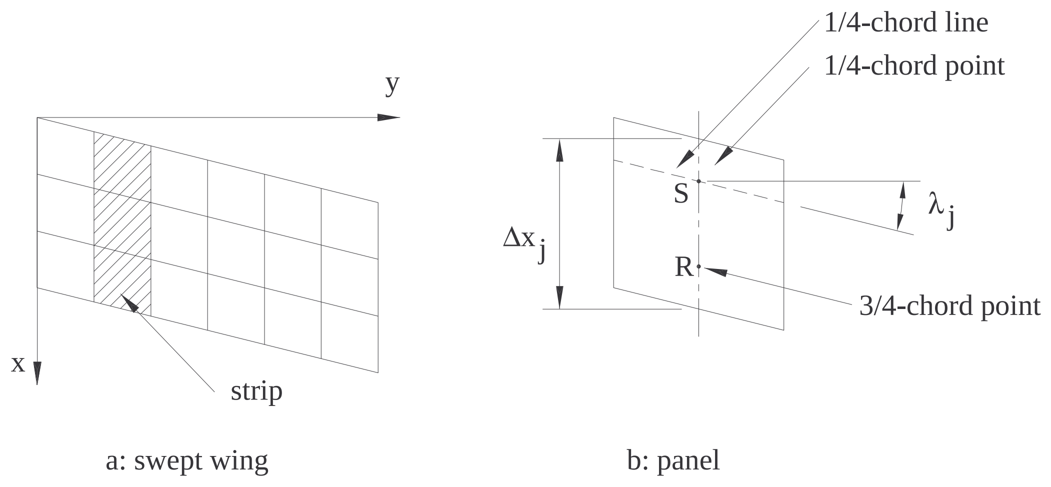

Figure 1: Surface representation by quadrilateral elements

In linearised theory an integral expression may be derived [1], which relates the velocity normal to the surface of the configuration to the pressure difference across the surface:

$$w(x,y,z) = \frac{1}{8\pi}\iint \Delta C_p(\xi,\delta).K(x,y,z,k,M;\xi,\delta)\partial\xi\partial\delta \tag{1}$$

with

$$\overline{w}(x,y,z,t) = \mathrm{Re}\left[w(x,y,z).e^{i\omega.t}\right] \tag{2}$$

and

$$\overline{p}(x,y,z,t) = \frac{1}{2}\rho U^2.\mathrm{Re}\left[\Delta C_p(x,y,z).e^{i\omega.t}\right] \tag{3}$$

The linearised condition of tangential flow may be expressed as:

$$w(x,y,z) = \frac{d}{dx}h(x,y,z) + i\frac{k}{l}h(x,y,z) \tag{4}$$

Here *h(x,y,z)* is the deflection mode of the surface measured normal to the surface. By prescribing *h(x,y,z)* and equating (1) and (4) an integral equation for the unknown pressure amplitude ΔC_p is established.
In case of steady flow the lifting configuration and its wake can be represented equally well by a sheet of velocity potential doublets. The pressure difference is then proportional to the streamwise derivative of the doublet strength. In the present method the doublet strength is

piecewise constant, as a consequence of which the doublet sheet may be replaced by a set of horseshoe vortices and the pressure differences become proportional to the strength of these vortices.
Thus the problem of determining the function ΔC_p, for both the steady and unsteady case, is reduced to the problem of calculating a finite number of constants representing the strengths of the doublet lines / horseshoe vortices. By choosing as collocation points one so called receiving point on each panel located midway between strip edges on the ¾-chord line, a system of linear algebraic equations is obtained.

Replacing the doublet sheet by a system of doublet lines and panels, equation (1) reduces to

$$w(x,y,z) = \sum_{j=1}^{n} \frac{1}{8\pi} .\Delta C_p^j .\Delta x^j .\cos\lambda^j .\int_0^{l_j} K(x_i, y_i, z_i, k, M; \mu) \partial\mu \qquad (5)$$

where μ is the integration variable along the doublet line located at the j-th panel and where the kernel function K is given according to Landahl [7] In the work of Bennekers and Labrujere [1],[2] a detailed formulation corresponding to the code implementation is given..

$$w(x,y,z) = \sum_{j=1}^{n} \frac{1}{8\pi} .\Delta C_p^j .\Delta x^j .\cos\lambda^j .\int_0^{l_j} K(x_i, y_i, z_i, k, M; \mu) \partial\mu \qquad (6)$$

Let

$$D_{ij} = \frac{\Delta x^j .\cos\lambda^j}{8\pi} \int_0^{l_j} K(x_i, y_i, z_i, k, M; \mu) \partial\mu . \qquad (7)$$

then the set of linear equations becomes:

$$w(x_i, y_i, z_i) = \sum_{j=1}^{n} D_{ij} .\Delta C_p^j \qquad (8)$$

Notice that a solver for un-symmetric complex full matrices is needed.

From the calculated pressure coefficients the force and moment coefficients over every strip are computed with the aid of the following formulae:

Lift coefficient

$$k(y)=\frac{\int_{x_l}^{x_t}\Delta C_p.dx}{\pi(x_t-x_l)}\approx\frac{\sum_j \Delta C_p^j.\Delta x_j}{\pi(x_t-x\ )} \tag{9}$$

Moment coefficient

$$m(y)=2\frac{\int_{x_l}^{x_t}\Delta C_p.(x-x_{c/4}).dx}{\pi(x_t-x_l)^2}\approx 2\frac{\sum_j \Delta C_p^j.\Delta x_j.(x_j-x_{c/4})}{\pi(x_t-x_l)^2} \tag{10}$$

Control surface lift coefficient

$$r(y)=\frac{\int_{x_h}^{x_t}\Delta C_p.dx}{\pi(x_t-x_l)}\approx\frac{\sum_k \Delta C_p^k.\Delta x_k}{\pi(x_t-x_l)} \tag{11}$$

Hinge moment coefficient

$$n(y)=2\frac{\int_{x_l}^{x_t}\Delta C_p.(x-x_{c/4}).dx}{\pi(x_t-x_l)^2}\approx 2\frac{\sum_k \Delta C_p^k.\Delta x_k.(x_k-x_h)}{\pi(x_t-x_l)^2} \tag{12}$$

To determine the generalised aerodynamic force coefficients Q_{ij} the expression below was used, where (j) and (i) are the numbers if the deflection modes.

$$Q_{ij}=-\frac{1}{ls^2}\iint_O \Delta C_p^{(j)}.f^{(i)}.dxdy\approx-\frac{1}{ls^2}\sum_{k=1}^{n}\Delta C_p^{(j)}.W_j.f_k^{(i)}.O_k \tag{13}$$

Multiplication of ΔC_p with a scaling factor W_j is optional , as is the local Mach number correction on the effective downwash.

$$w_j=\frac{M_j}{M_\infty}\frac{dh_j}{dx}+i\frac{k}{l}h_j \tag{14}$$

3 Modern computing aspects

The B2DL processor is basically a port of the existing FORTRAN program VARDOB, written in the early seventies [7]. At that time memory was limited and disk space was used as additional scratch memory. Within the FORTRAN standard memory could only be allocated statically, which implies that a programmer had to reuse already declared arrays and vectors to minimise the memory footprint. For the same reason solvers decomposed the problem into small chunks to be held in core, while a lot of disk input/output was done to switch all the chunks and the partial results in and out of memory.
Since modern computers have multiple megabytes of memory, the usage of scratch files to store small chunks of decomposed data is outdated. The entire Doublet Lattice based aerodynamic calculation fits easily within the memory of modern computers. Within the B2000 environment dynamic memory allocation in FORTRAN is possible, which implies that the usage of predefined memory chunks (common block) is not longer necessary. One allocates memory when needed and frees it again when the job is done. This enables a clearer and more maintainable implementation of the analysis code. Since the size of the memory which is allocated is determined within the runtime of the program, predefined memory chunks are no longer a necessity. Hence, limits on the maximum number of panels or the maximum number of modes within an analysis are lifted within the B2DL implementation. Within the B2DL implementation there are no scratch files. All resulting data and input data is stored in a central data base. The solver in B2000, which calculates the set of algebraic equation (4), is a modern in-core solver.

4 B2000 Overview

B2000 has emerged from the need for a modular, hardware-independent, user-customised numerical analysis system, which corresponds to modern software standards [8]. Although primarily designed for Finite Element computations, B2000 also encompasses other numerical methods. B2000 consists of a series of physically independent program modules (processors), each of which performs a logically defined task. Data transfer between modules occurs only through the data base. Processors may be executed separately or in a cluster. The modular design of B2000 enables the user to select the necessary processors for solving a specific problem. New independent processors may be added to B2000. This feature is particularly suited for users wishing to develop their own methods. The most important requisite of a modular system is a well-defined and consistent data definition [9]. This implies:
- Self-descriptive and transparent data throughout all modules
- Data transfer between modules exclusively via the global data base
- External data bases and additional files are only known to a single processor.

4.1 Solution techniques

B2000 is designed to deal with a whole range of problems. Substructuring, partial element assembling and penalty functions to link degrees of freedom are integral parts of B2000. The LU equation solver can handle non-positive definite problems. There is no a priori limitation to the number of degrees of freedom. Eigenvalue solvers (Simultaneous inverse vector iteration and Lanczos iteration) are available for the solution of the generalised eigenvalue problem. Standard modules for the solution of non-linear problems (Riks method, modified Newton method) are provided. Special emphasis has been put on coordinate transformations. Facilities are provided for transformations from global coordinates to substructure-local coordinates and from node-local (surface) coordinates to global coordinates.

4.2 Input and Output

Input to B2000 may be directed to the Input Processor [10]. Alternatively the input data may be set up directly in the global data base format. All B2000 output data may be accessed through the global data base or by the graphics processor. The data may of course be translated to fit any other pre- and post-processing system. The modular design of B2000 and its integration in the MEM-COM data base management system [11] are predestinate to a distributed processing environment, in which (for instance) input processing, computations and visualisation are performed on different computers. The B2000 documentation, like the data base description or the processor user manuals, may be accessed on-line.
An additional B2000 input processor (B2DLIP) has been created and additional Aero Model Definition Language (AMDL) commands have been implemented [11]. This results in a user friendly definition capability of the analysis task. The post-processor BASPL++ can be used to graphically show the geometry and the results, while data viewers like MONITOR (text based) , MCBROWSER (GUI based) can be used to inspect numerical data within the data base [12].

4.3 Architecture of B2000

B2000 consists of independent processors connected to a common data base. All information necessary for a run is contained on the common data base. Processors consist of kernels, which call the subroutines needed for performing the logical functions required by the processors. The kernel itself is a FORTRAN/C/C++ subroutine, which must be called either by a main program or a main subroutine. Main programs for all processors are included in B2000. Processors may also be clustered, by linking them into a common executable code. Note that the individual processors are available both as stand-alone programs and as subroutines tied together by a main program. Processors are supervised by the following techniques:
- Running each processor in interactive or in batch mode.
- Executing B2000 by linking processors into a "Main Program".
- Procedures, written by the user.

5 Doublet Lattice Processor B2DL

The input for a Doublet Lattice analysis job is translated to database with the B2DLIP processor. The analysis itself is performed by the B2DL processor [11],[12].

5.1 Aero Model Definition Language

There are “General Commands” and “Branch Commands” within the Aero Model Definition Language. The “General Commands” are:

title	Define a problem title
branch	Initialises the branch definition
dlctrl	Define Doublet Lattice analysis control parameter
ias	Specify symmetry of modes
struc	Define list of nodal points of structural model

The “Branch Commands” are:

patch	Generate 4-sided flat mesh patches
mach	Specify local Mach number correction
wmat	Specify weighting factors
rud	Specify rudder hinge co-ordinates
tab	Specify tab hinge co-ordinates
wash	Specify nodal wash

Detailed information on these commands is given in [11] and [12]. An example is given in appendix A.

Notice that the deflection modes can be given within the B2DL input deck, OR, by defining a **structural** model using the regular input processor B2IP and calculating displacement or vibration modes. These modes can be transferred to the **aero** model using a simple surface spline technique. The structural nodes used within this surface spline based mapping are given by the *struc* command.

5.2 DL related data sets

In the data set listed below the following abbreviations are used

br	branch number
cyc	optimisation cycle number
br	reduced frequency number
mode	deflection number
k	reduced frequency number

The listing below refers only to DL related set, including future developments as mentioned in chapter 7.

AEROCOEF.*br.cyc.0.k*	Aerodynamic coefficients as calculated per strip
CHBYCHV.*cyc*	Chebichev interpolation matrix to extrapolate aerodynamic loading as function of the reduced frequency
DIS.*br.cyc.0.k*	Deflection of ¼-chord points of all panels in a branch
DIS.TOTAL.*cyc.0.k*	Deflection of ¼-chord points of all panels
DLCTRL	Doublet Lattice control parameters
GENFOR.*k.cyc*	Generalised forces as calculated by B2DL
GKA.TOTAL.*cyc*	Matrix which stores the data about mesh transformation from structural mesh to aero mesh
IAS.COEF	Symmetry flags indicating whether a move is (anti-) symmetrical
PRES.*br.cyc,k,mode*	Calculated pressure of all panels in a branch
PRES.TOTAL.*cyc.k.mode*	Calculated pressure of all panels
RDIS.*br.cyc.k.mode*	Deflection of ¾-chord points of all panels in a branch
RDIS.TOTAL.*cyc.0.k*	Deflection of ¾-chord points of all panels
REDFRQ.*cyc*	Set of reduced frequencies for which calculations must be made
ROER1.*cyc*	Co-ordinates of the rudder.
TAB1.*cyc*	Co-ordinates of the tab.
SNLST	List of structural nodes used in translating structural deflections to aero deflections and vice versa.
WASHFAC.*k.cyc*	The matrix with the washfactors, see formula (8)

6 Benchmarks

In order to validate B2DL benchmark cases as defined for the old VARDOB [7] code are used to compare B2DL results with VARDOB results. In [7] two configurations are used:

- A constant swept wing with a full flap (figure 2)
- A T-tail configuration of fin and stabiliser (figure 3)

These two benchmarks give **a 100% equivalent** set of results between B2DL and VARDOB. Since VARDOB is a proven technology program [6][7], comparing B2DL and VARDOB by listing identical numerical results, is meaningless. In this report a short graphical overview of the benchmarks is therefore given.

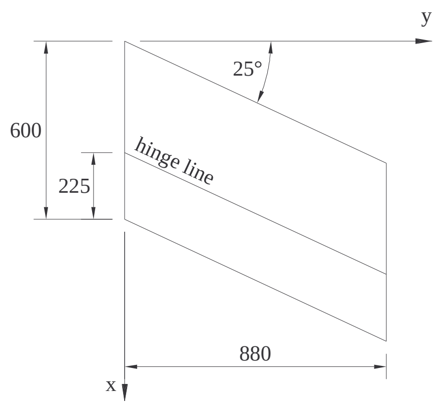

Figure 2: Swept wing planform definition

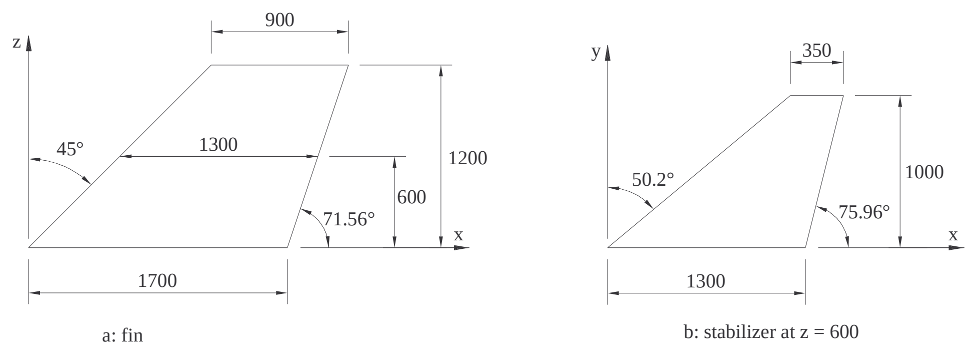

Figure 3: T-Tail planform definition

The benchmarks are incorporated within the test suite of B2000. For numerical details the reader is referred to this test suite or [6][7].

6.1 Steady flow - Swept wing

In the steady case a constant downwash has been prescribed, without flap deflection. Since in B2DL the deflection modes are given in terms of nodal displacements, the constant downwash is translated to a displacement. See formula (4).

$$h(x,y,z) = x \tag{15}$$

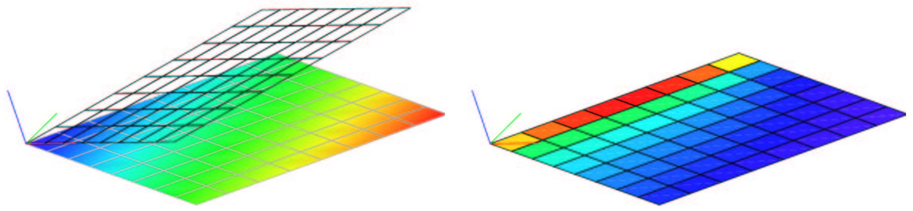

Figure 4: Displacement of Swept Wing in Steady flow (left)
Pressure distribution (Real part) of Swept Wing in Steady flow (right)

6.2 Steady flow - T-tail

In the steady case two different cases of normalwash have been prescribed:

$$\text{Case 1:}\quad \begin{cases} h(x,y,0.6) = x \\ h(x,0.0,z) = 0 \end{cases} \tag{16a}$$

$$\text{Case 2:}\quad \begin{cases} h(x,y,0.6) = 0 \\ h(x,0.0,z) = x \end{cases} \tag{16b}$$

Case 1 reflects a non-zero angle of incidence of the stabiliser and a zero angle of incidence of the fin. Case 2 reflects a non-zero angle of incidence of the fin and a non-zero angle of incidence of the fin.

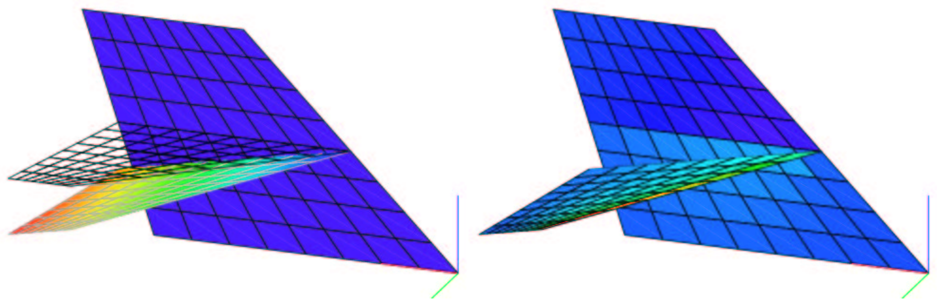

Figure 5a: Displacement (case 1) of T-Tail in Steady flow (left)
Pressure distribution (Real part, case 1) of T-Tail in Steady flow (right)

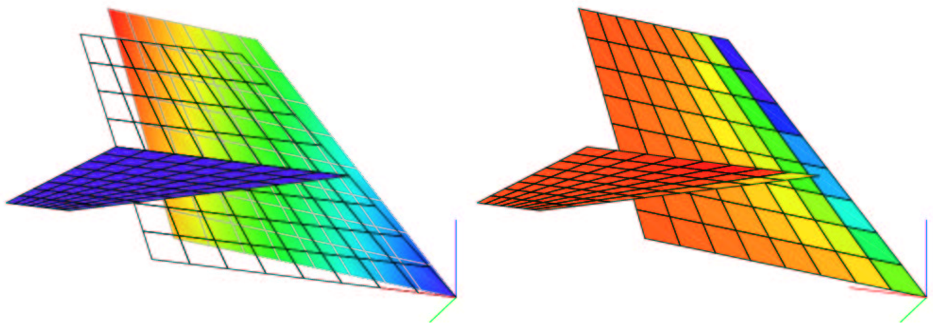

Figure 5b: Displacement (case 2) of T-Tail in Steady flow (left)
Pressure distribution (Real part, case 2) of T-Tail in Steady flow (right)

6.3 Un-steady flow - Swept wing

In the unsteady case the following deflection mode has been chosen:

$$h(x,y,z) = \begin{cases} 0 & x < x_h \\ x - x_h & x \geq x_h \end{cases} \tag{17}$$

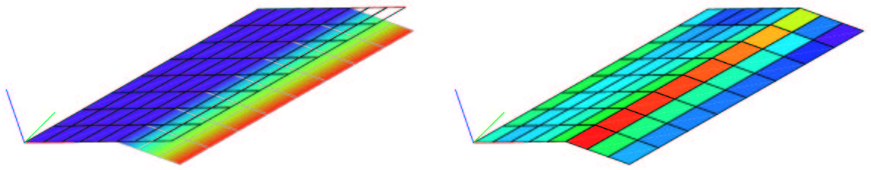

Figure 6: Displacement of Swept Wing in Un-steady flow (left)
Pressure distribution (Real part) of Swept Wing in Un-steady flow (right)

6.4 Un-steady flow - T-tail

Three different deflection modes have been considered for the T-tail:

Fin twist: $$h_1(x,y,z) = \begin{cases} h_{fin} = z.(x - 0.875.z - 3) \\ h_{stab} = 0 \end{cases} \tag{18a}$$

Fin bending: $$h_2(x,y,z) = \begin{cases} h_{fin} = z^2 \\ h_{stab} = 0 \end{cases} \tag{18b}$$

Stabiliser roll: $$h_3(x,y,z) = \begin{cases} h_{fin} = 0 \\ h_{stab} = y \end{cases} \tag{18c}$$

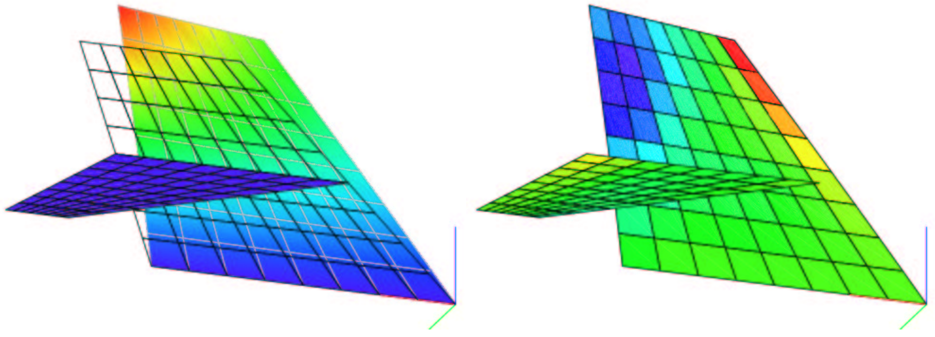

Figure 7a: Displacement (fin twist) of T-Tail in Un-steady flow (left)
Pressure distribution (Real part, fin twist) of T-Tail in Un-steady flow (right)

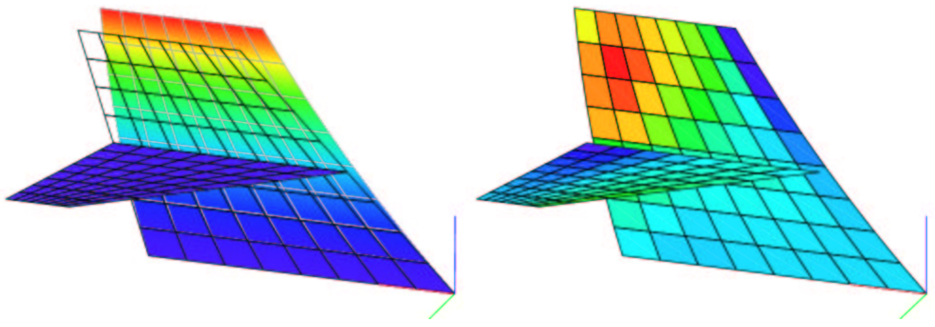

Figure 7b: Displacement (fin bending) of T-Tail in Un-steady flow (left)
Pressure distribution (Real part, fin bending) of T-Tail in Un-steady flow (right)

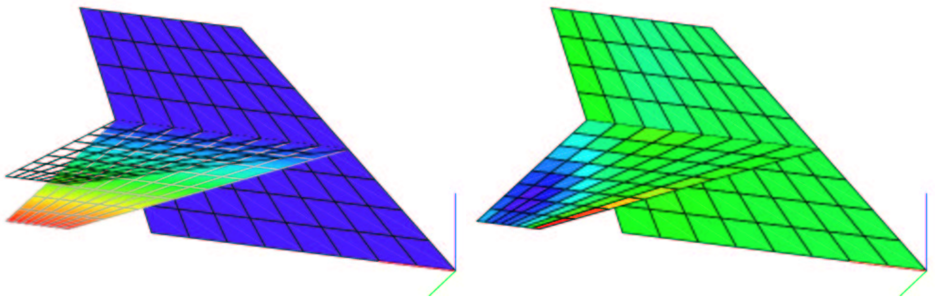

Figure 7c Displacement (stabiliser roll) of T-Tail in Un-steady flow (left)
Pressure distribution (Real part, stabiliser roll) of T-Tail in Un-steady flow (right)

7 Future Developments

As mentioned in the introduction, the application of the Doublet Lattice method lies in the fast calculation of aerodynamic loading in early phases of the design of an aircraft. In those phases the accuracy of the geometry, mass distributions and stiffness distributions is such that the use of Doublet Lattice methods is justified.
Within the development of the optimisation processor B2OPT [13] it is envisaged to incorporate the B2DL functionality within the definition of design constraints. The calculation of flutter constraints and their gradients with respect to design variables have been programmed already in a computer program called VLO [14]. This implies that the same (or similar) procedures to calculate flutter speeds and divergence speeds can be used. The actual implementation of B2DL is under preparation for incorporation within B2OPT.

In preparation to the definition of a flutter constraint a test case has been constructed based on a tapered wing (figures 8, 9 and 10) as defined in the work of Bisplinghoff [15]. Numerical results differ a bit in this case due to the fact numerical methods used by Bisplinghoff are different and last but not least material properties given in [15] are given by figures for a continues structure, while in B2DL the structures is discretised into beams.

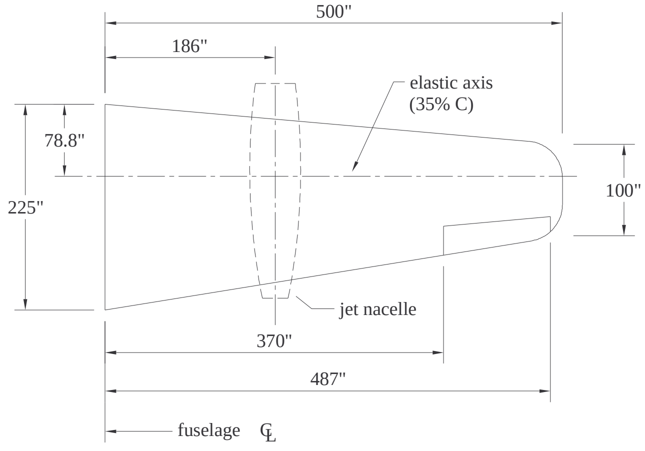

Figure 8: Planform definition of Bisplinghoff Jet Transport Wing

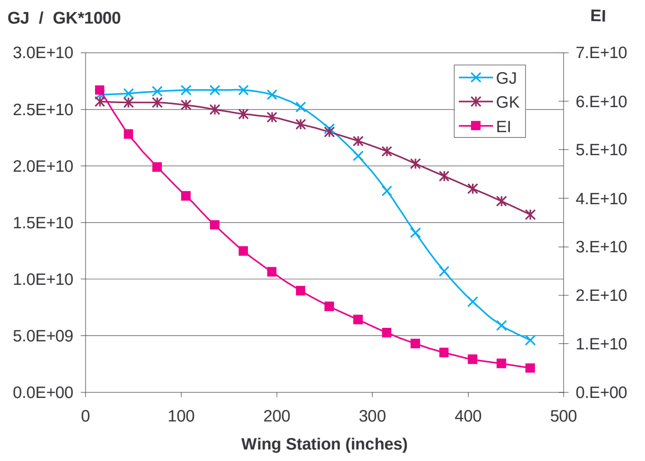

Figure 9: Stiffness Distribution of Bisplinghoff Jet Transport Wing

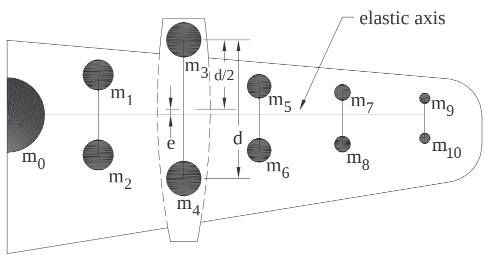

Figure 10: Dynamic Model of Bisplinghoff Jet Transport Wing

The structural model of the Bisplinghoff wing is a simple beam structure. The properties of the beam structures are per "segment" correspondent with figure 9.
The Doublet Lattice modelling is just a straight forward panel distribution.

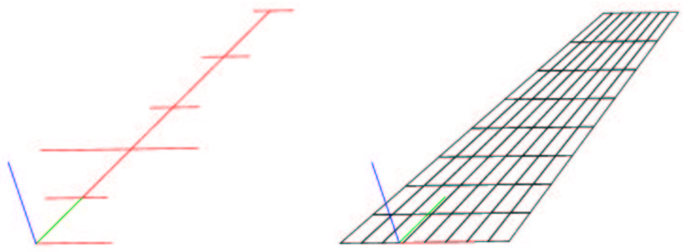

Figure 11: Structural and Doublet Lattice Geometry of Bisplinghoff Wing

The physical properties of the modelling gave some problems, apparently reading properties from a chart introduces significant errors. Model updating techniques are used to get those beam properties, which result in the influence coefficient matrix as prescribed in [15]. Having accomplished this, the vibration modes as found by B2DL are equivalent as those found in [15].

A nice feature already operational in B2DL is the mapping of nodal entities from one mesh to another, as taken from the theory described in [16]. In the pictures below (figure 11) the structural vibration modes are mapped upon the DL-mesh. As can be seen in the torsion mode (second) the mapping is sensitive to the precise location of mapping nodes near the root

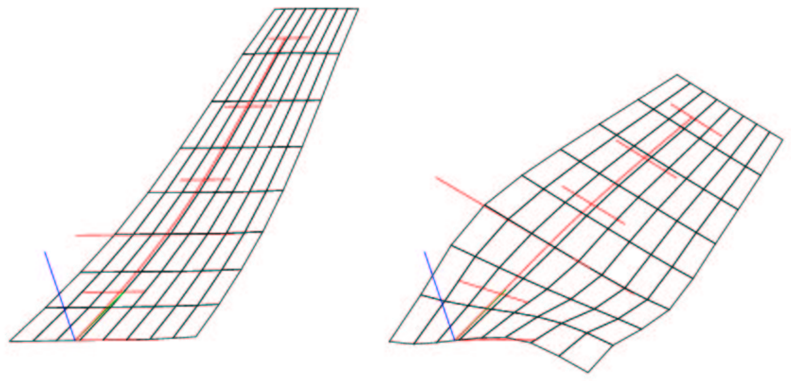

Figure 12: First (left) and second vibration mode mapped from structural to DL model

The pressure distributions given below are not comparable to results in [13], since in [13] the DL-based pressure distribution is not given, simply because the theory was not invented yet. In the development of a flutter analysis case, this Bisplinghoff wing can be used, since a numerical result for the flutter speed is given in [15]. The calculation of aerodynamic force for given set of reduced frequencies can be used in set op a Chebichev interpolation matrix. Based on this matrix, approximations of aerodynamic forces for other reduced frequencies can be made, including derivatives with respect the reduced frequency. This technique is part of the automated k-method to derive the flutter speed, which is to be implemented in B2OPT, using [14] as basis.

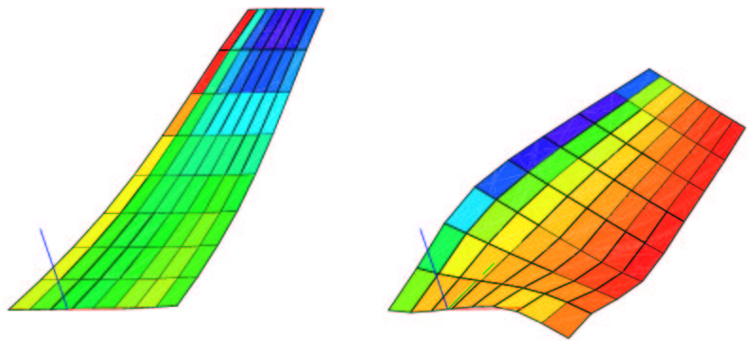

Figure 13: Pressure distribution belonging to first (left) and second vibration mode in Steady flow

8 References

[1] Albano, E., “*A Doublet-Lattice method for calculating lift distributions on oscillating surfaces in subsonic flows*”, AIAA Journal Volume 7 No. 2, page 279-285, 1969

[2] Stahl, B., Kalman, T.P., Giesing, J.P. and Rodden, W.P., “*Aerodynamic influence coeeficeinets for oscillating planar lifting surfaces by the Doublet Lattice method for subsonic flows, including quasi-steady fuselage interference, Part 1*”, Mc. Donnel Douglas Aircraft Corp., Rep. DAC-67201, 1969.

[3] Kalman, T.P., Rodden, W.P. and Giesing, J.P., “*Aerodynamic influence by the Doublet Lattice method for interfering non-planar lifting surfaces oscillating in a subsonic flow, Part 1*”, Mc. Donnel Douglas Aircraft Corp., Rep. DAC-67977, 1969.

[4] Rodden, W.P., Giesing, J.P. and Kalman, T.P., “*New developments and applications of the subsonic Doublet Lattice method for non-planar configurations*”, AGARD Symposium on unsteady aerodynamics for aero-elastic analysis on interfering surfaces, Paper 4, Douglas paper 5826, 1970.

[5] Zwaan, R.J., “*Theoretical unsteady pressure distributions and aerodynamic derivatives for the Fokker F28 T-tail wind tunnel model”*, NLR-report F256, National Aerospace Laboratory, NLR, Anthony Fokkerweg, Amsterdam, The Netherlands, 1965.

[6] Bennekers, B. and Labrujere, TH. E., *“Calculation of unsteady subsonic aerodynamic characteristics by means of the Doublet Lattice method”*, NLR-report TR 73031 U, National Aerospace Laboratory, NLR, Anthony Fokkerweg, Amsterdam, The Netherlands, 1973.

[7] Bennekers, B. and Labrujere, TH. E., *“A computer program for the calculation of subsonic aerodynamic characteristics of infinitely thin, non-planar, harmonically oscillating configurations”*, NLR-report TR 77138 C, National Aerospace Laboratory, NLR, Anthony Fokkerweg, Amsterdam, The Netherlands, 1972.

[8] S. Merazzi, *"B2000 - A Modular Finite Element Analysis System. Version 1.0"*, Memorandum SC-85-041 U, National Aerospace Laboratory, NLR, Anthony Fokkerweg, Amsterdam, The Netherlands, June 1985.

[9] SMR Corporation, *"B2000 Data Structure and Programming Handbook, version 1.77*", P.O. Box 41, CH2500 Bienne 4, Switserland, 1993

[10] SMR Corporation, *"B2000 Processors and Input Description, version 1.77*", P.O. Box 41, CH2500 Bienne 4, Switserland, 1993

[11] SMR Corporation, *"B2000 Online Programming Manual, version 2.5*", P.O. Box 41, CH2500 Bienne 4, Switserland, 2002

[12] SMR Corporation, *"B2000 On-line User Handbook, version 2.5*", P.O. Box 41, CH2500 Bienne 4, Switserland, 2002

[13] Arendsen, P., *"The B2000 Optimizaion Module: B2OPT"*, TP-94116 L, National Aerospace Laboratory, NLR, Anthony Fokkerweg, Amsterdam, The Netherlands, 1994.

[14] Arendsen, P., *"Multi-Level Optimalisatie met Aero-Elastische nevenvoorwaarden"*, Master-Thesis, Faculty of Aero-Space Engineering, Technical University of Delft, Delft, The Netherlands, 1988.

[15] Bisplinghoff, R.L., *“Aeroelasticity”*, Addison Wesley Publishing Inc, Library of Congress Catalog Card 55-6561, Cambridge, Mass. 1955.

[16] MSC, “*Handbook for Aeroelastic Analysis Volume 1*”, NASTRAN Version 65

Appendix A   B2DL input example (Benchmark Steady flow - Swept Wing)

```
(zero=0.0);
(half=0.5);
(nn1=9);
(nn2=9);
(ne1=nn1-1);
(ne2=nn2-1);
(ibeg=7);
(iend=9);
(ihinge=6);
(ione=1);
(itot=nn1*nn2);
(PICONSTANT=3.14159265);
(x1=zero);    (y1=zero);    (z1=zero);
(x2=600.00);  (y2=zero);    (z2=zero);
(x3=1010.4);  (y3=880.00);  (z3=zero);
(x4=410.4);   (y4=880.00);  (z4=zero);
(xhb=375.00);
(xhe=785.4);
(shiftx=xhe-xhb);
(shiftx=shiftx/ne2);
(deltax=x2-x1);
(deltax=deltax/ne1);
#
TITLE='Swept Wing Vortex Lattice';
branch=1;
  patch;
    nn1=(nn1); nn2=(nn2);
    p1  (x1) (y1) (z1);
    p2  (x2) (y2) (z2);
    p3  (x3) (y3) (z3);
    p4  (x4) (y4) (z4);
  end;
  wash;
    case=1; dof=3;
    (i=1);
    while ( i<=nn2 ) (
       (j=1);
       (const=((i-1)*shiftx));
       while ( j<=nn1 ) (
         val=(const+(deltax*(j-1))); node=(j+((i-1)*nn1));
         (j=j+1);
       );
       (i=i+1);
    );
  end
endbranch
dlctrl
   igegen=1 ipress=1 icode=2 isym=1 iground=1
   scorder=(600.00) hspan=(880.00)
   nredkr=0 mach=0.8
end
ias
   sym 1
end
run
```