Australian Government
Department of Defence
Defence Science and
Technology Organisation

HyPAq: Software for the transient analysis of hypersonic vehicles in *Abaqus*

Nigel S. A. Smith

Air Vehicles Division

Defence Science and Technology Organisation

DSTO–TR–2236

ABSTRACT

The Defence Science and Technology Organisation (DSTO) has embarked upon a research and development program for airbreathing hypersonic vehicles. Part of this program involves the selection and development of software tools useful in the aerothermo-structural design of these vehicles. *HyPAq* is a hypersonic package for *Abaqus* – a set of programs and subroutines to be used in conjunction with that particular finite element analysis code. *HyPAq* interpolates pressures and convective heat fluxes derived from any arbitrary set of CFD results onto *Abaqus* transient coupled temperature-displacement analyses. This report describes the algorithms which underpin *HyPAq* and provides an example set of results. The example results are from an analysis which aims to determine a suitable heat-sink material for the nose section of a lifting body undergoing Mach 8 sub-orbital re-entry. The complete user guide provided with the *HyPAq* software distribution is included in an appendix.

Published by

DSTO Defence Science and Technology Organisation
506 Lorimer St,
Fishermans Bend, Victoria 3207, Australia

Telephone: (03) 9626 7000
Facsimile: (03) 9626 7999

AR No. AR-014-370
October 2008

HyPAq*: Software for the transient analysis of hypersonic vehicles in *Abaqus

Executive Summary

The Defence Science and Technology Organisation (DSTO) has embarked upon a research and development program for airbreathing hypersonic vehicles. In collaboration with the United States Air Force Research Laboratory, DSTO is engaged in the HIFiRE (Hypersonic International Flight Research Experiments) program of ten hypersonic flight experiments launched from Woomera over an eight year period. HIFiRE has as its ultimate goal, the achievement of sustained level flight by a hypersonic airbreathing vehicle. Meeting this goal is an indispensable step along the path towards fielding an operational defence system. The promise of airbreathing propulsion over conventional rocketry is that such a system will be able to carry a heavier payload over hundreds of kilometres in a handful of minutes; all the while manoeuvering under power and at great speed to avoid defences and countermeasures.

In a flight regime where near-wall gas temperatures can exceed 3000 K, the challenge of sustained hypersonic flight for an airbreather is to remain physically intact once thermal equilibrium has been reached. In 2008, the *TempEst* code (version 1.16) was released to quickly and accurately evaluate the time-dependent heat loads and the thermal responses of hypersonic vehicles as they follow complete flight trajectories. *TempEst* allows designers to rapidly explore the thermostructural viability of different hypersonic airbreathing concepts before proceeding to detailed design, testing and manufacture. The developmental arc of *TempEst* has reached a natural end point due to its original intent and its applicability only to surfaces adjacent to vehicle flow paths whose geometry is known in advance.

HyPAq is a complementary tool to *TempEst* which trades off speed and simplicity for a dramatic improvement in general applicability. It is a set of programs and subroutines which is to be used in conjunction with the *Abaqus* finite element analysis code. *HyPAq* seamlessly interpolates pressures and convective heat fluxes derived from any arbitrary set of computational fluid dynamic results onto *Abaqus* transient coupled temperature-displacement analyses. In contrast to *TempEst* , *HyPAq* will be used in the final stages of design where detailed analyses of arbitrarily complex vehicle structures in arbitrary flow conditions are required.

This report describes the algorithms which underpin *HyPAq* and provides an example set of results from past use relevant to the HIFiRE Project. This example deals with analysis to determine an appropriate heat sink material for the nose section of a lifting body undergoing Mach 8 sub-orbital re-entry. Recommendations are made for the further development of *HyPAq* to improve its ease of use and applicability to exotic materials. The complete user guide provided with the *HyPAq* software distribution is included in an appendix of this report as a detailed reference.

Author

Nigel S. A. Smith
Air Vehicles Division

Nigel Smith completed a Bachelor of Engineering with Honours (Mechanical) at the University of Western Australia in 1990. Between 1991 and 1994, he studied towards a Ph.D. at the University of Sydney, in the Department of Mechanical Engineering. His research dealt with the development and testing of the Conditional Moment Closure method for predicting turbulent combustion processes. Along the way, he was able to travel and work extensively at the Combustion Research Facility at Sandia National Laboratories in Livermore, California, and at Cambridge University.

Upon completion of his Ph.D. research and thesis in September of 1994, he took up a position as a postdoctoral fellow at the Center for Turbulence Research, a joint research facility operated by Stanford University and the NASA Ames Research Center. After two years of research into turbulent combustion using direct numerical simulation, he found his way back from Northern California to Melbourne, where he has taken up a research scientist position in the then Airframes and Engines Division.

Since that time he has conducted extensive research into the accurate numerical modelling of fundamental processes in turbulent combustion; from gas-phase turbulence-chemistry interactions, soot formation and spray combustion in gas turbine combustors to infra-red emissions from the coronal flame mantles of rocket exhaust plumes. He has written several computational fluid dynamic codes for use in DSTO and elsewhere to predict many of these phenomena.

Nigel Smith is currently Head of the Signatures and Thermodynamics Group in Air Vehicles Division. One role of this Group is to undertake detailed numerical analyses of the fluid flows, loads and performance of hypersonic airbreathing propulsion systems. This is being done in support of a collaborative project arrangement between DSTO, the United States Air Force Research Laboratory, and the University of Queensland.

Contents

Appendices

1 Introduction

Hypersonic airbreathing propulsion is the subject of a significant research initiative in DSTO. The promise of airbreathing propulsion over conventional rocketry is that such a system will be able to carry a heavier payload over hundreds of kilometres in a handful of minutes; all the while manoeuvering under power and at great speed to avoid defences and countermeasures.

In partnership with the US Air Force, DSTO is currently working towards the technological goal of achieving level flight by a hypersonic airbreathing vehicle for a duration on the order of minutes. In the past, hypersonic airbreathing flight has been demonstrated for a number of seconds[1, 2] as proof of engine flowpaths and performance. Sustained flights require a much stronger focus on understanding and managing the aerothermodynamic loads on vehicles. In a flight regime where near-wall gas temperatures can exceed 3000 K, the challenge of sustained hypersonic flight for an airbreather is to remain physically intact once thermal equilibrium has been reached. Achieving the goal of sustained flight is an indispensable step along the path towards fielding an operational defence system.

Perhaps more than any other class of hypersonic vehicle, the hypersonic airbreather is subject to the heat loads associated with high speed atmospheric flight. This is due in part to the air intakes and combustion chambers required in an airbreather, with their larger surface areas exposed to fierce stagnation and frictional heating. More importantly, airbreathers are obliged to remain at altitudes where there is sufficient oxygen to generate useful thrust and so are subject to prolonged heating. In contrast, rocket-propelled vehicles either rapidly transit the middle atmosphere or, if they are meant to remain within the atmosphere, they inevitably exhaust their propellant and so rapidly drop into the more moderate heating regime associated with low supersonic flight.

Over the past few years, DSTO has been building the capability to predict the heat loads associated with hypersonic flight[3] and the corresponding structural responses[4]. Software development has produced a rapid design tool[5, 6] known as *TempEst* which allows transient coupled thermo-structural heating to be examined for selected vehicle flow paths and skin structures on prescribed flight trajectories.

For completely arbitrary geometries where flow paths are not known *a priori* or where more than simple thin-skin structures are to be examined, then a more comprehensive approach is required. The feasibility of a single software solution that will simultaneously model transient three-dimensional fluid flows and structures in a hypersonic domain is quite low. This is partially due to the wide disparity in the time scales relevant to hypersonic (potentially reacting) fluid flows and conduction and deformation in solid structures. Another complication is that structural analysis is usually conducted in a Lagrangian frame of reference, while fluid analysis is conducted in an Eulerian frame.

Instead a general approach should attempt to direct outputs from three-dimensional computational fluid dynamics (CFD) for viscous hypersonic flows as inputs into three-dimensional transient coupled thermo-structural finite element analysis (FEA). For this reason, attention will be focussed on the commercial FEA package *Abaqus* that is licensed for use in DSTO. *Abaqus* allows transient analysis of thermo-structural responses, has the capacity to incorporate user-defined Fortran routines and has excellent on-line documentation[7]. These features make *Abaqus* a good FEA candidate for adopting such an approach.

This report describes *HyPAq* (version 0.90), a Hypersonic Package for *Abaqus* (version 6.7). *HyPAq* consists of stand-alone programs and *Abaqus* user-defined subroutines that can dynamically interpolate convective heat fluxes and pressures from arbitrary CFD software onto the surfaces of structures that are the subject of *Abaqus* transient coupled temperature-displacement analyses.

In the following, the underlying algorithms in *HyPAq* are described (Section 2), a simple example is provided of how *HyPAq* has been used in practice for a hypersonic re-entry vehicle (Section 3), and recommendations are made for future development (Section 4). The comprehensive user documentation, which is bundled with the version 0.90 software distribution, is provided for reader reference in Appendix B.

2 Methods

HyPAq simultaneously employs three distinct types of processes in order to provide CFD convective heat flux data to *Abaqus* during a coupled temperature-displacement analysis. Firstly, it interpolates CFD data over its defined surface onto the surface locations required by *Abaqus* (see Section 2.1). Secondly, for these interpolated locations, it interpolates between adjacent CFD data sets in time (see Section 2.2). Lastly, it moderates the fluxes to account for disparities between the surface temperatures computed by *Abaqus* at each location and time step, and those found through the above spatial and temporal interpolation (see Section 2.3).

2.1 Spatial Interpolation

Spatial interpolation involves using the set of variable values at discrete data points generated by the CFD source as a basis for finding consistent values at the surface point locations required by the FEA analysis. To avoid spatial interpolation difficulties, the surface geometry of the CFD defined object should nominally agree with the *Abaqus* defined geometry, even though their surface discretisations will vary.

The *HyPAq* spatial and temporal interpolator (`intphyper.f`) generates the interpolation data file required by *Abaqus* at run time. The interpolator reads in a file (`hyper_meshpt.txt`) created by *HyPAq* during a preliminary run of *Abaqus* which contains all the physical locations in space for which *Abaqus* expects the provision of convective film conditions. The interpolator also reads in a series of data files which have been parsed from their native CFD source format into a simple *HyPAq* native format, with all the spatial locations defined for a single variable for a single time point per file. The *HyPAq* native format required of these files is described in the User Guide listed in Appendix B. The variables required from the CFD source are the convective heat flux, wall temperature, local total temperature and local static temperature for convective film conditions, and the static pressure for surface distributed loading. Note that the total and static temperatures referred to here are for the flow adjacent to the surface but outside the boundary layer. These values are used to apply a reference enthalpy correction to heat fluxes where the *Abaqus* computed wall temperature deviates from the interpolated wall temperature as described in Section 2.3.

Each of the above variable files contains a data field defined at a single point in time and at any number of points in space defined in (x, y, z) coordinates. Note that the spatial distribution of points within these files need not agree between variables nor time indices. This flexibility was deemed essential given the likelihood that there would be different CFD meshing requirements for different flow cases throughout a flight trajectory.

Spatial interpolation is carried out once only before the commencement of the coupled temperature-displacement analysis using the initial locations of *Abaqus* surface points. This is done primarily because, due to computational cost, it is highly unlikely that progressively deformed surface geometries would be used for a series of CFD analyses to provide the perturbed input data. Structures with a high enough level of deformation to warrant flow re-computation are likely to have failed from the point of view of integrity in any case. Lastly, the cost of the general unstructured spatial interpolation scheme described here is too high to be applied at every time-step. In this scheme, for each *Abaqus* designated location (X, Y, Z) different subsets of the complete set of CFD-designated points $(x_{i,j}, y_{i,j}, z_{i,j})$ where $i = 1, \ldots, N_j$ are examined. For the j-th subset, which has N_j member points, the i-th point has an inverse square distance value

$$s_{i,j} \equiv \max\Big(\,\epsilon,\ (x_{i,j} - X)^2 + (y_{i,j} - Y)^2 + (z_{i,j} - Z)^2\,\Big)^{-\frac{1}{2}}\ , \tag{1}$$

where ϵ is a very small number that avoids a singularity if the i-th point coincides with the given *Abaqus* point. The set of these inverse distance values is employed to generate non-dimensional weights

$$w_{i,j} \equiv s_{i,j}\left[\sum_{i=1}^{N_j} s_{i,j}\right]^{-1} \tag{2}$$

for each point, where it is clear that each weight is bounded $(0 < w_{i,j} \leq 1)$ and each set of weights has a unit sum. The weights for the j-th subset of CFD-defined points allow any data field $f_{i,j}$ to be spatially interpolated to give the appropriate value $f_{w,j}$ at the given *Abaqus* point (X, Y, Z), where

$$f_{w,j} = \sum_{i=1}^{N_j} w_{i,j}\, f_{i,j}\ . \tag{3}$$

In general, the interpolated value will depend on the selection of member points in the subset. Consequently, the quality of the approximation does vary between subsets.

There is a simple means to check the quality of the approximation for any given set of CFD points and *Abaqus* point. This check is indicative of how well linear data fields will be interpolated. The interpolation of non-linear data fields depends additionally on variable curvature and still-higher derivatives in space. Using this check, the subset member locations $(x_{i,j}, y_{i,j}, z_{i,j})$ and their weights $w_{i,j}$ can be used to find a relative interpolated displacement

$$\Delta_j \equiv S_j\left[(x_{w,j} - X)^2 + (y_{w,j} - Y)^2 + (z_{w,j} - Z)^2\right]^{\frac{1}{2}} \tag{4}$$

for the *Abaqus* point location (X, Y, Z), where the physical coordinates resulting from the interpolation (as for Eqn 3) are

$$x_{w,j} = \sum_{i=1}^{N_j} w_{i,j}\ x_{i,j}\ ,$$

$$y_{w,j} = \sum_{i=1}^{N_j} w_{i,j}\ y_{i,j}\ ,$$

$$z_{w,j} = \sum_{i=1}^{N_j} w_{i,j}\ z_{i,j}\ ,$$

and the distance scale factor is the minimum inverse displacement for the j-th set of points,

$$S_j \equiv \min\left(s_{i,j}\right)\ . \tag{5}$$

The subset which produces the smallest relative displacement is selected as the best subset for interpolation onto the given *Abaqus* point. It has been found that the best approximation occurs for that subset which includes points which spatially bracket the given *Abaqus* point. Typically, if this displacement is zero then the *Abaqus* point is bracketted, or else it is bracketted in a reduced number of dimensions on a surface that is displaced in the remaining orthogonal dimension. Where more than one subset produces an equal minimum displacement, the subset which has the largest value of S_j is selected since it is more local to the given *Abaqus* point.

The number of subsets (n_{ss}) potentially involved in the selection process is governed by the factorial expression

$$n_{ss} = \frac{M!}{(M-N)!\ N!}\ , \tag{6}$$

where N is the size of an allowable subset (N_j is assumed constant for all j) and M is the number of CFD points available from which to choose. In order to avoid an astronomical number of subsets for each *Abaqus* point, it is neccessary to limit the 'pool-size' (M) to a number usually much less than the complete set of CFD points. This is done by specifying that the selection pool is only made up of the M points closest to the designated *Abaqus* point in each case.

The user should select N so that it provides enough CFD designated points to properly bracket an *Abaqus* designated point in the topology of the problem at hand. This relates to the structure and dimensionality of the CFD designated object (see Figure 1). For example, for a two-dimensional CFD surface consisting of quadrilaterals, then N should be four to properly bracket any given *Abaqus* point. For a CFD surface consisting of triangles, then N should equal three. For one-dimensional line boundaries on two-dimensional objects, N should equal two. Interpolation for spaces with any number of dimensions can be undertaken in principle but are not required for use in this context where only one- and two-dimensional surfaces are possible for two- and three-dimensional objects.

In choosing M, the user must bear Equation 6 in mind and understand that n_{ss} subsets will be examined at each *Abaqus* point for each variable and time index. This fact would suggest that M should be only slightly larger than N; for a regular structured CFD surface mesh with $N = 4$, a choice of $M = 8$ would give $n_{ss} = 70$ as the number of subsets

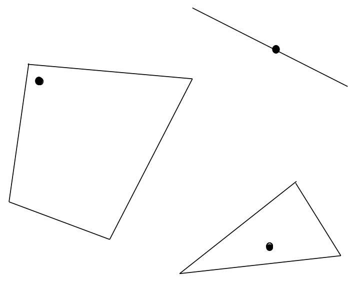

Figure 1 (U): *Illustration of the number of mesh vertices required to bracket designated points on one- and two-dimensional surfaces of two- and three-dimensional objects. In two dimensions, structured meshes require four points and unstructured meshes require three. In one dimension, only two mesh points are required.*

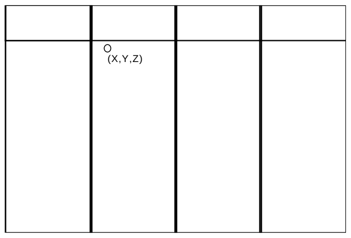

Figure 2 (U): *Example of how non-uniform mesh spacing can require a large selection pool of vertices to ensure the bracketting of the designated point. In this case, the pool-size must be at least 12 in order to select the bracketting vertices.*

evaluated for each *Abaqus* point. On the other hand, some CFD geometries with highly non-uniform spacing can require quite distant points to adequately bracket some *Abaqus* points (see Fig 2).

For the poorly-spaced CFD mesh depicted in Figure 2, if M is less than 12 then the example *Abaqus* point will fail to be bracketted. It is clear that the selection of M and N requires careful thought on the part of the user and potentially more than one attempted run with the interpolator to get a good interpolation solution everywhere. To assist in this process, check output files are always written which contain Δ_j values for the selected subsets for each variable at each *Abaqus* point for all time indices (see Appendix B).

In general, CFD surface meshes which are finer than the *Abaqus* surface mesh lead to better interpolation outcomes. Abrupt changes in surface discretisation density of the CFD defined object should be avoided wherever possible.

2.2 Temporal Interpolation

Interpolation between adjacent time indexed CFD data is done by one of two methods. The first of these is simple linear interpolation in time. More commonly, however, another monotonic interpolation quantity will be specified. This is of value where CFD data sets are sparse in time and simple linear interpolation based on time between them is inappropriate. For example, during a flight trajectory, one might only have two data sets at widely spaced altitudes. In that case it would make more sense to linearly interpolate between sets based on altitude or dynamic pressure. This functionality allows altitude or dynamic pressure or any other salient variable to be tabulated to allow this to occur.

Formalising this example, consider the two CFD data sets. The first contains variables only at time τ_1 which are denoted F_1. The second's variables, F_2, apply at time τ_2. *HyPAq* has been provided by the user (see Appendix B) with a table of interpolation function (g) values at regularly-spaced intervening time values ($\tau_1 \le t \le \tau_2$). During the transient *Abaqus* computation, *HyPAq* uses the provided time arguments to linearly interpolate among the time entries to find the appropriate value of the interpolation function

$$g(t) = g(t_{k-1}) + (g(t_k) - g(t_{k-1}))\,\frac{t - t_{k-1}}{t_k - t_{k-1}} \quad \text{where} \quad t_{k-1} \le t \le t_k\ , \tag{7}$$

and then uses that value to interpolate between the monotonic function values for the two CFD data sets,

$$f(g) = \frac{g(t) - G_1}{G_2 - G_1}\,(F_2 - F_1) + F_1 \quad \text{where} \quad G_1 \le g(t) \le G_2\ , \tag{8}$$

to yield the desired variable values for pressure, heat flux and so on.

2.3 Surface Temperature Interpolation

There will practically always be a disparity between the surface temperature computed by *Abaqus* at each time step and the surface temperature used in the derivation of CFD data at the same surface location. This disparity must be accounted for in determining the convective heat flux appropriate to the *Abaqus* surface condition.

Firstly, *Abaqus* employs the following film condition to model convective heat transfer,

$$\dot{q} = K_{film}\left[T_{sink} - T_w\right]\ , \tag{9}$$

where the heat flux $\dot{q}$ is a function of the film coefficient K_{film} and 'sink' temperature T_{sink} which must be provided by *HyPAq* to *Abaqus* and do not vary over the course of a time-step. The wall temperature T_w in the above equation is that computed by *Abaqus* and used by it at the surface for each step. Convective heat transfer at high Mach numbers is more appropriately treated using an enthalpy-based expression[8] such as

$$\dot{q} = \rho_* V C_H\left[h_{aw} - h_w\right]\ , \tag{10}$$

where V is the tangential velocity of the flow adjacent to the wall but outside any boundary layer, and ρ_* is a 'reference' density[8]. This is the density corresponding to a fluid state where the pressure equals the static pressure and the enthalpy is given by

$$h_* = 0.5h_w + 0.22h_{aw} + 0.23h_{static}\ . \tag{11}$$

In the above two equations, the convective coefficient C_H is analogous to a skin friction coefficient. The wall enthalpy h_w is the enthalpy of fluid immediately adjacent to the wall which is at the wall temperature. The adiabatic wall enthalpy h_{aw} is given by

$$h_{aw} = h_{static} + \frac{1}{2}V^2 Pr^{\frac{1}{2}}\ , \tag{12}$$

where h_{static} is the static enthlpy of the fluid outside the boundary layer, and the square root of the Prandtl number is a recovery factor. In order to have the film condition expressed in Equation 9 met by the enthalpy-based expression, it is necessary to regroup Equation 10 so that

$$\dot{q} = \rho_* V C_H \frac{h_w}{T_w} \left[T_{sink} - T_w \right] \quad \text{where} \quad T_{sink} \equiv h_{aw} \frac{T_w}{h_w} \, . \tag{13}$$

In the above heat flux expression, the product of variables ahead of the temperature difference is an effective film coefficient K_{film} for the duration of one time-step. After each time-step the effective film coefficient and sink temperature must be re-evaluated given new values of the computed wall temperature. Note that the velocity and adiabatic wall enthalpy are not functions of wall temperature.

In order to use Equation 9, selected items of information must be stored for interpolation by *HyPAq* in conjunction with *Abaqus* convective films. Note that in the following, a prime notation ($'$) is used to signify a variable which is stored and interpolated rather than directly computed. The stored film coefficient

$$K'_{film} = \frac{\dot{q}'}{T'_{sink} - T'_w} \tag{14}$$

is evaluated from CFD data before *Abaqus* run time, based on the CFD derived heat flux $\dot{q}'$, wall temperature, and the sink temperature corresponding to the CFD wall temperature and flow conditions using definitions given above. At run time, the difference between the stored and computed wall temperatures is used to correct the stored film coefficient K'_{film} to give the actual film coefficient

$$K_{film} = K'_{film} \left[\frac{h_w T'_w}{h'_w T_w} \right] \left[\frac{T'_* \mu_*}{T_* \mu'_*} \right]^{\frac{1}{2}} \, , \tag{15}$$

where terms involving the wall temperature and enthalpy are renormalisations that account for the change in wall temperature. The influence of a change in wall temperature is evident in the expression for determining the computed sink temperature

$$T_{sink} = h'_{aw} \frac{T_w}{h_w} \, , \tag{16}$$

where only the adiabatic wall enthalpy is from the stored CFD data. The terms containing asterisks ($*$) in Equation 11 again denote reference quantities which require either the stored or computed wall temperature to find the appropriate reference enthalpy as given in Equation 11. The reference term renormalisation grouping in Equation 15 goes toward accounting for changes in heat transfer arising from wall temperature effects on reference density and the local boundary layer width. Reference to Equation 13 shows that the film coefficient is proportional to density and the heat transfer coefficient C_H. The heat transfer coefficient is in turn proportional to the inverse square root of the local Reynolds number[8]. These proportionalities combine according to,

$$K_{film} \propto \rho_* C_H \propto \rho_* / Re_*^{\frac{1}{2}} \propto (\mu_* \rho_*)^{\frac{1}{2}} \propto \left(\frac{\mu_*}{T_*} \right)^{\frac{1}{2}} \, , \tag{17}$$

Table 1: *Approximated temperature-invariant properties used for different materials for nose section of re-entry vehicle.*

Property	Copper	Stainless Steel
Thermal Conductivity $[W/(mK)]$	401	16.2
Specific Heat $[J/(kgK)]$	400	500
Density $[kg/m^3]$	8960	7990
Young's Modulus $[Pa]$	1.1E11	1.77E11
Expansion Coefficient	1.65E-5	1.60E-5
Poisson's Ratio	0.34	0.30
Surface Emissivity	0.3	0.3

thus yielding the requirement for the renormalisation given in Equation 15. The reference temperature effect can be thought of as increases in wall temperature leading to reductions in heat transfer coefficient due to density-related boundary layer thickening. Conversely, increases in wall temperature lead to increases in the reference viscosity which somewhat offsets this behaviour.

For the above interpolations, *HyPAq* uses in-built polynomial data fits for air for (a) sensible enthalpy as a function of temperature and vice versa, and (b) dynamic viscosity as a function of temperature. These fits are applied consistently in the interpolator and the *Abaqus* routines so that the actual identity of the gases present in the fluid is not used at any stage. In this regard, these fits are meant to represent the non-linearity of gases in general at elevated temperatures rather than the exact details of curves for particular species.

3 Results

HyPAq capability is illustrated in this section through the coupled thermo-structural analysis of the nose section of a lifting body undertaking a hypersonic re-entry. The lifting body is of a waverider design whose aerodynamic coefficients were assessed using inviscid CFD. These coefficients were then used to compute the re-entry trajectory which terminates with the vehicle at a 60° nose-down flight path angle at 28 km altitude at Mach 8. During the five minute re-entry from a sub-orbital apogee altitude of 450 km, the vehicle has pulled-up through 18° from its maximum nose-down flight path angle. Further details of the re-entry trajectory are provided in Appendix A.

During the 23 second atmospheric portion of the trajectory, the nose section of the vehicle is subjected to extreme heating. At the conclusion of the trajectory, the vehicle's structural integrity is not required any further. So, based on *HyPAq* analysis, a simple design choice of heat sink material is to be made for the nose. Two common materials are considered for the task, namely copper and stainless steel, whose relevant elastic and thermal constitutive properties are listed in Table 1. The principal difference in these quantities between the two materials lies with thermal conductivity. Copper is an excellent conductor of heat while stainless steel is not.

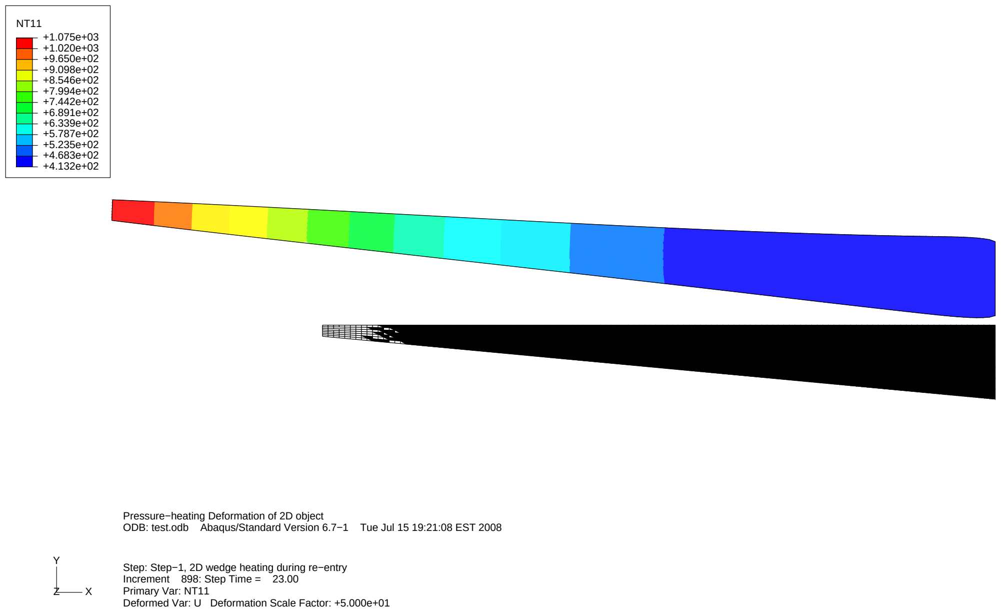

Figure 3 (U): *Copper temperature contours at the completion of re-entry plotted on the corresponding deformed structure (50-fold magnification). The offset black silhouette is the original undeformed structure. Temperatures range from* 413 *to* 1075 *K in twelve* 55 *K increments.*

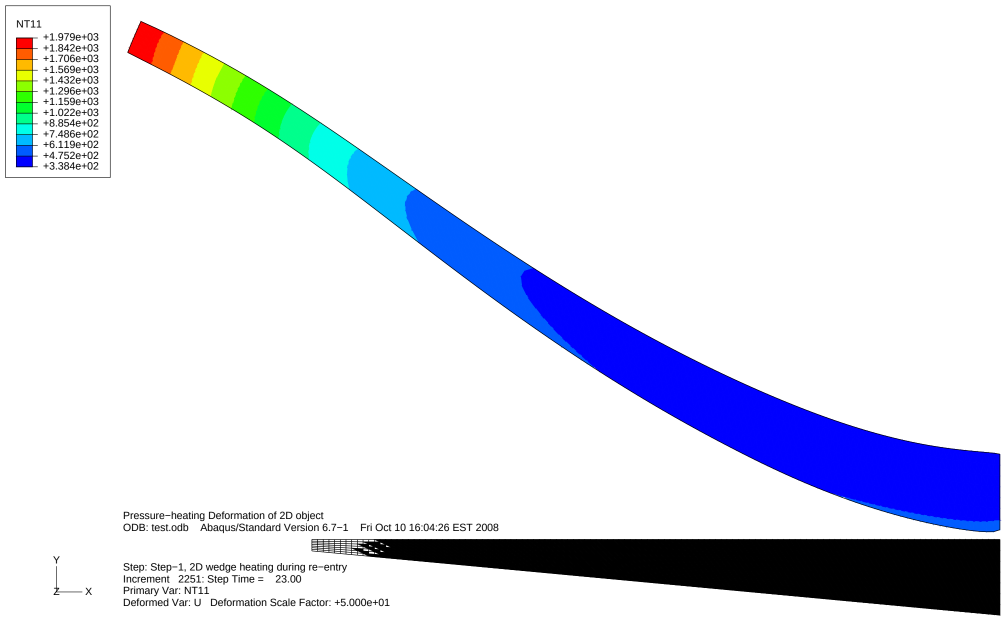

Figure 4 (U): *Stainless steel temperature contours at the completion of re-entry plotted on the corresponding deformed structure (50-fold magnification). The offset black silhouette is the original undeformed structure. Temperatures range from* 338 *to* 1979 *K in twelve* 136 *K increments.*

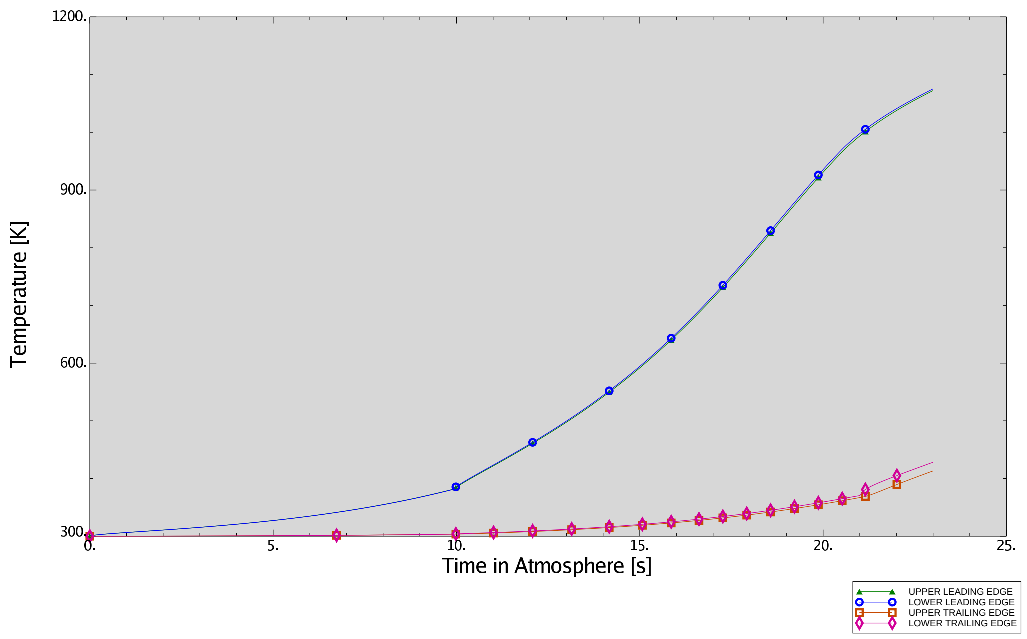

Figure 5 (U): *Copper temperature traces as a function of time at upper and lower surface corners with the leading and trailing edges.*

Although not directly used in the *HyPAq* analysis, the melting points of the two metals also differ. Stainless steel melts at 1811 K while copper melts at 1357 K. For copper to be successful as a heat sink material in this role, its higher thermal conductivity will be required to transfer heat away quickly enough from hot spots to avoid localised melting.

For this analysis, the nose section is modelled as a monolithic two-dimensional block on the centreline of the vehicle. Transient heat transfer rates over the re-entry trajectory were estimated for the upper and lower flowpaths on the centreline using the *TempEst* code[5, 6]. The nose block is a quadrilateral with the proportions illustrated in Figure 3 by the black silhouette. The quadrilateral has a length of 120.95 mm, a leading edge (left hand side) thickness of 2 mm, and a trailing edge (right hand side) thickness of 13.284 mm. A semi-circular leading edge of 1 mm radius is neglected here, with a cut-off blank face instead. Leading edge heat transfer rates are thickness-averaged for this case from a *TempEst* analysis of an actual 1 mm radius nose. The trailing edge is fixed mechanically in space and is an adiabatic barrier to heat transfer. Radiant emission occurs from all surfaces, except the fixed trailing edge, with an emissivity of 0.3 and a background temperature of 300 K.

The initial temperature of the entire nose section is 300 K prior to re-entry. Time-steps in the transient *Abaqus* analysis are limited by a maximum change in temperature of 1 degree for any of the 976 (122×8) cells present in the two-dimensional mesh.

Figures 3 and 4 are plots of the material temperatures of the copper and stainless steel variants of the nose structure at the end of the re-entry trajectory. In each case the temperature plots are made on the final deformed shape of the structure. The offset shape in each figure is the original undeformed structure whose dimensions are given above and this is provided as a visual reference. It is immediately apparent from these figures that the

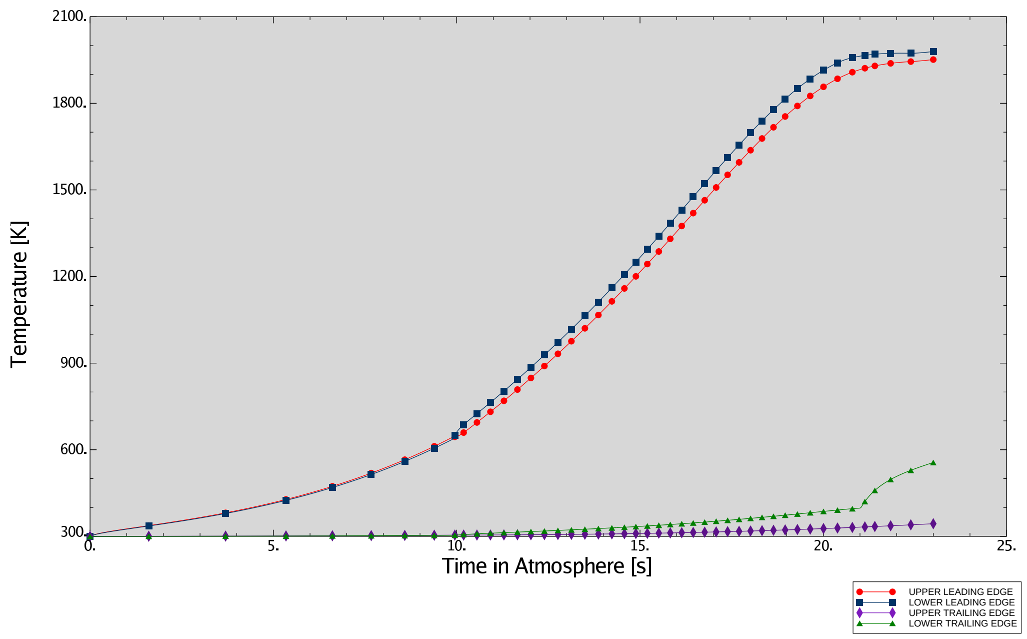

Figure 6 (U): *Stainless steel temperature traces as a function of time at upper and lower surface corners with the leading and trailing edges.*

stainless steel structure is significantly more deformed (2 mm upward deflection) compared with the copper structure (0.14 mm upward deflection). This is despite experiencing the same pressure and aero-heating environment. The increased deformation in the stainless steel is due to the greater degree of differential thermal expansion which is in turn due to the wider range in temperatures across the structure which result from its poorer thermal conductivity. Close attention to the figures shows that the peak temperature for the stainless steel structure is around 900 K higher than for the copper structure, while its minimum is 75 K lower than for the copper.

Figures 5 and 6 are plots of the temperature histories of selected locations on the copper and stainless steel nose sections respectively. The selected locations are the upper and lower corners of the leading and trailing edges. Over the 23 seconds of atmospheric heating, the copper leading edge corners warm from 300 to 1075 K with very little distinction between the upper and lower surfaces. The trailing edge corners warm to 413 and 421 K for top and bottom surfaces, which is a reflection of the fact that the lower surface is subjected to much higher heating rates as the vehicle re-enters at a significant angle of attack.

Compare the above copper traces with that for stainless steel and its poor conductivity. The leading edges warm to 1951 and 1979 K for the top and bottom surfaces. The trailing edges warm to 338 and 561 K for the top and bottom surfaces. As a result of poor conductivity, the difference in surface temperature between top and bottom surfaces that can be sustained by stainless steel is very large compared to copper. This temperature difference drives the differential expansion so evident from comparing Figs 3 and 4.

Lastly, it is important to note that the peak temperature for the copper structure is some 282 K below its melting point. However, for stainless steel, the peak temperature exceeds its melting point by 168 K and so localised melting is deemed to occur. In the absence of temperature-dependent yield strength properties for the materials, it is not possible to conclude that copper has adequate strength to avoid failing under the re-entry pressure load. On the other hand, local melting of the structure precludes stainless steel as a practical material for the nose section. It is worthwhile to note that other applications of heat sink materials in hypersonic heating have allowed some localised yield provided that the overall structure retains its integrity[4].

Although not examined in this case, differential expansion in constrained configurations leads to significant stresses that can cause local yielding even in the absence of surface pressure loads. The existing implementation of *HyPAq* in *Abaqus* is perfectly capable of being employed to examine stresses arising in these instances.

4 Concluding Remarks

The spatial, temporal and temperature interpolation methods employed in *HyPAq* have been described. These allow the input of CFD data from generic sources which is completely devoid of any structured or unstructured topological information. These data are employed by *HyPAq* to provide transient surface pressure loads and convective film interactions for coupled temperature-displacement *Abaqus* analyses.

A simple example of past *HyPAq* use has been provided. These *HyPAq* results demonstrate that copper may be an appropriate heat sink material for the leading edge nose section of a lifting body for the described Mach 8 re-entry trajectory. In contrast, stainless steel was found to deform excessively due to differential thermal expansion and also exhibited localised melting.

Future developmental work related to *HyPAq* should focus on :

- Enhancing the user's ability to check the goodness of the spatial interpolation solution. Presently, an output file is generated with relative displacement information in the same unstructured order that is provided by the CFD data source. This check-output should be made in some convenient plot format instead.
- Providing the user with a means to visualise the interpolated pressure and convective film values on the FEA model itself. Currently *Abaqus* does not directly support the visualisation of user-defined fields for distributed loads and convective film interactions. There may, however, be a convenient way of creating ersatz data fields which can be plotted in the *Abaqus* viewer. This technique may also provide a means of examining the interpolation solution quality (see above item).
- Temperature dependent constitutive properties for materials. *HyPAq* analyses are only as valid as the constitutive property information available. This information, particularly in the case of yield strength at very high temperatures, is somewhat sparse. Wherever possible, efforts should be directed to increase the accumulated knowledge in this area.

References

1. Boyce, R. R., Gerard, S., and Paull, A.. 2003. "The HyShot scramjet flight experiment - flight data and CFD calculations compared", AIAA Paper AIAA-2003-7092. Presented at the 12th AIAA International Space Planes and Hypersonic Systems Conference, USA, 2003.

2. McClinton, C. R.. 2006. "X-43 Scramjet Power Breaks the Hypersonic Barrier - Dryden Lectureship in Research for 2006", AIAA Paper AIAA-2006-1, Presented at the 44th AIAA Aerospaces Sciences Meeting and Exhibit, 9-12 January 2006, Reno, Nevada.

3. Higgins, K. 2008. *Comparison of Engineering Correlations for Predicting Heat Transfer in Zero-pressure-gradient Compressible Boundary Layers with CFD and Experimental Data.* DSTO Technical Report, DSTO-TR-2159. Defence Science and Technology Organisation, Commonwealth of Australia.

4. Ho, S.-Y. and Paull, A.. 2006. "Coupled thermal, structural and vibrational analysis of a hypersonic engine for flight test", *Aerospace Science and Technology*, v10, pp420-426.

5. Smith, N. S. A. 2007. *TempEst : A Program for Estimating the Surface Temperatures of Hypersonic Vehicles.* DSTO Technical Report, DSTO-TR-2014. Defence Science and Technology Organisation, Commonwealth of Australia.

6. Smith, N. S. A. 2008. *Enhancing capability in estimating the response of structures to hypersonic flow loads through the further development of the TempEst code.* DSTO Technical Report, DSTO-TR-2237. Defence Science and Technology Organisation, Commonwealth of Australia.

7. *Abaqus Version 6.7 On-line Documentation*, Dassault Systemes, United States of America, 2007.

8. Anderson, J. D. 1989. *Hypersonic and High Temperature Gas Dynamics.* American Institute of Aeronautics and Astronautics Inc., Reston, Virginia, USA.

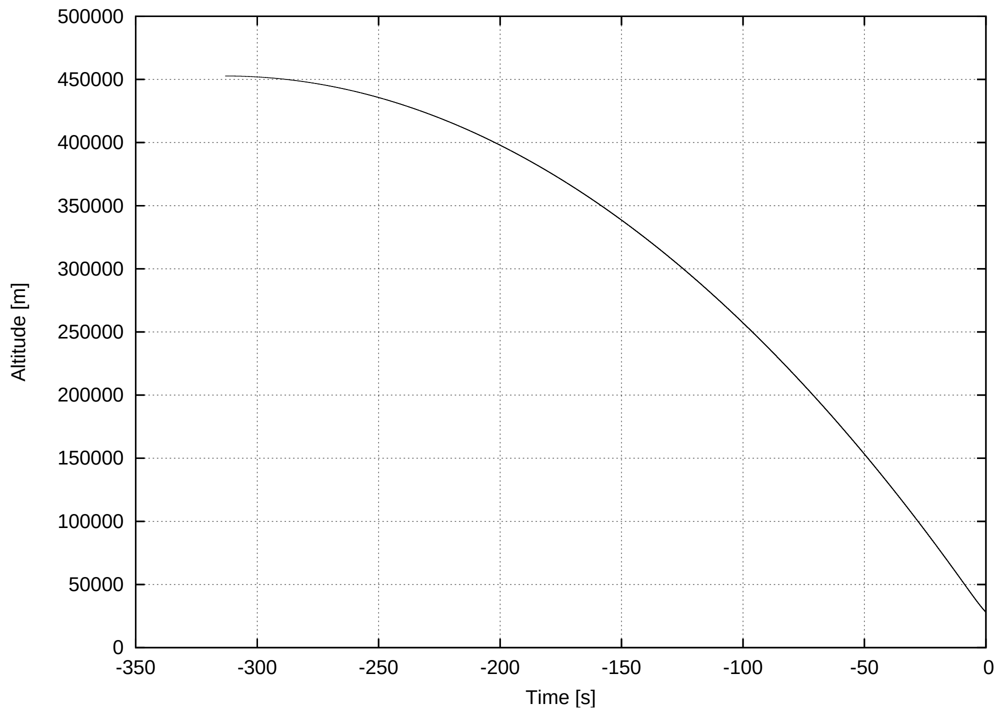

Figure A1 (U): *Altitude as a function of time before reaching 28 kilometres altitude during lifting-body re-rentry.*

Appendix A Hypersonic Re-entry Trajectory

The following trajectory was computed using *FlightPath* , a two degree-of-freedom (2-DOF) trajectory program written for controlled winged re-entry vehicles and bundled with the *TempEst* software release[5, 6]. It was computed by starting the vehicle at the desired target altitude and airspeed in each case and solving the equations of motion backwards in time. This was done subject to certain constraints on allowable angles of attack and lift-induced accelerations until apogee conditions were reached. Aerodynamic coefficients of lift and drag were taken from a look-up table that had been populated by inviscid CFD results for the vehicle design at differing Mach numbers, angles of attack and altitudes (atmospheric densities).

Figure A1 is a plot of altitude as a function of time before the target condition is reached. In this case the target condition is Mach 8 flight at 28 kilometres altitude with a downward flight path angle of 60 degrees. It is evident that it takes some 300 seconds from sub-orbital apogee for the lifting body to achieve the target condition.

Figure A2 is a plot of vehicle air speed as a function altitude below 100 km. The trajectory shows variation in air speed above 45 km due to acceleration due to gravity. It is apparent that the pull-out in the trajectory leads to a great reduction in air speed. It is evident that the relatively high angle of attack reached by the lifting body during the pull-out attracts a substantial drag penalty. The dynamic pressure experienced by the re-entry vehicle as a result of decreasing altitude and air speed is plotted in Figure A3.

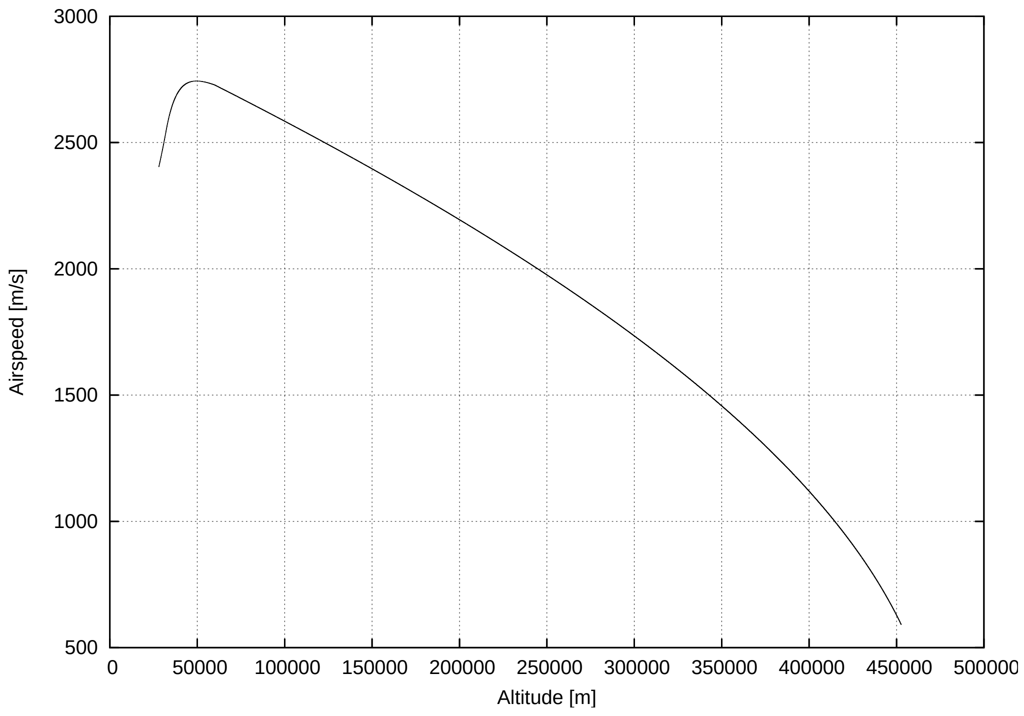

Figure A2 (U): *Air speed as a function of altitude during lifting-body re-rentry.*

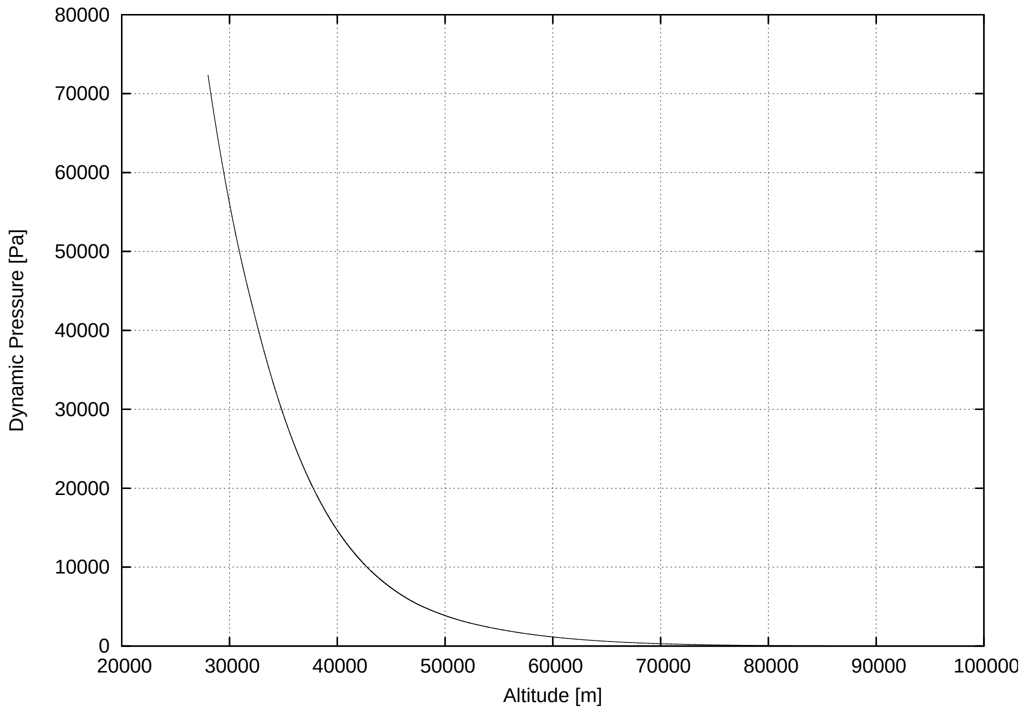

Figure A3 (U): *Dynamic pressure as a function of altitude during lifting-body re-rentry.*

Appendix B *HyPAq* User Guide

```
===============================================================================

                                  _--_|\
                                 /      \
                                (  DSTO  )
                                 \_.--,_/
                                       v

    __  __      ____  ___          =======================================
   / / / /_  __/ __ \/   | ____ _  =======================================
  / /_/ / / / / /_/ / /| |/ __ `/  =======================================
 / __  / /_/ / ____/ ___ / /_/ /   =======================================
/_/ /_/\__, /_/   /_/  |_\__, /    =======================================
      /____/               /_/     =======================================

Hypersonics Package for use with Abaqus (version 0.90)

===============================================================================


            "A   U S E R   G U I D E   T O   H Y P A Q   v0.90"

                              Nigel Smith

                              October 2008


SECTION 1  INTRODUCTION AND DISCLAIMER=======================================

SECTION 1.1 INTRODUCTION AND CONTENTS-----------------------------------------

This documentation is meant to provide the prospective user with an
introductory guide in how to set-up and compute hypersonic flow and heating
problems using HyPAq v0.90 in conjunction with the Abaqus/CAE and Abaqus
Standard (version 6.7) software packages written by Abaqus Inc. for
structural analysis.

HyPAq provides convective film and surface pressure data to Abaqus during
execution at any arbitrary time step in order to satisfy user-defined
boundary conditions (in the case of pressure) and interfaces (in the case
of convective films).  This data is interpolated in space and time
from user-provided CFD data of general format. Further, HyPAq corrects
convective film heat fluxes for differences between interpolated and
```

computed flow-exposed surface temperatures. In practice, all the user need do to use HyPAq is to:

a) set up their Abaqus model (usually in Abaqus/CAE) with user-defined distributed load (pressure) boundaries and convective film interfaces,
b) run an Abaqus coupled thermo-structural transient case in batch mode with the HyPAq routines compiled-in,
c) take the resultant mesh point output file (hyper_meshpt.txt) generated by Abaqus/HyPAq,
d) run the HyPAq interpolator (intphyper) on the mesh point file with CFD-derived data in HyPAq-native format (provided by parsers),
e) re-start the Abaqus/HyPAq job again in batch mode with the interpolation output file (intp_outdata.txt) present in the run directory, and
f) analyse Abaqus solution data using Abaqus/CAE or Abaqus/Viewer

HyPAq consists of a series of Fortan-77 user subroutines, that are meant to be compiled with the Abaqus package, a standalone Fortran-77 interpolator program (intphyper.f) that generates input files to be used by the subroutines, and a series of standalone Fortran utility programs that reformat common computational fluid dynamic data formats into input for the interpolator.

The sections contained in this guide include descriptive guides of the necessary input files required by HyPAq programs in order to run and some illustrative worked examples of Abaqus calculations made using HyPAq functionality that are included in the software distrbution. It is assumed that the prospective HyPAq user is already familiar with Abaqus/CAE and Abaqus/Standard. Abaqus software comes with extensive high quality on-line documentation which the user should read to fully understand what functionality HyPAq provides to Abaqus.

*NOTE: In using HyPAq, all units used in the establishment of Abaqus models must be strictly S.I.; all lengths must be in METRES, all masses in KILOGRAMS, all times in SECONDS, all temperatures must be in KELVINS, and so on.

SECTION 1.1.1 CONTENTS- -

```
Section 3  HyPAq Interpolator (intphyper.f)
   Section 3.1 HyPAq-Native Input File Format
   Section 3.2  Standard Input Structure

Section 4  CFD Data Parsing

Section 5  Abaqus/HyPAq worked example



SECTION 1.2 COPYRIGHT NOTICE AND DISCLAIMER FOR HYPAQ-----------------------

!==============================================================!
!                                                              !
! COPYRIGHT held by the Commonwealth of Australia, 2008        !
!                                                              !
! THE DISTRIBUTION AND/OR USE OF THIS SOFTWARE IS STRICTLY     !
! PROHIBITED IN THE ABSENCE OF THE EXPRESS WRITTEN CONSENT     !
! OF THE AUTHOR OR THE CHIEF DEFENCE SCIENTIST OF THE          !
! DEFENCE SCIENCE & TECHNOLOGY ORGANISATION (DSTO) OF          !
! AUSTRALIA.                                                   !
!                                                              !
!                                                              !
! AUTHOR:                                                      !
!                                                              !
! Dr Nigel S. A. Smith                                         !
!                                                              !
!                                                              !
! Air Vehicles Division                                        !
! Defence Science and Technology Organisation                  !
! 506 Lorimer Street                                           !
! Fishermans Bend, VIC 3207                                    !
! AUSTRALIA                                                    !
!                                                              !
! nigel.smith@defence.gov.au                                   !
!                                                              !
!                                                              !
! DISCLAIMER                                                   !
!                                                              !
! THIS SOFTWARE WAS PREPARED AS A RESULT OF WORK SPONSORED     !
! BY THE GOVERNMENT OF THE COMMONWEALTH OF AUSTRALIA. NEITHER  !
! THE COMMONWEALTH NOR THE AUSTRALIAN DEPARTMENT OF DEFENCE,   !
! NOR ANY OF THEIR EMPLOYEES, NOR ANY OF THEIR CONTRACTORS,    !
! SUBCONTRACTORS, OR THEIR EMPLOYEES, MAKES ANY WARRANTY,      !
! EXPRESS OR IMPLIED, OR ASSUMES ANY LEGAL LIABILITY OR        !
! RESPONSIBILITY FOR THE ACCURACY, COMPLETENESS OR USEFULNESS  !
! OF ANY INFORMATION, APPARATUS, PRODUCT OR PROCESS DISCLOSED, !
! OR REPRESENTS THAT ITS USE WOULD NOT INFRINGE PRIVATELY      !
! OWNED RIGHTS.                                                !
!                                                              !
!==============================================================!
```

SECTION 2 HYPAQ AND ABAQUS==

In this section, the relationship between the HyPAq routines and their host code Abaqus is described. Each of the HyPAq routines is discussed in terms of its function as an Abaqus User Subroutine.

SECTION 2.1 ABAQUS USER SUBROUTINES THAT MAKE UP HYPAQ------------------------

The following subroutines comprise the user subroutine portion of HyPAq given in terms of their Abaqus name and HyPAq file name. The argument lists and requirements of each can be found in the Abaqus online documentation. The details of the routines functions which are unique to HyPAq are provided here.

UEXTERNALDB (hyperIO.f)---------
This routine is called by Abaqus at the start and end of computational steps and the start and end of complete analyses during Abaqus execution. HyPAq uses Abaqus calls to this routine to input and output interpolation information to external files. At start-up, this routine searches the run directory for a file named 'intp_outdata.txt'. If it finds this file it reads in its contents and sets a flag in the internal data structure so that this data is used to provide interpolated convective and pressure boundary conditions to Abaqus (see FILM and DLOAD). If it does not find 'intp_outdata.txt' then it sets an internal flag that causes FILM and DLOAD to record all physical locations for which they are invoked by Abaqus in the course of a single step. At output, if 'intp_outdata.txt' was not present, UEXTERNALDB outputs the stored physical locations and indexing information, for where convective and pressure boundary conditions were requested by Abaqus, to 'hyper_meshpt.txt'. If the file 'intp_outdata.txt' was present, then at output UEXTERNALDB outputs the most recent interpolated values in the last step for potential error checking and post-analysis. Note that 'intp_outdata.txt' is generated by the HyPAq standalone interpolator program intphyper.f using the UEXTERNALDB-generated file 'hyper_meshpt.txt' and reformatted CFD data sets (see Section 3).

FILM (hyperfilm.f)-----------
This routine provides convective film conditions at discrete locations as called by Abaqus. When interpolation data has been input, this routine performs an interpolation in time between stored convective information

at bracketing time stations for the referenced physical location (already interpolated in space by intphyper.f). The routine also accounts for the disparity between the stored surface temperature and the temperature provided by Abaqus in the subroutine call. FILM adjusts the convective film conditions (heat transfer coefficient and effective sink temperature) to account for the effect of the wall temperature difference:

```
sinkT = wall_temp * recovery_enthalpy' / wall_enthalpy

hcoef = hcoef' * ( wall_temp' / wall_enthalpy' )
      * ( wall_enthalpy / wall_temp )
      * sqrt[ ( ref_temp' * ref_visc ) / ( ref_temp * ref_visc' ) ]
```

All terms carrying a ' are time-interpolated stored values and all else is determined from Abaqus data provided in the call. Viscosities, various enthalpies and time interpolation factors are determined by HyPAq auxilliary routines (see Section 2.3).

The above correction is based on a reference enthalpy concept of heat transfer, thereby taking account of non-linearities in specific heats of gases with temperature. The correction includes the effects of wall temperature on local fluid density and viscosity.

Where no interpolation data has been read, FILM stores the physical location, element number and interpolation point number for each call so that this information can be used to generate interpolation data for a subsequent restart of the same run.

DLOAD (hyperload.f)-----------
This routine provides surface pressure values interpolated in time in the same way as FILM except no correction for surface temperature is appropriate or required. As with FILM, where no interpolation data is available, DLOAD also records the locations and indexing of all calls to it made by Abaqus to be later output for use by intphyper.f in generating spatial interpolation data.

*NOTE: In using HyPAq, all units used in the establishment of Abaqus models must be strictly S.I.; all lengths must be in METRES, all masses in KILOGRAMS, all times in SECONDS, all temperatures must be in KELVINS, and so on.

SECTION 2.2 AUXILLIARY HYPAQ ROUTINES---

Two additional Fortran-77 files of subroutines are included with the above routines in the bundle that is compiled with Abaqus. They are:

hypertime.f------------
This routine is passed a time argument by FILM and DLOAD and determines what stored time data points bracket that time and what interpolation factors to apply to the pressure and convective data at each stored time point.

hypercurv.f------------
This file contains three routines which contain polynomial fits for the enthalpy and dynamic viscosity of air given temperature, and the temperature of air given its enthalpy. These data are required for the temperature correction process described for FILM (Section 2.2).

SECTION 3 HYPAQ INTERPOLATOR (intphyper.f)=====================================

The HyPAq interpolator (intphyper.f) generates the interpolation data file 'intp_outdata.txt' from a series of input files. This output file is sought at every run of Abaqus/HyPAq. If it is found then, a normal run will proceed. If it is not found, the Abaqus/HyPAq will run for a single step and then output a file called 'hyper_meshpt.txt' which contains the locations of all points defined by Abaqus as requiring pressure and/or convective film information.

The interpolator reads in 'hyper_meshpt.txt' and a series of data files which have been 'parsed' from their native CFD source format into a simple HyPAq-native format, with one time point per file. These parsed data files are stored in a sub-directory of the Abaqus run directory called 'intp_files/' and have the following naming conventions where XXXX is a four digit time index :

pressure file	intp_prsXXXX.txt
heat flux file	intp_qdtXXXX.txt
wall temperature file	intp_wtpXXXX.txt
total temperature* file	intp_ttpXXXX.txt
static temperature* file	intp_stpXXXX.txt

[*Note that the total and static temperatures referred to here are for the flow adjacent to the surface but outside the boundary layer. These values are used to apply a recovery enthalpy correction to heat fluxes where the Abaqus-computed wall temperature deviates from the interpolated wall temperature as described in Section 2.1. Section 4 describes how these static and total temperature values are typically found.]

```
Each of the above file-types contains a data field defined at a single point
in time and at any number of points in space defined in (x,y,z) coordinates.
Section 3.1 describes the HyPAq-native format required of these files.
The HyPAq interpolator reads in all of these files along with some timing
information from standard input and interpolates all of the above data fields
spatially onto the locations provided by Abaqus/HyPAq in 'hyper_meshpt.txt'
for each time point.  The spatial disposition of points within each of the
above files need not agree.

The spatial interpolation is carried out using an unstructured 'cloud weighting'
scheme.  In this scheme, for each Abaqus designated location (x_pt,y_pt,z_pt)
different subsets of the complete set of CFD-designated points are examined.

For the j-th subset, which has N_j member points, each point (i = 1, N_j)
has an inverse square distance weight generated:

   dx2_i = ( x_i - x_pt )^2
   dy2_i = ( y_i - y_pt )^2
   dz2_i = ( z_i - z_pt )^2

   dist_i = max[1.e-20, sqrt( dx2_i + dy2_i + dz2_i )]

   weight_i = 1 / [ dist_i * summation_i=1,N_j { 1/dist_i } ]

Along with the weights for all points in the j-th subset, interpolated data
for the Abaqus designated point is given by:

   f_intp = summation_i=1,N_j { weight_i * f_i }

However, not all subsets provide a good approximation. For the best results
the subset must include CFD-points which bracket the Abaqus-designated points.
It turns out that this can be checked using the original Abaqus points
(x_pt,y_pt,z_pt), the subset members, their weights, and locations.  Whichever
subset gives the minimum interpolated spatial displacement (displ_j) is the
best subset.  Typically, if this displacement is zero then the point is
bracketted, else it is bracketted in a plane that is displaced in an orthogonal
dimension.

   x_chk_j = summation_i=1,N_j { weight_i * x_i }
   y_chk_j = summation_i=1,N_j { weight_i * y_i }
   z_chk_j = summation_i=1,N_j { weight_i * z_i }

   displ_j = sqrt( (x_chk_j - x_pt)^2
           + (y_chk_j - y_pt)^2
           + (z_chk_j - z_pt)^2 )
```

```
It is possible that more than one subset will give the same minimum
displacement, in which case the subset with smallest mean distance
from the Abaqus-designated point is selected from these 'tied' subsets.

The above selection process requires that the user choose the size of
N_j (N_j = N, a constant) and M.  The M closest CFD-designated points to the
Abaqus-designated points are used as the pool from which N_j sized subsets
are chosen and evaluated in the above process.

The user should select N so that it provides enough CFD-designated points
to properly bracket an Abaqus-designated point.  This relates to the structure
and dimensionality of the CFD-designated object. For example, for a two
dimensional CFD object surface consisting of quadrilaterals then N should
equal 4 to properly bracket any given point.  For a surface consisting of
triangles, then N should equal 3.  For one-dimensional line boundaries on
two dimensional objects, then N should equal 2.  Interpolation for three
dimensional spaces can be undertaken in principle but are not required for
use in this context where only one and two dimensional surfaces are
possible for two and three dimensional objects.

        +----------+
        |         /
        |   o    /      +     +-------o--+
        |       /      / \
        |      /      /   \
        +-----+      /o    \
                    +-------+

In choosing the 'pool-size' M, the user must consider that the selection
process will examine M-choose-N combinations to find the optimal subset
of CFD-designated points for *each* Abaqus designated location.  This can
be very costly if M is a large number.  On the other hand, some CFD-designated
geometries with highly non-uniform spacing can require quite distant points
to adequately bracket some Abaqus points (see below).


       +-+-+-+-+-+-+-+------------------------------------------------+
       | | | | | | | | o                                              |
       +-+-+-+-+-+-+-+------------------------------------------------+

In the above poorly spaced CFD-mesh, if M is less than 18 then the Abaqus
point ('o') will fail to be bracketted.  It is clear that the selection of
M and N requires careful thought on the part of the user and potentially
more than one attempted run with intphyper.f to get a good interpolation
solution everywhere.

To assist in this process displacement-check output files are written for
```

```
each variable and time index according to the convention chkintp_prsXXXX.txt
for pressure (and so on) into the intp_files directory.  These text files
contain a list of Abaqus-designated points written as x_pt,y_pt,z_pt followed
by the interpolation displacement (displ_j) calculated for the supposed
optimal subset and then this value normalised by the CFD-designated mean point
spacing near that location.  This last column provides the most useful
measure. All points should have a relative displacement of 0.001 or less,
but preferrably close to machine zero.

Since CFD-designated files are free to have completely different geometric
dispositions from one another, it is necessary for the user to specify
distinct M and N values for each such file.  Typically, the data 'parser'
programs (see Section 4) available with the HyPAq software distribution
prompt the user for these values as they reformat CFD data from different
sources into the HyPAq-native format required by the intphyper.f program.

To avoid interpolation difficulties the surface of the CFD-defined object
should nominally agree with the Abaqus-defined surface, while surface
discretisations will naturally vary.  In general CFD surface meshes which
are finer that the Abaqus-surface mesh lead to better interpolation outcomes.
Abrupt changes in surface discretisation density of the CFD-defined object
should be avoided where possible.


SECTION 3.1 HYPAQ-NATIVE INPUT FILE FORMAT------------------------------------

The format of the text files (intp_prsXXXX.txt etc) described in the preceding
section is known as HyPAq-native and conforms to the following structure:

Line 1:

  N [integer], M [integer]

    N is the subset size described in the preceding section.

    M is the 'selection pool' size described in the preceding section.
    Note that M must be greater than or equal to N.

Line 2:

  NCFDD [integer]

    NCFDD is the number of CFD-designated data points contained in the
    file.

Line 3:
  IX1 [integer], X1OFF [real], X1SCAL [real]

    IX1 is the index of the entry in each data line of the file which
    which corresponds to the X1 coordinate in the Abaqus model.  The
    specification of these IX1, IX2, and IX3 indices allows coordinate
    axes to be switched between the CFD-designation of the modelled
    surface and the Abaqus-designation of the same model.

    X1OFF is the offset in metres between what is read in the file and
    what will be used as the X1 coordinate value by intphyper.f.

    X1SCAL is a scale factor in between what is read in the file and
    what will be used as the X1 coordinate value by intphyper.f.

       eg. X1(I) = DATA_LINE(IX1,I) * X1SCAL + X1OFF

Line 4:

  IX2 [integer], X2OFF [real], X2SCAL [real]

     As for Line 3 definitions above except they pertain to X2 the
     coordinate.

Line 5:

  IX3 [integer], X3OFF [real], X3SCAL [real]

     As for Line 3 definitions above except they pertain to X3 the
     coordinate.

Line 6:

  IDVAR [integer], DVOFF [real], DVSCAL [real]

     As for Line 3 definitions above except they pertain to the
     dependent variable that is the subject of the file.  In this
     instance, DVOFF has units relevant to the dependent variable
     rather than metres.


Line 7 to 6+NCFDD :

  DATLINE(1), ... , DATLINE(L) [reals]

     DATLINE is an array with L elements.  These elements can contain
     any number of different variables, but only those nominated using
     IX1, IX2, IX3, and IDVAR (as described above) are used by the
```

```
          interpolator.  Obviously, it is an error for any of these entry
          indices to be less than 1 or greater than L.  It is also an error
          for any two indices to have the same value.  Thus L must be greater
          than or equal to four.

          Note that there need not be any structure to the arrangement of
          data points in this file since the HyPAq interpolator functions
          independently of mesh structure.

     End of File


     SECTION 3.2 STANDARD INPUT STRUCTURE------------------------------------------

     The interpolator (intphyper.f) reads time point information from standard
     input.  Typically, this information is set out in an instruction file and
     piped into standard input at run time.  This file must have the following
     structure:

     Line 1:

      KNFTIM [integer], TMOFF [real], TMSCAL [real]

         KNFTIM is the number of discrete time points for which CFD
         data sets exist (ie. how large does the index XXXX in the
         intp_prsXXXX.txt etc. files get?)

         TMOFF is an offset in seconds between what is read in the
         subsequent time point listing and what intphyper.f should
         record as the time for each point.

         TMSCAL is a scale factor in between what is read in the
         subsequent time point listing and what intphyper.f should
         record as the time for each point.

            eg. time_value(i) = KFTIME(i) * TMSCAL + TMOFF

         **Note that the CFD data set files (intp_qdt0001.txt etc) do
         not contain time information.  That information is specified
         in this input stream.

     Line 2 to KNFTIM+1 :

       KFTIME(i) [real]

          KFTIME(i) is the i-th time point entry in a list of KNFTIM entries
```

```
Line KNFTIM+2 :

  KNITIM [integer]

     KNITIM is the number of non-linear interpolation 'function' points.
     If this greater than zero, then the interpolator will read in data
     which will allow non-linear interpolation to be undertaken between
     adjacent data sets in time.  This is of value where data sets are
     sparse in time and simple linear interpolation based on time between
     them is inappropriate.  For example during a flight trajectory, one
     might only have 4 data sets at widely spaced altitudes.  In that case
     it would make more sense to linearly interpolate between sets based
     on altitude or dynamic pressure.  This functionality allows altitude or
     dynamic pressure or any other salient variable to be tabulated to allow
     this to occur.

If KNITIM is zero, then no further input is read.  Otherwise:

Line KNFTIM+3 :

  KITIM0 [real], KDITIM [real]

     KITIM0 is the start time in seconds of the KNITIM regularly spaced
     interpolation function points.

     KDITIM is the constant time step in seconds between the KNITIM
     interpolation function points.

Line KNFTIM+4 to KNFTIM+4+KNITIM :

     KITVAR(i) [real]

        KITVAR is the i-th real value entry of the selected interpolation
        function that consists of KNITIM regularly spaced entries.

End of File


SECTION 4 CFD DATA PARSING==================================================

The HyPAq-native format described in Section 3.1 is quite flexible so that,
in many cases, ordinary text-formatted output from CFD codes for surfaces can
simply have the six descriptive header lines added and be used directly.

In some cases this is not possible, and some automated 'parsing' of the CFD
data is required to render it into HyPAq-native form.  Thus far two parsers
are included in the default HyPAq distribution; one for TempEst and one for
```

CHYMERA. It is planned to introduce a CFD++ parser in the near future.

However, the task of parsing CFD data into HyPAq-native is not a particularly onerous one and the user is encouraged to write their own code-fragment to carry out the reformatting for their choice of CFD tool.

This section provides some guidance as to how to go about doing this in general terms. First let us examine the requisite HyPAq-native files (also see Section 3) :

pressure file	intp_prsXXXX.txt
heat flux file	intp_qdtXXXX.txt
wall temperature file	intp_wtpXXXX.txt
total temperature file	intp_ttpXXXX.txt
static temperature file	intp_stpXXXX.txt

Recall that each of the above files corresponds to a single variable definition at a single time (index XXXX). These files need not have the same geometric arrangement between times or even between variables at the same time. It is necessary for all time indices to be contiguous and start from 0001. No more than 9999 time entries are supported.

The parsing of CFD-defined surface data for pressure, heat flux, and wall temperature should not require anything more than simple reformatting.

Parsing static and total temperature requires a level of understanding as to what purpose these variables are to be put. A reference enthalpy method is applied to modify heat fluxes at run time where the Abaqus-computed wall flux varies from the values interpolated from those defined in the files intp_wtpXXXX.txt (see Section 2.1). The interpolated total and static temperatures are used with the interpolated and Abaqus-computed wall temperatures to evaluate reference enthalpies and reference temperatures. Ratios of reference values derived from Abaqus-computed and interpolated sources are used to modify the convective heat flux coefficient used at each Abaqus time step.

To this end, the static and total temperatures need to be representative of the flow at the edge of the boundary layer. Obviously, the static and total temperatures are equal right at any viscous surface. Some CFD methods solve for an inviscid flow and then apply approximate heat transfer correlations after the fact (eg. TempEst). In this instance, total and static temperatures can be appropriately defined at the surfaces themselves and again the problem of parsing is reduced to simple reformatting. However,

the majority of CFD codes for predicting heat transfer will be viscous and
a more sophisticated approach is required.

In these instances it is recommended that the static and total temperature
information by extracted from the cell which is one cell orthogonal to the
surface for each surface location. This is easily achieved in structured
CFD meshes by exploiting the regular indexing arrangement of cells. For
unstructured CFD meshes, distances need to be computed and used to reject
fluid cells too close to each surface cell.

The flexibility of the HyPAq interpolator can be fully employed to reduce the
complexity of the parsing task for these variables. There is no reason why
entire (not just surface) CFD computational fields cannot be exported for
static and total temperatures minus the surface locations. The interpolator
will find the best CFD-point subsets for each Abaqus-designated surface
location. In this case, since no CFD-surface information will be included,
the interpolator will not be able to bracket each Abaqus-surface location
(see Section 3) but will have an orthogonal offset. This approach is
employed in the present CHYMERA data parser.

SECTION 5 ABAQUS/HYPAQ WORKED EXAMPLE===

The following example relates to the heating on re-entry of a nose structure
for a flight vehicle. It relies on TempEst version 1.16 for heating data
input. This example is for a two-dimensional model. Three-dimensional
models require precisely the same methodology.

1. To begin, copy the HyPAq TEMPLATE run directory $HOME/hypaq/cases/TEMPLATE
to another name : cp -r $HOME/hypaq/cases/TEMPLATE $HOME/hypaq/cases/myexample
while keeping the completed example ($HOME/hypaq/cases/EXAMPLE) as a
reference.

2. Enter the new directory : cd $HOME/hypaq/cases/myexample

3. The user must now create a two-dimensional Abaqus model in Abaqus/CAE.

*NOTE: In using HyPAq, all units used in the establishment of Abaqus models
must be strictly S.I.; all lengths must be in METRES, all masses in
KILOGRAMS, all times in SECONDS, all temperatures must be in KELVINS,
and so on.

The model should be a four sided (straight line) figure with the
following vertices (not to scale, lengths in metres) :

```
   ^ y
   |
|
---> x

           +(0.0,0.0)                               +(0.12095,0.0)


           +(0.0,-0.002)

                                                    +(0.12095,-0.013284)
```

The model should be established for a coupled temperature-displacement
solution. The following surface designation system is to be used
henceforth:

```
^ y
|
|
---> x
                             Surface 2
           +                                        +
 Surface 1
                                                     Surface 4
           +
                          Surface 3
                                                    +
```

Note that surfaces 1, 2, and 3 should be set up as having user
defined convective film interactions. Surface 4 should be set
up as an 'encastre' adiabatic boundary. Surfaces 1, 2 and 3
should also be established as a user-defined distributed load
boundaries.

A structured mesh should be established with 8 cells on surfaces
1 and 4, and 122 cells on surfaces 2 and 3.

The material used should be Copper with the following temperature
invariant properties. Temperature dependent properties can, and should,
be entered but the following will suffice for this illustration.

```
Thermal Conductivity = 401 W/(m K)
Specific Heat = 400 J/(kg K)
Density = 8960 kg/(cubic m)
Youngs Modulus = 1.1E11 Pa
Poisson's Ratio = 0.34
Expansion Coefficient = 1.65E-5
```

```
   The coupled temperature-displacement case should be set to
   run in a transient mode with a real time duration of 23 seconds.
   The initial time step should be 0.1 seconds, with a maximum
   step of 1 second and a minimum of 1.e-5 seconds.  The maximum
   number of steps should be 10000.  The maximum allowable
   temperature change per step should be 1 degree (Kelvin).

   All initial temperatures should be set to 300 Kelvins.

   The completed model should be output as a job named 'wedge'.

   Exit Abaqus/CAE

4. Copy the Abaqus generated input file (wedge.inp) into the
   default input name (0test.inp) so that the automatic scripts
   can access it.

5. Run Abaqus (to generate geometric data) using the script provided:

   ./.run

6. After Abaqus has completed its run (no more than 5-10 seconds),
   examine the output text files 'test.log' and 'test.msg'.

   The end of 'test.log' will contain the following message :

Abaqus Error: Abaqus/Standard Analysis exited with an error - Please see the
message file for possible error messages if the file exists.
Abaqus/Analysis exited with errors

   This is expected since HyPAq has called an interruption
   to the normal Abaqus analysis process.  There will be
   a message to this effect at the end of 'test.msg' :

 ***NOTE: UEXTERNALDB: =====NORMAL HYPERPACK HALT CALLED============

 ***NOTE: UEXTERNALDB: =====USER MUST EXECUTE INTPHYPER============

 ***NOTE: UEXTERNALDB: =====THEN RE-RUN ABAQUS FROM START============

 ***ERROR: Analysis is being terminated from a user subroutine


7. Delete any files in $HOME/hypaq/cases/myexample/intp_files.

8. Copy the interpolator input files from the parsed TempEst heating
```

case from the EXAMPLE directory ie:

```
cp $HOME/hypaq/cases/EXAMPLE/intp_files/intp* intp_files/
```

Examination of these files (see Section 3.1 for their structure) will reveal that the interpolator is being instructed to find the two best fit points out of the 20 closest in the data file (M=20, N=2).

9. Copy the corresponding instruction file to the interpolator (1ns_intp.txt) from the EXAMPLE directory ensure (the meaning of the entries is described in Section 3.2) :

```
cp $HOME/hypaq/cases/EXAMPLE/1ns_intp.txt .
```

10. Run the HyPAq interpolator using the script provided:

```
./.runint
```

11. Examine interpolator log file '1og_intp.txt' for output messages. A nominally successful outcome will have the following on its last line:

```
Done writing OUTPUT file : intp_outdata.txt
```

The user may also wish to examine the interpolator check files generated in the sub-directory intp_files (see Section 3). At present, there is no means to examine the interpolated data fields themselves in Abaqus/CAE or Abaqus/Viewer, so plotting the check files and looking for large displacement errors is the only point-by-point check available to the user. A means of having these fields plotted by Abaqus is a subject of future development.

12. Clear the run directory of the results files from the previous Abaqus run:

```
./.clear
```

13. Run Abaqus to complete the transient analysis using the script provided:

```
./.run
```

14. The run should terminate after approximately 250 CPU seconds. Examination of the 'test.log' and 'test.msg' files will reveal that the job completed normally. Abaqus Viewer can be used to examine the resultant structural solution over all the time steps by opening the database file 'test.odb'. The viewer can be used to probe the solution and output data for subsequent use.

```
15. All solution and run files can be cleared by typing ./.clearall

Outputs from this and other worked examples are provided in the technical
report accompanying this manual.


================================END OF FILE===================================
```

Page classification: UNCLASSIFIED

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION DOCUMENT CONTROL DATA	1. CAVEAT/PRIVACY MARKING
2. TITLE *HyPAq*: Software for the transient analysis of hypersonic vehicles in *Abaqus*	3. SECURITY CLASSIFICATION Document (U) Title (U) Abstract (U)
4. AUTHOR Nigel S. A. Smith	5. CORPORATE AUTHOR Defence Science and Technology Organisation 506 Lorimer St, Fishermans Bend, Victoria 3207, Australia

6a. DSTO NUMBER DSTO–TR–2236	6b. AR NUMBER AR-014-370	6c. TYPE OF REPORT Technical Report	7. DOCUMENT DATE October 2008

8. FILE NUMBER 2008/1101488/1	9. TASK NUMBER LRR 07/250	10. SPONSOR DSTO	11. No OF PAGES 0	12. No OF REFS 8

13. URL OF ELECTRONIC VERSION http://www.dsto.defence.gov.au/corporate/reports/DSTO–TR–2236.pdf	14. RELEASE AUTHORITY Chief, Air Vehicles Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved For Public Release

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS

No Limitations

18. DSTO RESEARCH LIBRARY THESAURUS

Hypersonics
Aerothermodynamics Thermostructural analysis

19. ABSTRACT

The Defence Science and Technology Organisation (DSTO) has embarked upon a research and development program for airbreathing hypersonic vehicles. Part of this program involves the selection and development of software tools useful in the aerothermo-structural design of these vehicles. *HyPAq* is a hypersonic package for *Abaqus* – a set of programs and subroutines to be used in conjunction with that particular finite element analysis code. *HyPAq* interpolates pressures and convective heat fluxes derived from any arbitrary set of CFD results onto *Abaqus* transient coupled temperature-displacement analyses. This report describes the algorithms which underpin *HyPAq* and provides an example set of results. The example results are from an analysis which aims to determine a suitable heat-sink material for the nose section of a lifting body undergoing Mach 8 sub-orbital re-entry. The complete user guide provided with the *HyPAq* software distribution is included in an appendix.

Page classification: UNCLASSIFIED