14th AIAA/AHI Space Planes and Hypersonic Systems and Technologies Conference
Nov 6–9, 2006/Canberra, Australia

# High-Speed MHD Flow Control using Adjoint-Based Sensitivities

Andre C. Marta*, Juan J. Alonso†
*Stanford University, Stanford, CA 94305*

**As advances in integrated hypersonic flight vehicles materialize, it is worth devoting some effort to optimize their design. The plasma generated at high Mach numbers permits the use of electromagnetic actuators to control the flow. One can view this situation as an optimization problem in which a cost function is to be minimized by varying a set of control variables, while satisfying a specific set of constraints. The flow under the presence of a magnetic field is typically modeled with the equations of magneto-hydrodynamics (MHD) and the scientific community has already devoted much effort into their analysis. The focus of this work is to tackle the problem of efficiently estimating the sensitivity of cost functions and/or constraints with the use potential of a gradient-based optimizer in mind. This paper outlines the implementation of a scalable tool to compute sensitivities of cost functions that arise in such optimization problems. The sensitivities are computed using the discrete adjoint method that has already been successfully tested by the authors for this class of problems. The derivation of the discrete adjoint equations is done with the help of Tapenade, an automatic differentiation (AD) tool, that only requires some minor re-writing of the original flow solver code but avoids the tedious hand-differentiation approach. This strategy has proved to be much faster to implement and much less error-prone than a hand-differentiation approach and the resulting sensitivities are in exact agreement with the discrete sensitivities that would be produced by exact finite-differences of the original code. The linear adjoint system of equations obtained is solved using the Portable, Extensible Toolkit for Scientific computation (PETSc). The low magnetic Reynolds number MHD governing equations are used to model the flow but this methodology can be transparently applied to any set of governing equations, cost functions and design variables. Sensitivities of aerodynamic coefficients such as lift, drag and moment, with respect to the electrical conductivity of the flow are computed in a multi-block grid using this approach and their values are verified, totaling more than 180,000 design variables.**

## I. Introduction

Recent years have brought significant advances in hypersonic flight, in particular with the successful flight test of the NASA X-43,[1] a hypersonic scramjet-powered research aircraft. More recently, the Air Force, Pratt & Whitney Rocketdyne and NASA successfully tested the hypersonic Ground Demonstration Engine (GDE-2)[2] using hydrocarbon fuels and a thermally-balanced setup. DARPA is proceeding along with the Falcon program[3] and a number of other efforts are being persued at NASA and abroad. There are still many technical and scientific obstacles to overcome but the community is getting closer and closer to the stage where hypersonic flight may be possible. With that in mind, it is now the time to start looking at ways to optimize such designs in an efficient manner.

When air flows at hypersonic speeds around vehicles, strong shock waves emerge in the regions of intense flow deceleration. There, much of the mean flow kinetic energy is converted into internal energy which causes the temperature of the air to increase dramatically. In turn, this temperature increase leads to

*Doctoral Candidate, AIAA Member
†Associate Professor, AIAA Member

the dissociation and ionization of the air and plasma is generated. Since the medium can then be locally polarized, the possibility opens up to use electromagnetic actuators to optimally control the flow. Now we can view this problem as an optimization problem in which a cost function must be minimized by varying a set of control variables, while satisfying a specific set of constraints. One alternative to carry out these optimizations is to use a gradient-based non-linear optimizer.

To tackle this kind of optimization problem, two important conditions must be met: the flow must be accurately predicted and the sensitivities of the cost and constraint functions must be computed at reasonable expense, with respect to a significant number of parameters. As for the first condition, in order to model the flow, one must couple the flow governing equations (typically the Navier-Stokes equations) to the Maxwell equations, leading to a new set of equations called the magneto-hydrodynamic (MHD) equations. There are plenty of references regarding this, covering a large spectrum of model complexity, ranging from the low magnetic Reynolds number assumption,[4] the ideal MHD,[5] the full MHD,[6] the inclusion of turbulence models,[7] up to thermal and chemical non-equilibrium models.[8,9] The second condition poses the most interesting challenge for the authors since there has been some work to try to control hypersonic flow[10] but no true effort has appeared to pose this problem from the point of view of nonlinear programming.

In order to advance the use of optimization tools in hypersonic flow, the authors have prototyped a design framework which provides automated optimization capabilities using the control theory approach, also called the adjoint method. The control theory approach has been mathematically well documented[11,12] and the method has already been extensively applied to aerodynamic shape optimization of problems governed by the Euler[13] and Navier-Stokes equations.[14]

The authors have pioneered the extension of the discrete adjoint approach to the control of a hypersonic flow in the presence of magnetic fields, and successfully demonstrated its feasibility using both the inviscid low magnetic Reynolds number model[15] and the ideal MHD model.[16] The discrete adjoint approach was selected over the continuous alternative based on past experience with both methods. Tecause the former, being derived from the discretized form of the flow governing equations, produces gradients that are consistent with the flow solver. In addition, it can treat arbitrary cost functions and constraints and allows the use of automatic differentiation (AD) tools in its derivation. The development of continuous adjoint solvers requires long times and significant investments and a number of short cuts must be included when the governing equations (such as RANS with turbulence models and/or MHD) are complicated. However, our previous work was developed for a single-block, single-processor flow solver that cannot handle realistically-sized problems, and the derivation of the discrete adjoint equations was done by manually differentiating the flow solver code.

The goal of the present work is to apply the discrete adjoint theory to a massively parallel MHD flow solver and to make use of automatic differentiation tools to derive the corresponding adjoint equations. This approach to automate the derivation of the discrete adjoint equations follows closely the work recently presented by Martins et al.[17] Such automation, not only reduces the implementation time but also eliminates any negative effect that might arise when making approximations in hand differentiation, such as those described by Dwight.[18]

In the following sections we describe the various components of the discrete adjoint solver that we have created to compute sensitivities of cost functions in MHD design problems. We start with the description of the physical model, in particular the governing equations of the hypersonic flow under the influence of magnetic fields, and the discrete adjoint formulation. We then provide details of the implementation of both the flow and adjoint solvers. Next, a test case is shown to illustrate the steps that lead to the calculation of cost function adjoint-based sensitivities using a large set of design variables. This is followed by a verification of the results using sensitivity values obtained using the traditional finite-difference approach. Lastly, some remarks are made concerning steps to be taken in the future in order to incorporate the tool developed into an automatic design framework.

## II. Flow Governing Equations

The equations governing the 3-D flow of a compressible conducting fluid in a magnetic field are obtained by coupling the Navier-Stokes equations to the Maxwell equations.

If the environment of interest is characterized by a low magnetic Reynolds number, then the magnetic field induced by the current is much smaller than that imposed on the flow and therefore it can be neglected.[10] This way there is no need to solve the three induction equations in the governing equations and the electromagnetic

forces and energy show up as straightforward source terms in the Navier-Stokes equations. Furthermore, if the viscous effects and heat transfer are neglected, the Navier-Stokes equations can be simplified to the so-called Euler equations. In this case, the non-dimensional MHD equations governing the flow are, in conservation form

$$\frac{\partial \mathbf{W}}{\partial t} + \frac{\partial \mathbf{E}}{\partial x} + \frac{\partial \mathbf{F}}{\partial y} + \frac{\partial \mathbf{G}}{\partial z} = \mathbf{S}\,, \tag{1}$$

where $\mathbf{W}$ is the vector of conservative variables and $\mathbf{E}$, $\mathbf{F}$ and $\mathbf{G}$ are the inviscid fluxes in the x, y and z directions, respectively, defined as

$$\mathbf{W} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix}, \quad \mathbf{E} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ \rho H u \end{pmatrix}, \quad \mathbf{F} = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho vw \\ \rho H v \end{pmatrix} \quad \text{and} \quad \mathbf{G} = \begin{pmatrix} \rho w \\ \rho wu \\ \rho wv \\ \rho w^2 + p \\ \rho H w \end{pmatrix}, \tag{2}$$

where $\rho$ is the density, $u$, $v$ and $w$ are the Cartesian velocity components, $p$ is the static pressure and $H$ is the total enthalpy, which is related to the total energy by $H = E + \frac{p}{\rho}$.

The source term $\mathbf{S}$ includes the magnetic field terms and is given by

$$\mathbf{S} = \begin{pmatrix} 0 \\ Q\sigma\left[B_z\left(E_y + wB_x - uB_z\right) - B_y\left(E_z + uB_y - vB_x\right)\right] \\ Q\sigma\left[B_x\left(E_z + uB_y - vB_x\right) - B_z\left(E_x + vB_z - wB_y\right)\right] \\ Q\sigma\left[B_y\left(E_x + vB_z - wB_y\right) - B_x\left(E_y + wB_x - uB_z\right)\right] \\ Q\sigma\left[E_x\left(E_x + vB_z - wB_y\right) + E_y\left(E_y + wB_x - uB_z\right) + E_z\left(E_z + uB_y - vB_x\right)\right] \end{pmatrix}, \tag{3}$$

where $\mathbf{B}$ is the magnetic field, $\mathbf{E}$ is the electric field, $\sigma$ is the electrical conductivity and $Q$ is the magnetic interaction parameter defined as $Q = \frac{\sigma_{ref} B_{ref}^2 L_{ref}}{\rho_{ref} U_{ref}} = R_b Re_\sigma$. The other non-dimensional parameters found in this formulation are the magnetic force number $R_b$ and the magnetic Reynolds number $Re_\sigma$, given by

$$R_b = \frac{B_{ref}^2}{\rho_{ref} U_{ref}^2 \mu_{m_{ref}}} \quad \text{and} \quad Re_\sigma = \mu_{m_{ref}} \sigma_{ref} U_{ref} L_{ref} \quad , \tag{4}$$

being $\mu_{m_{ref}}$ the magnetic permeability of the medium.

In our work, the magnetic field $\mathbf{B}$ is imposed by placing a collection of hypothetical electric circuits which imposes a magnetic field on the flow. Each elementary circuit is thought to produce a dipole like magnetic field given by $\mathbf{B} = \frac{\mu_m m}{4\pi r^3}\left[2\cos\theta \mathbf{e}_r + \sin\theta \mathbf{e}_\theta\right]$, where $r$ and $\theta$ define the dipole orientation and $m$ is the dipole strength.

In previous work,[16] we have shown results using the ideal MHD equations. In this paper we are using the low magnetic Reynolds number approximation to simplify the development, but everything we are presenting can be directly applied to more refined versions of the MHD equations.

## III. Discrete Adjoint Formulation

The control theory approach has been used extensively in the last years for both aerodynamic shape optimization[14,19] and aero-structural design[13,20] and has proven successful in MHD design.[15,16] This approach is well known for its capability to effectively handle design problems involving a large number of design variables and a few number of objective/constraint functions.

The sensitivities are obtained by solving a system of equations of complexity equivalent to the governing equations of the flow. When compared to traditional finite-difference methods, the adjoint approach enables large computational savings, at the expense of a more complex implementation.[20] This work employs a discrete adjoint formulation: the adjoint system of equations is obtained by differentiating the discretized form of the governing equations.

Let $\mathbf{W}$ be the set of all flow variables at discrete grid points arising from an approximate solution of the governing equations, $\alpha$ the set of design variables which influence the flow, and $J$ the scalar function which approximates the desired cost function. Then, within the context of control theory, the design problem can be posed as

$$\begin{aligned} \textit{Minimize} \quad & J(\mathbf{W},\alpha) \\ \textit{w.r.t.} \quad & \alpha, \\ \textit{subject to} \quad & R(\mathbf{W},\alpha)=0 \\ & C_i(\mathbf{W},\alpha)=0 \qquad i=1,...,m \end{aligned} \tag{5}$$

where $R(\mathbf{W},\alpha)=0$ represents the discrete governing equations and boundary conditions that must be satisfied and $C_i(\mathbf{W},\alpha)=0$ are $m$ additional constraints that may or may not involve the flow solution.

When using a gradient-based optimizer to solve the problem (5), the sensitivity of both the cost function $J$ and the constraints $C_i$ with respect to the design variables are required. By constructing the adjoint system of equations (6) and solving for the vector of adjoint variables $\lambda$

$$\left[\frac{\partial R}{\partial W}\right]^T \lambda = \left[\frac{\partial J}{\partial W}\right]^T, \tag{6}$$

the sensitivity of the cost function is simply given by

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial \alpha} - \lambda^T \frac{\partial R}{\partial \alpha}. \tag{7}$$

The reader is encouraged to consult the references[15,16] for more details. An additional adjoint system has to be solved for each additional constraint function $C$, which includes computing a new right-hand side for the system (6).

The sensitivity obtained from (7) can then be used to find the search direction of the gradient-based optimization algorithm as indicated in the schematic in figure 1.

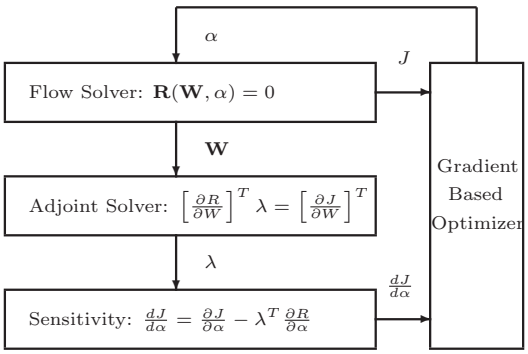



**Figure 1. Schematic of the adjoint-based optimization algorithm.**

The advantage of the adjoint approach can be seen from equation (7), which is independent of $\delta W$, meaning that the gradient of $J$ with respect to an arbitrarily large vector of design variables $\alpha$ can be determined without the need for additional flow-field evaluations.

## IV. Numerical Model

### A. MHD Solver

The MHD governing equations (1) are solved using a finite-difference discretization scheme. For this purpose, a coordinate transformation from physical coordinates $(x,y,z)$ to computational coordinates $(\xi,\eta,\zeta)$ is used, resulting in

$$\frac{\partial \bar{\mathbf{W}}}{\partial t} + \frac{\partial \bar{\mathbf{E}}}{\partial \xi} + \frac{\partial \bar{\mathbf{F}}}{\partial \eta} + \frac{\partial \bar{\mathbf{G}}}{\partial \zeta} = \bar{\mathbf{S}}\,, \tag{8}$$

where

$$\begin{cases} \bar{\mathbf{W}} = \frac{\mathbf{W}}{J} \\ \bar{\mathbf{E}} = \frac{1}{J}(\xi_x \mathbf{E} + \xi_y \mathbf{F} + \xi_z \mathbf{G}) \\ \bar{\mathbf{F}} = \frac{1}{J}(\eta_x \mathbf{E} + \eta_y \mathbf{F} + \eta_z \mathbf{G}) \\ \bar{\mathbf{G}} = \frac{1}{J}(\zeta_x \mathbf{E} + \zeta_y \mathbf{F} + \zeta_z \mathbf{G}) \\ \bar{\mathbf{S}} = \frac{1}{J}\mathbf{S} \end{cases}, \tag{9}$$

with $J$ the coordinate transformation Jacobian. More details about generalized coordinate transformations can be found in Hoffmann.[21]

After the spatial discretization of the governing equations (8) has been carried out, a system of ordinary differential equations (ODE) is obtained

$$\frac{d}{dt}\left(\frac{W_{ijk}}{J_{ijk}}\right) + R_{ijk} = 0. \tag{10}$$

where the tryad $ijk$ represents each and every node in the mesh. The resulting set of coupled ODEs (10) are marched in time using an explicit five-stage modified Runge-Kutta scheme to steady state.

The flow solver used is a new higher-order solver that has been developed at Stanford University under the sponsorship of the Department of Energy. The solver is a generic node-centered multi-block solver for the Reynolds-averaged Navier-Stokes equations. The finite-difference operators and artificial dissipation terms follow the work by Mattson[22,23] and the boundary conditions are implemented by means of penalty terms, according to the work by Carpenter.[24,25] The additional magnetic source terms had to be included in this solver so that MHD computations could be performed.

In the present case, the residual $R_{ijk}$ in equation (10) includes the inviscid fluxes, the artificial dissipation fluxes, the penalty terms for the boundary conditions and the magnetic source terms.

### B. Discrete Adjoint Solver

The theory of the implementation of the discrete adjoint solver for the MHD equations basically follows that of the authors previous works[15,16] which is based on the work of Giles.[11] There are some new developements worth pointing out in the present work.

The discrete adjoint system of equations (6) was constructed by differentiating all the numerical fluxes that comprised the residual $R_{ijk}$ in the discretized governing equations (10). However, taking advantage of using a discrete adjoint approach, we moved from a tedious and error-prone hand differentiation technique to the use of an automatic differentiation (AD) tool. There are a number of software tools for automatic differentiation, such as ADIFOR,[26] Tapenade[27] and TAF,[28] but Tapenade was chosen because it is the only non-commercial tool that supports Fortran 90.

The baseline implementation of the flow solver computes the residual using loops over the nodes of each computational block in each of the processors in the calculation. To make the implementation of the discrete adjoint solver more efficient, it was necessary to re-write the flow residual routine such that it computed the residual for a single specified node. This required a process of copy-and-paste from the original code that turned out to be straightforward. The re-engineered residual routine became a function with the residual `rAdj` returned as an output argument and the stencil of flow variables `wAdj` that affected that residual provided as an input argument

```
subroutine residualAdj(wAdj,rAdj,i,j,k)
```

The stencil of flow variables `wAdj` extended two nodes in each direction to allow for a second-order discretization as shown in figure 2.

The boundary condition penalty terms were moved to a separate routine that also had the boundary subface `mm` and the corresponding flow variable that was the donor to the penalty state `wDonorAdj` as input parameters

```
subroutine residualPenaltyAdj(wAdj,dwAdj,mm,wDonorAdj,i,j,k)
```

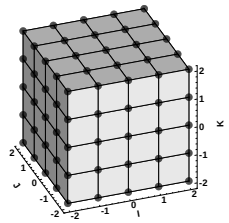


**Figure 2. Flow variables stencil `wAdj` used in the AD.**

This penalty residual routine was only called when the node `(i,j,k)` was located at a boundary face.

Having verified the re-written residual routines by comparing the residual values to the ones computed with the original code, these routines were then fed into Tapenade. Using the reverse mode of automatic differentiation in Tapende, the automatically differentiated routines were obtained as

```
SUBROUTINE RESIDUALADJ_B(wAdj,wadjb,rAdj,radjb,i,j,k)
```

and

```
SUBROUTINE RESIDUALPENALTYADJ_B(wAdj,wadjb,rAdj,radjb,mm,wdonoradj,wdonoradjb,i,j,k)
```

These AD routines readily compute the entries of $\frac{\partial R}{\partial W}$ since

$$\texttt{wAdjb}(\texttt{ii},\texttt{jj},\texttt{kk},\texttt{n}) = \frac{\partial R(i,j,k,m)}{\partial W(i+ii,j+jj,k+kk,n)}\,. \tag{11}$$

This allows for an easy assembly of the flux Jacobian matrix $\frac{\partial R}{\partial W}$ as shown in the routine pseudo-code in figure 3.

```
subroutine setupADjointMatrix
(...)
! Loop over the local computational blocks.
do nn=1,nDom
  ! Loop over location of output (R) cell of residual
  do k=1,kl
    do j=1,jl
      do i=1,il
        ! Transfer state w to auxiliar stencil array wAdj(:,:,:,:)
        call copyADjointStencil(wAdj,i,j,k)
        ! Loop over the outputs (R)
        do m=1,nwFlow
          ! Initialize the seed for the reverse mode to return dR(m)/dw
          rAdjB(:)=zero; rAdjB(m)=one; rAdj(:)=zero; wAdjB(:,:,:,:)=zero
          ! Call reverse mode of residual computation
          call residualAdj_B(wAdj,wAdjB,rAdj,rAdjB,i,j,k)
          ! Store block Jacobians (by rows).
          Aad(m,:) = wAdjB(0,0,0,:); Bad(m,:) = wAdjB(-1,0,0,:)
          (...)
        enddo
        ! Transfer block Jacobians to PETSc matrix.
        ! >>> center block dR(i,j,k)/W(i,j,k)
        call MatSetValuesBlocked(dRdW,1,idxmgb,1,idxngb,Aad,INSERT_VALUES,ierr)
        ! >>> west block B < W(i-1,j,k)
        call MatSetValuesBlocked(dRdW,1,idxmgb,1,idxngb,Bad,INSERT_VALUES,ierr)
        (...)
```

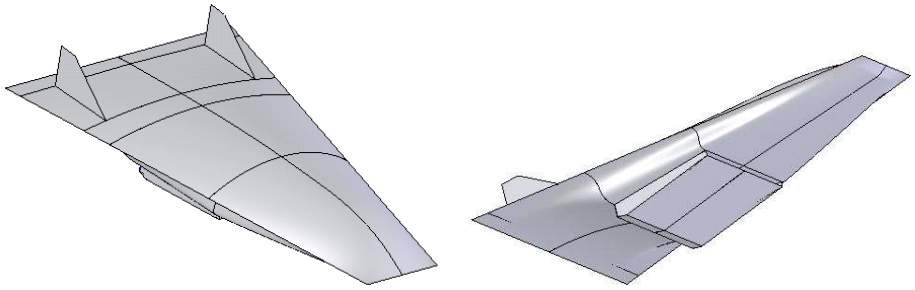
**Figure 3. Flux Jacobian matrix assembly routine.**

In order to solve the large discrete adjoint matrix problem (6), the Portable, Extensible Toolkit for Scientific Computation (PETSc) [29] was used. This allows for MPI parallel implementations, it has several linear iterative solvers and preconditioners available and performs very well, provided that a careful object creation and assembly procedure is followed. All the adjoint and partial sensitivity matrices and vectors were created as PETSc objects, the adjoint system of equations (6) was solved using a Krylov subspace method. More specifically, a Generalized Minimum Residual (GMRES) method was used, preconditioned with the block Jacobi method, with one block per processor, each of which solved with ILU(0) preconditioning. The total sensitivity (7) was computed using the matrix-vector operation routines provided in PETSc.

# V. Results

## A. Test Case

As a test case for this work, we selected a in-house modeled hypersonic vehicle geometry, as shown in figure 4, and simplified by removing the scramjet propulsion system and the vertical fins.



(a) Top view (b) Bottom view

**Figure 4. Concept of hypersonic vehicle configuration.**

Since all the simulations were run without any side-slip angle, only half the body must be modeled, as shown in figure 5, where the plane $y = 0$ was set to be the symmetry plane.

A collection of five hypothetical dipoles is placed inside the body, at the locations indicated in table 1, which imposes a magnetic field on the flow. The resulting field given by the superposition of dipoles is shown in figure 6. Even though only half domain was simulated, all 5 dipoles were taken into account to compute the imposed magnetic field.

| Dipole # | Location | Orientation |
|---|---|---|
| 1 | $(0.5, 0, 0)$ | $(-1, 0, 0)$ |
| 2/3 | $(1.5, \pm 0.96, 0)$ | $(0, \pm 1, 0)$ |
| 4/5 | $(3.5, \pm 1.24, 0)$ | $(0, \pm 1, 0)$ |

**Table 1. Dipole locations.**

The runs were made on a parallel processor workstation, with four 3.2Ghz nodes and 8GB of RAM. The multi-block computational domain is shown in figure 7 and had a total of 183,342 nodes. The block dimensions and boundary conditions (BC) are enumerated in table 2, where all the physical BCs are supersonic and the wall inviscid.

| Block | Dimensions (i,j,k) | i Min | i Max | j Min | j Max | k Min | k Max |
|---|---|---|---|---|---|---|---|
| A : Rear | (49,49,17) | Inflow | Outflow | Outflow | Symmetry | Internal | Outflow |
| B : Body bottom | (25,25,33) | Inflow | Wall | Internal | Symmetry | Internal | Internal |
| C : Body top | (33,25,25) | Internal | Internal | Symmetry | Internal | Outflow | Wall |
| D : Side bottom | (33,25,25) | Internal | Internal | Internal | Outflow | Internal | Inflow |
| E : Side top | (25,25,33) | Internal | Outflow | Internal | Outflow | Internal | Internal |
| F : Front | (25,49,49) | Internal | Inflow | Symmetry | Outflow | Outflow | Inflow |

**Table 2. Multi-block domain and BCs.**

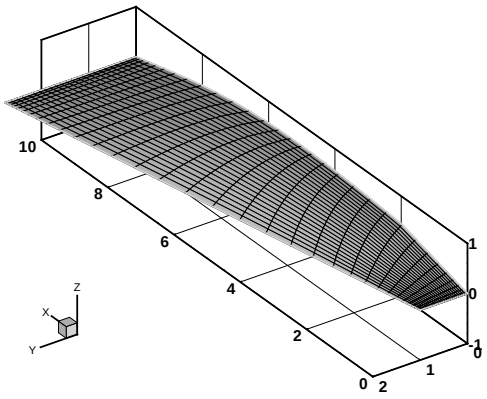



**Figure 5. Half-body configuration modeled.**

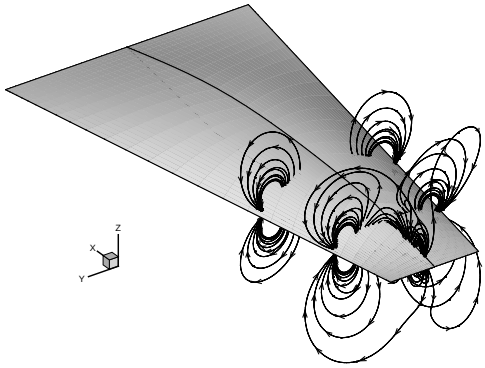



**Figure 6. Imposed magnetic field.**

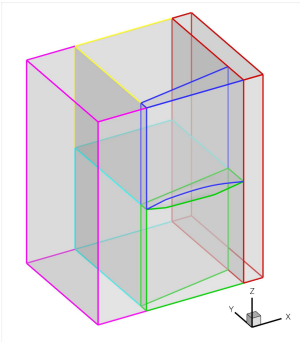



**Figure 7. Multi-block domain.**

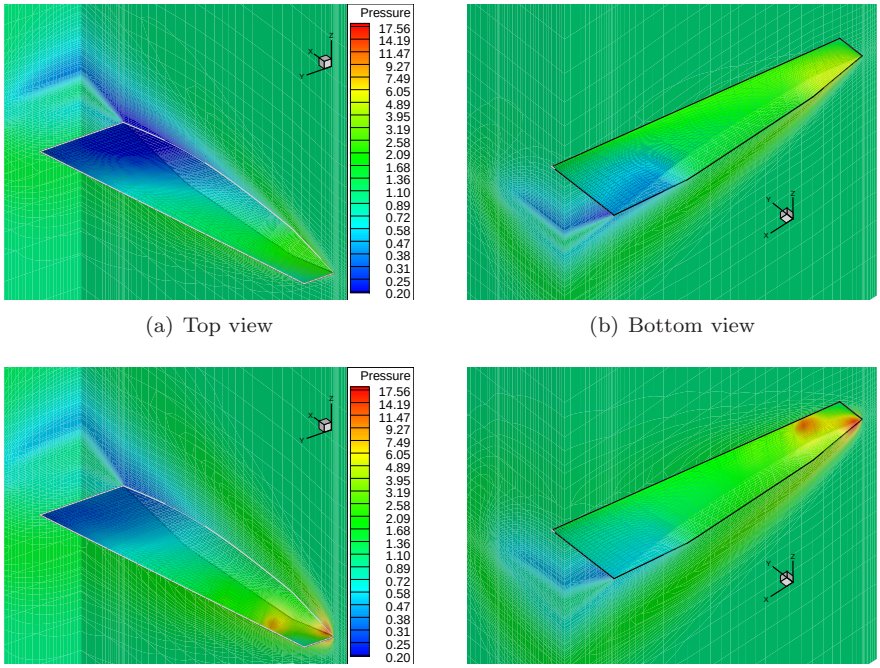
In the present work, the usual aerodynamic coefficient definitions (lift, drag, moment in x-,y-,z-direction) were used as the cost function $J$ and the electrical conductivity $\sigma$ in each and every computational node was taken as the design variables. This led to a total of 5 cost functions and 183,342 design variables.

## B. Flow Solution

The free-stream flow conditions chosen were Mach 5 and an angle of attack of $5^\circ$. The flow solver was first run without any imposed magnetic field, corresponding to the Euler equations, and then the dipoles were turned on to obtain the MHD solution. All the dipoles were set to the same strength and the baseline electrical conductivity was such that a magnetic Reynolds number of $Re_\sigma = 0.57$ and a magnetic interaction parameter $Q = 0.3$ were used. The computed pressure at the body surface and on the plane of symmetry can be seen in figure 8.

As expected, there is large pressure increase close to the dipoles due to the imposed magnetic field. This effect is caused by the additional source terms in the MHD equations and makes the numerical solution much less stable, being necessary to run at significantly lower CFL numbers. Because of this, while the Euler solution took only 768 seconds for the residual to converge ten orders of magnitude, the equivalent MHD solution took 6,245 seconds. This clearly rules out the use of finite-differences to compute cost function gradients and highlights the importance of an alternative approach such as the discrete adjoint-based gradients. Moreover, the slower convergence highlights the need for an implicit treatment of the source terms in the MHD solution that we intend to persue in the near future.

The resulting cost function values are summarized in table 3. The reference area and moment arms were taken as $18.8m$ and $1m$, respectively.




(a) Top view (b) Bottom view

(c) Top view (d) Bottom view

**Figure 8. Euler and MHD solutions.**

| | |
|---|---|
| $C_L$ | 0.0520750437522571 |
| $C_D$ | 0.0179831679282085 |
| $C_{Mx}$ | 0.0496412105981379 |
| $C_{My}$ | 0.0247952565444227 |
| $C_{Mz}$ | 0.0504365242914952 |

**Table 3. Baseline cost function values.**

## C. Adjoint System of Equations

The entries of the adjoint matrix $\frac{\partial R}{\partial W}$ and vector $\frac{\partial J}{\partial W}$ were computed by calling the automatically differentiated routines, and then globally assembled using PETSc. The non-zero pattern can be visualized using the PETSc routine calls

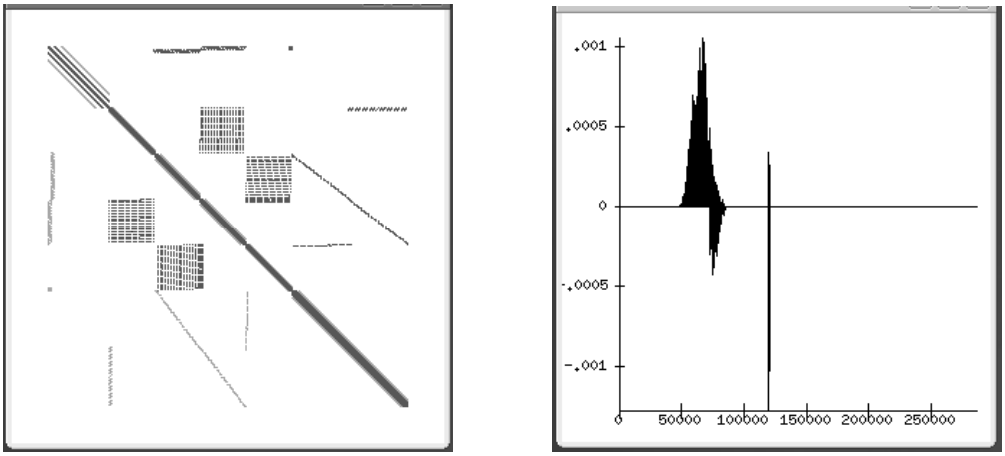
```
call MatView(dRdW,PETSC_VIEWER_DRAW_WORLD,ierr)
call VecView(dJdW,PETSC_VIEWER_DRAW_WORLD,ierr)
```

and are shown in figure 9. The vector $\frac{\partial J}{\partial W}$ is shown for $J = C_L$ but similar results are obtained for the other cost functions.

The assembly time of the Jacobian matrix was 10.17 seconds, including all the calls to the automatically differentiated routines and PETSc matrix functions, whereas the adjoint vector assembly time was negligible.

Once the adjoint system of equations (6) was set up, the GMRES solver provided by PETSc was used. To be consistent with the flow solver, the adjoint solution residual convergence criterion was also set to $10^{-10}$. The iterative solver consistently shows very good robustness and convergence properties. For all the different cost functions tested, the convergence was typically achieved after about 65 iterations, which took 55 seconds to run. The residual history of the adjoint solution using PETSc for $J = C_L$ is illustrated in figure 10.

The flow solution and the associated adjoint solution are shown in figure 11. It is interesting to point out that the adjoint solution, for the cost functions used in this work, is similar to the flow solution, except that the flow direction is reversed. This occurrence has already been observed in other works involving the adjoint method.[30]




(a) Adjoint Matrix: $\frac{\partial R}{\partial W}$ (b) Adjoint Vector: $\frac{\partial J}{\partial W}$

**Figure 9. Adjoint matrix and vector sparsity pattern.**

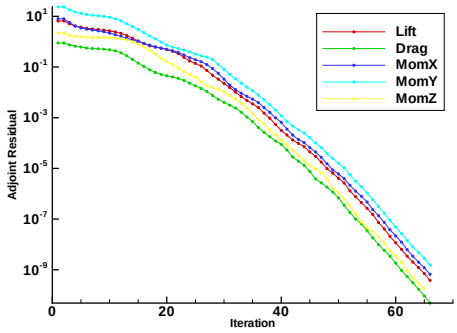



**Figure 10. Adjoint residual convergence history.**

In addition, using a coarser grid with only 57,214 nodes, the Jacobian matrix entries $\frac{\partial R}{\partial W}$ were computed with both central finite-differences and the automatically differentiated routines. This was primarily done to debug the code but also showed the dramatic performance gain obtained using the AD routines, as shown in table 4. It is important to notice that this timings correspond to a single processor computation using a non-optimized executable, but the ratio is expected to remain similar for other conditions.

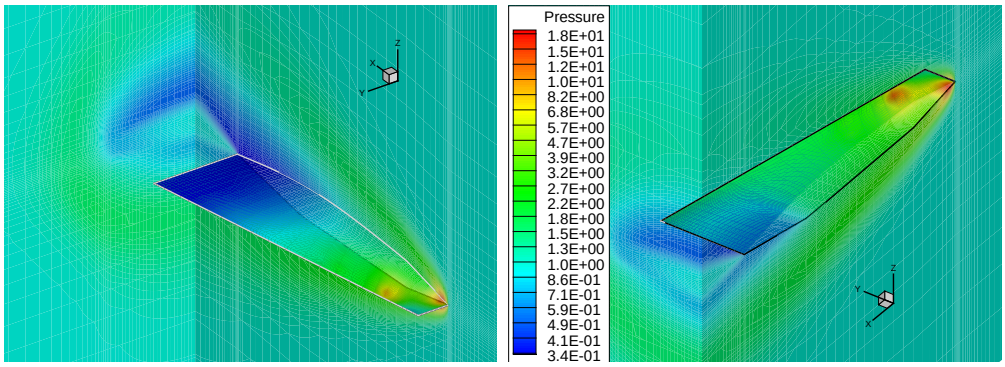
| | |
|---|---|
| Finite-Differences (central) | 1856 sec |
| Automatic Differentiation | 102 sec |
| Performance Gain Ratio | 18.2 × |

**Table 4. Computational time.**

## D. Total Sensitivity

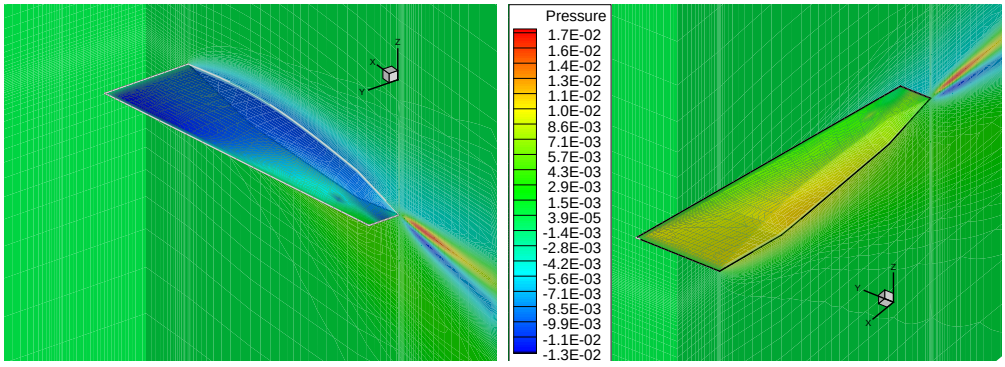
Once the adjoint solution is obtained, the total sensitivity can then quickly by computed using (7), thus requiring the calculation of the partial derivatives $\frac{\partial R}{\partial \alpha}$ and $\frac{\partial J}{\partial \alpha}$. However, since the design variables chosen were the electrical conductivity in each computational node, these partial derivatives were computed analytically: $R$ depends linearly on $\sigma$ due to its MHD source term and all the aerodynamic coefficients used as cost function $J$ do not depend explicitly on $\sigma$. The non-zero patterns of these objects are shown in figure 12.

The total sensitivity was then computed for the different cost functions and exists everywhere in the volume since the design variable $\sigma$ spanned the entire problem domain. For visualization purposes, the values at the body surface and symmetry plane are presented in figures 13, 14 and 15, corresponding to the lift, drag and pitching moment coefficient sensitivities with respect to the electrical conductivity on these surfaces, respectively.



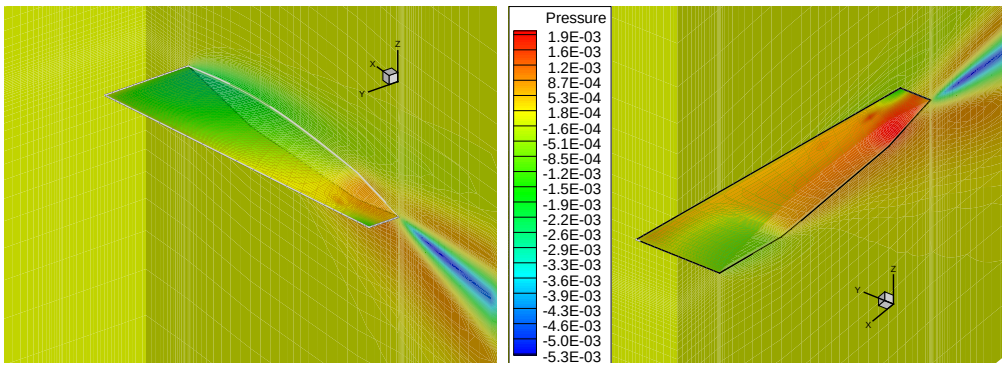

(a) Flow pressure: top view

(b) Flow pressure: bottom view



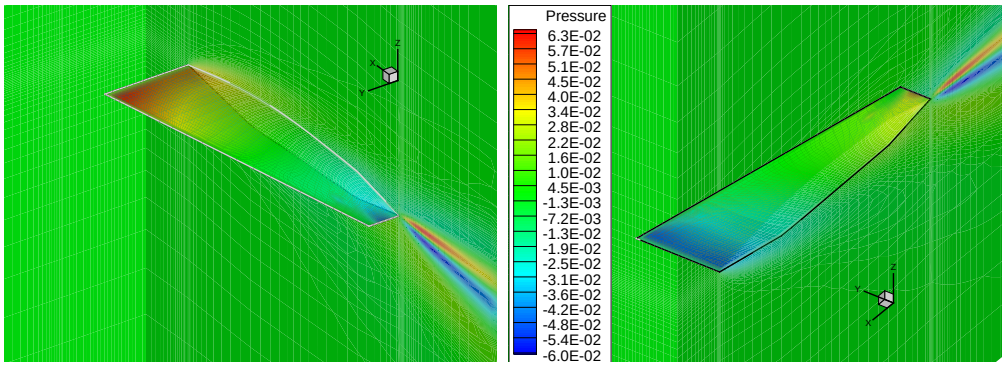

(c) Adjoint pressure $J = C_L$ : top view

(d) Adjoint pressure $J = C_L$ : bottom view




(e) Adjoint pressure $J = C_D$ : top view

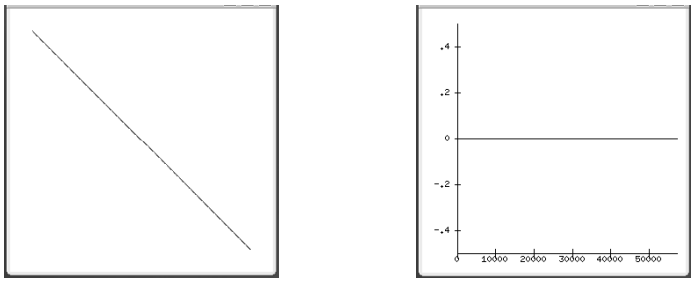
(f) Adjoint pressure $J = C_D$ : bottom view




(g) Adjoint pressure $J = C_{My}$ : top view

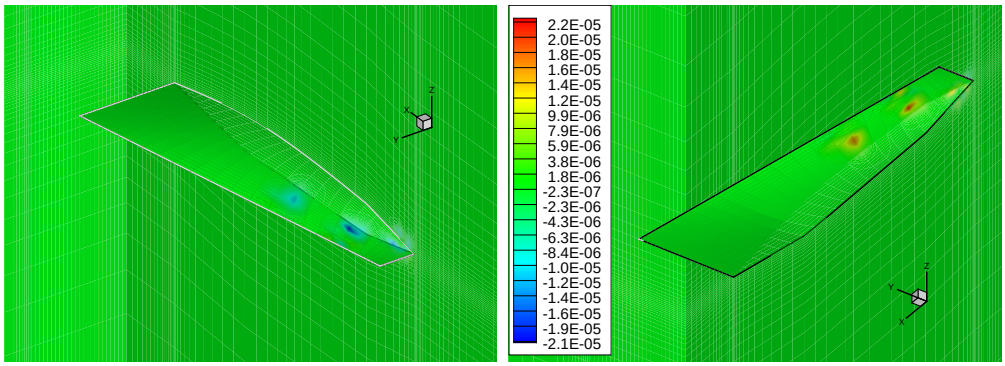
(h) Adjoint pressure $J = C_{My}$: bottom view

**Figure 11. Flow and adjoint solutions.**



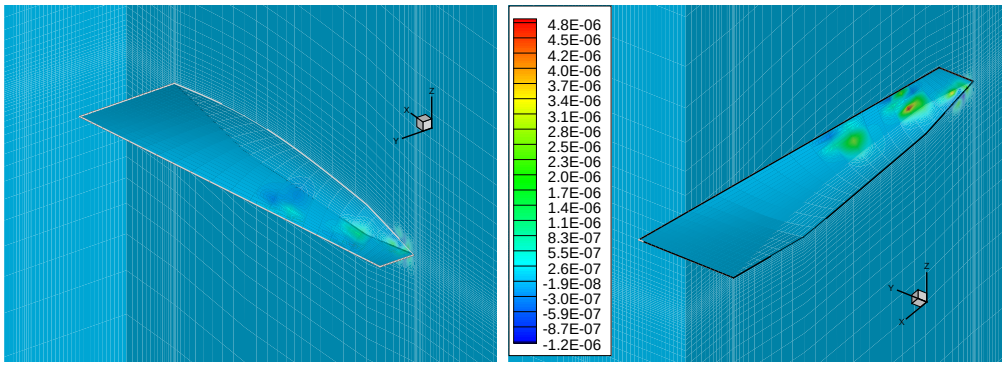

(a) Gradient matrix: $\frac{\partial R}{\partial \alpha}$ (b) Gradient vector: $\frac{\partial J}{\partial \alpha}$

**Figure 12. Parcial gradient matrix and vector pattern.**




(a) Top view (b) Bottom view

**Figure 13. Lift coef. sensitivity with respect to the electrical conductivity: $\frac{dC_L}{d\sigma}$.**



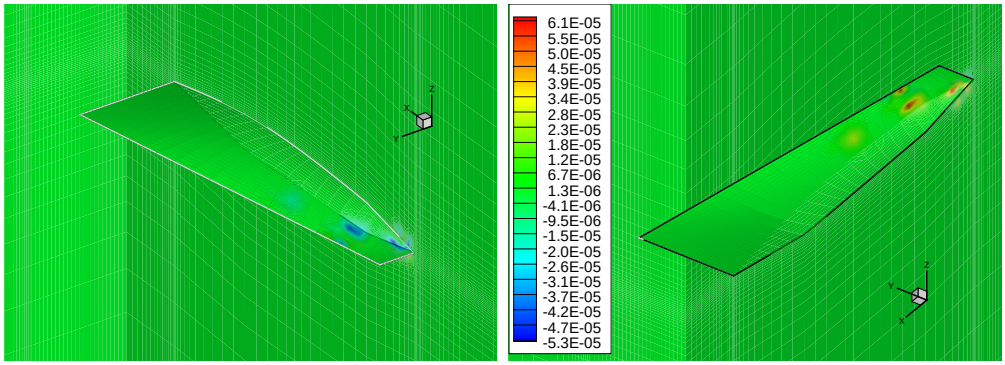

(a) Top view (b) Bottom view

**Figure 14. Drag coef. sensitivity with respect to the electrical conductivity: $\frac{dC_D}{d\sigma}$.**

## E. Verification of the Sensitivities

The sensitivities obtained using the discrete adjoint approach were matched against values obtained using the forward finite-difference solution. The comparison was made using three control nodes located on the body surface, as indicated in table 5, and the results are summarized in table 6 using two different finite-difference perturbation step sizes. The values in table 6 demonstrate two things: firstly, the agreement between the two different approaches is excellent, successfully verifying the adjoint-based gradient values; secondly, it shows how the finite-difference approach can break down when choosing a perturbation step that leads to subtractive cancellation issues, as seen with control node #3. If the complex-step method[31] had been used, these issues would have been overcome.

This verification also revealed that it would have been computationally prohibitive to compute the sensitivities with respect to such large number of design variables using anything but the adjoint method - to get




(a) Top view (b) Bottom view

**Figure 15. Pitching moment coef. sensitivity with respect to the electrical conductivity: $\frac{dC_{My}}{d\sigma}$.**

| Node # | Description | Zone | Location (i,j,k) |
|---|---|---|---|
| 1 | body bottom, under dipole 2 | B | iMax @ (25,11,10) |
| 2 | body top, above dipole 1 | C | kMax @ (28,6,25) |
| 3 | slightly below body nose | F | iMin @ (1,3,29) |

**Table 5. Location of the control nodes.**

| Control Node # | Cost Function J | Adjoint | Finite-Diff. (step $10^{-3}$) | Δ | Finite-Diff. (step $10^{-4}$) | Δ |
|---|---|---|---|---|---|---|
| 1 | $C_L$ | 2.28079E-5 | 2.28042E-5 | -0.016 % | 2.28075E-5 | -0.002 % |
| | $C_D$ | 4.90557E-6 | 4.90473E-6 | -0.017 % | 4.90548E-6 | -0.002 % |
| | $C_{Mx}$ | 1.03367E-5 | 1.03348E-5 | -0.018 % | 1.03365E-5 | -0.002 % |
| | $C_{My}$ | 6.25820E-5 | 6.25715E-5 | -0.017 % | 6.25809E-5 | -0.002 % |
| | $C_{Mz}$ | 6.61905E-6 | 6.61751E-6 | -0.023 % | 6.61889E-6 | -0.002 % |
| 2 | $C_L$ | -1.49820E-5 | -1.49798E-5 | -0.015 % | -1.49820E-5 | 0.000 % |
| | $C_D$ | 1.22216E-6 | 1.22198E-6 | -0.015 % | 1.22215E-6 | -0.001 % |
| | $C_{Mx}$ | -4.70550E-6 | -4.70480E-6 | -0.015 % | -4.70551E-6 | 0.000 % |
| | $C_{My}$ | -5.37240E-5 | -5.37161E-5 | -0.015 % | -5.37234E-5 | -0.001 % |
| | $C_{Mz}$ | 4.11529E-6 | 4.11471E-6 | -0.014 % | 4.11532E-6 | 0.001 % |
| 3 | $C_L$ | 2.26830E-7 | 2.26822E-7 | -0.003 % | 2.24640E-7 | -0.966 % |
| | $C_D$ | 5.73601E-8 | 5.73601E-8 | 0.000 % | 5.71386E-8 | -0.386 % |
| | $C_{Mx}$ | 3.61174E-8 | 3.61146E-8 | -0.008 % | 3.41968E-8 | -5.318 % |
| | $C_{My}$ | 1.28740E-6 | 1.28739E-6 | -0.001 % | 1.28649E-6 | -0.071 % |
| | $C_{Mz}$ | 6.13412E-8 | 6.13439E-8 | 0.004 % | 6.10878E-8 | -0.413 % |

**Table 6. Electrical conductivity sensitivity verification: $\frac{dJ}{d\sigma}$.**

the flow solver to converge (starting from the baseline solution) every time the electrical conductivity was perturbed in a single node in the domain, took roughly 5 minutes. Extrapolating to all nodes, corresponding to the 183,342 design variables, it would have taken almost two years to obtain the same results that took less than a minute for the adjoint method described. The detailed computational costs are summarized in table 7. It is important to notice that the flow solver has not been optimized for MHD computations yet. However, in realistic design problems with optimized flow and adjoint solvers, 5-10 cost functions and about 100 design variables, the automatic discrete adjoint-based gradients are expected to be at least 20-50 times faster compared with finite-difference approximations.

| Solver | Wall clock time |
|---|---|
| Flow Solution - Euler (explicit RK5) | 13 minutes |
| Flow Solution - Euler (implicit DDD-ADI) | 2.5 minutes |
| Flow Solution - MHD (explicit RK5) | 1 hour |
| Sensitivities via Adjoint Solution | 1 minute / cost function |
| Sensitivities via Finite-Differences | 5 minutes / design variable |

**Table 7. Computational cost comparison.**

## VI. Conclusions

In this paper the authors have obtained sensitivities of different cost functions with respect to almost two hundred thousand design variables for problems governed by the magneto-hydrodynamics equations under the low magnetic Reynolds number assumption.

The sensitivities were obtained using a discrete adjoint system of equations that was derived using automatic differentiation. Besides drastically reducing the implementation time, this method produces gradients that are exactly consistent with the flow solver discretization and permits the use of arbitrary cost functions and design variables. To make this approach time- and memory-efficient, only the flow solver routine responsible for the residual calculation was transformed in this fashion, where a node-per-node approach was used. The adjoint-based results were verified against sensitivities obtained by finite differencing the flow solver and a near perfect agreement was demonstrated. This also highlighted the large computational savings that result from using the discrete adjoint method.

This work represents the first time this approach is employed to obtain sensitivities in an intrinsically parallel computational MHD environment, since both the flow and adjoint solvers benefit from a parallel MPI implementation. When using gradient-based optimizers, it paves the way to perform design of realistic configurations using the MHD governing equations.

Further work needs to be done in order to incorporate non-ideal MHD effects, such as the viscous Navier-Stokes terms and the magnetic dispersive terms in the equations, to include other design variables and cost functions, and to integrate these tools in an automatic design framework.

The authors believe the implementation methodology described in this paper is particularly well suited to compute the gradients of any cost function in an optimization problem involving any set of governing equations that can be cast in the form $R(W, \alpha) = 0$, when the number of design variables considerably outnumbers the number of cost functions or when the solution of the governing equations is computationally too expensive to allow the use of finite-differences.

## VII. Acknowledgments

The authors are grateful for the Air Force Office of Scientific Research contract No. FA9550-04-C-0105 under tasks monitored by John Schmisseur. The first author also acknowledges the support of the *Fundação para a Ciência e a Tecnologia* from Portugal. We would also like to thank Dr. Edwin van der Weide for his assistance with the flow solver and San Gunawardana for providing the computational multi-block mesh.

## References

[1]NASA, "X-43A Hypersonic Scramjet-Powered Research Aircraft," www.nasa.gov/missions/research/x43-main.html.

[2]Whitney, P. ., "Pratt & Whitney Rocketdyne Completes Mach 5 Testing of Worlds First Closed-Loop Hydrocarbon-Fueled Hypersonic Propulsion System," http://www.pratt-whitney.com/pr_0727063.asp, accessed in October 2006.

[3]DARPA, "Falcon program, Tactical Technology Office," http://www.darpa.mil/tto/programs/falcon.htm, accessed in October 2006.

[4]Poggie, J. and Gaitonde, D. V., "Computational Studies of Magnetic Control in Hypersonic Flow," *AIAA Paper* 2001-0196, Jan. 2001, Proceedings of the $39^{th}$ AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV.

[5]Powell, K. G., "An Approximate Riemann Solver for Magnetohydrodynamics (that works in more than one dimension)," ICASE Report 94-24, Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA, 1994.

[6]Gaitonde, D. V., "Higher-Order Solution Procedure for Three-Dimensional Nonideal Magnetogasdynamics," *AIAA Journal*, Vol. 39, No. 11, Nov. 2001, pp. 2111–2120.

[7]Gaitonde, D. V. and Poggie, J., "Implicit Technique for Three-Dimensional Turbulent Magnetoaerodynamics," *AIAA Journal*, Vol. 41, No. 11, Nov. 2003, pp. 2179–2191.

[8]MacCormack, R. W., "A Computational Method for Magnetofluid Dynamics," *AIAA Paper* 2001-2735, June 2001, Proceedings of the $32^{nd}$ AIAA Plasmadynamics and Lasers Conference, Anaheim, CA.

[9]Walters, R. W., Cinnella, P., Slack, D. C., and Halt, D., "Characteristic-based Algorithms for Flows in Thermo-Chemical Nonequilibrium," *AIAA Paper* 1990-0393, Jan. 1990, Proceedings of the $28^{th}$ AIAA Aerospace Sciences Meeting, Reno, NV.

[10]Gaitonde, D. V., "Simulation of Local and Global High-Speed Flow Control with Magnetic Fields," *AIAA Paper* 2005-0560, Jan. 2005, Proceedings of the $43^{rd}$ AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV.

[11]Giles, M. B. and Pierce, N. A., "An Introduction to the Adjoint Approach to Design," *Flow, Turbulence and Combustion*, Vol. 65, Kluwer Academic Publishers, 2000, pp. 393–415.

[12]Lewis, R. M., "Numerical Computation of Sensitivities and the Adjoint Approach," *Computational Methods for Optimal Design and Control*, edited by J. Borggaard, J. Burns, E. Cliff, and S. Schreck, Birkhäuser, 1998, pp. 285–302, Also available as ICASE technical report 97–61.

[13]Reuther, J. J., Jameson, A., Alonso, J. J., Rimlinger, M. J., and Saunders, D., "Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers, Part 1," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 51–60.

[14]Jameson, A., Pierce, N. A., and Martinelli, L., "Optimum Aerodynamic Design using the Navier Stokes Equations," *Theoretical and Computational Fluid Dynamics*, Vol. 10, Springer-Verlag GmbH, Jan. 1998, pp. 213–237.

[15]Marta, A. C., Alonso, J. J., and Tang, L., "Automatic Magnetohydrodynamic Control of Hypersonic Flow Using a Discrete Adjoint Formulation," *AIAA Paper* 2006-0370, Jan. 2006, Proceedings of the $44^{th}$ AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV.

[16]Marta, A. C. and Alonso, J. J., "Discrete Adjoint Formulation for the Ideal MHD Equations," *AIAA Paper* 2006-3345, June 2006, Proceedings of the $3^{rd}$ AIAA Flow Control Conference, San Francisco, CA.

[17]Martins, J., Alonso, J., and van der Weide, E., "An Automated Approach for Developing Discrete Adjoint Solvers," *AIAA Paper* 2006-1608, May 2006, Proceedings of the $47^{th}$ AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Newport, RI.

[18]Dwight, R. P. and Brezillon, J., "Effect of Various Approximations of the Discrete Adjoint on Gradient-Based Optimization," *AIAA Paper* 2006-0690, Jan. 2006, Proceedings of the $44^{th}$ AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV.

[19]Nadarajah, S. K. and Jameson, A., "Studies of the Continuous and Discrete Adjoint Approaches to Viscous Automatic Aerodynamic Shape Optimization," *AIAA Paper* 2001-2530, June 2001, Proceedings of the $15^{th}$ AIAA Computational Fluid Dynamics Conference, Anaheim, CA.

[20]Martins, J. R. R. A., Alonso, J. J., and Reuther, J. J., "Complete Configuration Aero-Structural Optimization Using a Coupled Sensitivity Analysis Method," *AIAA Paper* 2002-5402, Sept. 2002, Proceedings of the 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, GA.

[21]Hoffmann, K. A., *Fundamental Equations of Fluid Mechanics*, Engineering Education System, 1996.

[22]Mattsson, K. and Nordström, J., "Summation by Parts Operators for Finite Difference Approximations of Second Derivatives," *Journal of Computational Physics*, Vol. 199, No. 2, Sept. 2004, pp. 503–540.

[23]Mattsson, K., Svärd, M., and Nordström, J., "Stable and Accurate Artificial Dissipation," *Journal of Scientific Computing*, Vol. 21, No. 1, Aug. 2004, pp. 57–79.

[24]Carpenter, M. H., Gottlieb, D., and Abarbanel, S., "Time-Stable Boundary Conditions for Finite-Difference Schemes Solving Hyperbolic Systems: Methodology and Application to High-Order Compact Schemes," *Journal of Computational Physics*, Vol. 111, No. 2, April 1994, pp. 220–236.

[25]Carpenter, M. H., Nordström, J., and Gottlieb, D., "A Stable and Conservative Interface Treatment of Arbitrary Spatial Accuracy," *Journal of Computational Physics*, Vol. 148, No. 2, Jan. 1999, pp. 341–365.

[26]Bischof, C., Carle, A., Hovland, P., Khademi, P., and Mauer, A., "ADIFOR 2.0 User's Guide (Revision D)," Tech. Rep. Technical Memorandum No.192, Argonne National Laboratory, 1998.

[27]Dervieux, A., Hascoet, L., Pascual, V., Koobus, B., and Vazquez, M., "TAPENADE Web page," 2005, http://www-sop.inria.fr/tropics/tapenade.html.

[28]Giering, R. and Kaminski, T., "Applying TAF to generate efficient derivative code of Fortran 77-95 programs," *PAMM*, Vol. 12, No. 1, 2003, pp. 54–57.

[29]Balay, S., Buschelman, K., Gropp, W. D., Kaushik, D., Knepley, M. G., McInnes, L. C., Smith, B. F., and Zhang, H., "PETSc Web page," 2001, http://www.mcs.anl.gov/petsc.

[30]Lee, K.-H., Alonso, J. J., and van der Weide, E., "Mesh Adaptation Criteria for Unsteady Periodic Flows Using a Discrete Adjoint Time-Spectral Formulation," *AIAA Paper* 2006-0692, Jan. 2006, Proceedings of the $44^{th}$ AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV.

[31]Martins, J. R. R. A., Kroo, I. M., and Alonso, J. J., "An Automated Method for Sensitivity Analysis Using Complex Variables," *AIAA Paper* 2000-0689, Jan. 2000, Proceedings of the 38th Aerospace Sciences Meeting, Reno, NV.