Discrete and continuous adjoint method for compressible CFD

J. Peter ONERA

J. Peter

ONERA DMFN

October 2, 2014
Outline

1. Introduction
2. Discrete adjoint method
3. Continuous adjoint method
4. Discrete vs Continuous adjoint
5. Conclusions
Well-known aerodynamic optimization problems of the utmost importance

- Aircraft drag reduction
- Reduction of total pressure losses of a blade row.

Strongly constrained problems (from aerodynamics, structure...)

Several approaches for researches and studies in external aerodynamics

- Flight tests
- Wind tunnel experiments (with flight Re/lower than flight Re)
- Numerical simulation

Numerical simulation most adapted
(Non solvable) pde $\rightarrow$ numerical simulation. Finite-volume simulation in this talk.

Infinite dimension possible deformation $\rightarrow$ parametrization

Finite dimensional maths

Which type of optimization method ?
Local or global optimization ?
Global optimization
- genetic/evolutionary algorithms, particle swarm, ant colony, CMA-ES...
- large number of function evaluations required
- combined with surrogate models
- in particular used for design space exploration with low fidelity models

Local optimization
- very valuable when starting from pre-optimized shapes
- pattern methods. e.g. simplex method
- gradient-based methods. e.g. steepest descent, conjugate gradient

Popular and efficient descent methods require objective and constraint sensitivities w.r.t. design parameters
Needed sensitivities w.r.t. design parameters

...not a trivial task in numerical simulation as state variables change with shape via the equations of the mechanical problem

Sensitivity calculation

- 70's 80's finite differences. Scaling with number of shape parameters
- Control theory  [Lions 71, Pironneau 73,74] aerodynamics shape optimization
  [Jameson 88] adjoint method. Scaling with the number of functions to be differentiated

Other applications of adjoint method: understanding zones of influence for function value, goal-oriented mesh refinement
Outline

1. Introduction
2. Discrete adjoint method
3. Continuous adjoint method
4. Discrete vs Continuous adjoint
5. Conclusions
Discrete adjoint method

- Framework: compressible flow simulation using finite volume method. Discrete approach for sensitivity analysis

- Notations
  - Volume mesh $X$, flowfield $W$ (size $n_a$)
  - Wall surface mesh $S$
  - Residual $R$, $C^1$ regular w.r.t. $X$ and $W$ – steady state: $R(W, X) = 0$
  - Vector of design parameters $\alpha$ (size $n_d$), $X(\alpha)$, $S(\alpha)$ $C^1$ regular

- Assumption of implicit function theorem
  - $\forall (W_i, X_i) / R(W_i, X_i) = 0$  $\partial R/\partial W(W_i, X_i) \neq 0$
  - Unique steady flow corresponding to a mesh
Introduction

Discrete gradient calculation methods

- Functions of interest
  - $J_k(\alpha) = J_k(W(\alpha), X(\alpha))$  \(k \in [1, n_f]\)
  - Flowfield and volume mesh linked by flow equations $R(W(\alpha), X(\alpha)) = 0$

- Sensitivities $dJ_k/d\alpha_i$  \(k \in [1, n_f]\)  \(i \in [1, n_d]\) to be computed

- Discrete gradient computation methods
  - Finite differences – $2n_d$ flow computations (non linear problems, size $n_a$)
  - Direct differentiation method – $n_d$ linear systems (size $n_a$)
  - Adjoint vector method – $n_f$ linear systems (size $n_a$)
Finite difference method

- Choose steps $\delta \alpha_i$. Get shifted meshes $X(\alpha + \delta \alpha_i), X(\alpha - \delta \alpha_i)$
- Solve flows

$$R(W(\alpha + \delta \alpha_i), X(\alpha + \delta \alpha_i)) = 0 \quad R(W(\alpha - \delta \alpha_i), X(\alpha - \delta \alpha_i)) = 0$$

$$\frac{dW}{d\alpha_i (FD)} = \frac{W(\alpha + \delta \alpha_i) - W(\alpha - \delta \alpha_i)}{2\delta \alpha_i}$$

- Compute outputs sensitivities

$$\frac{dJ_k}{d\alpha_i (FD)} = \frac{J_k(W(\alpha + \delta \alpha_i), X(\alpha + \delta \alpha_i)) - J_k(W(\alpha - \delta \alpha_i), X(\alpha + \delta \alpha_i))}{2\delta \alpha_i}$$

- Two issues: definition of $\delta \alpha_i$, cost of shifted flow solves
Discrete adjoint method

Direct differentiation method (1/2)

- Discrete equations for mechanics (set of $n_a$ non-linear equations)
  \[ R(W(\alpha), X(\alpha)) = 0 \]

- Differentiation with respect to $\alpha_i$ $i \in [1, n_d]$. Derivation of $n_d$ linear system of size $n_a$
  \[ \frac{\partial R}{\partial W} \frac{dW}{d\alpha_i} = -\left( \frac{\partial R}{\partial X} \frac{dX}{d\alpha_i} \right) \]

- Calculation of derivatives
  \[ \frac{dJ_k}{d\alpha_i} = \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha_i} + \frac{\partial J_k}{\partial W} \frac{dW}{d\alpha_i} \]
Gradient vectors

\[ \nabla_\alpha J_k(\alpha) = \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha} + \frac{\partial J_k}{\partial W} \frac{dW}{d\alpha} \]

Check the flow sensitivities using finite differences

\[ R(W(\alpha + \delta \alpha_i), X(\alpha + \delta \alpha_i)) = 0 \quad R(W(\alpha - \delta \alpha_i), X(\alpha - \delta \alpha_i)) = 0 \]

\[ \frac{dW}{d\alpha_i} \approx \frac{W(\alpha + \delta \alpha_i) - W(\alpha - \delta \alpha_i)}{2\delta \alpha_i} \]

Check the outputs sensitivities

\[ \frac{dJ_k}{d\alpha_i} \approx \frac{J_k(W(\alpha + \delta \alpha_i), X(\alpha + \delta \alpha_i)) - J_k(W(\alpha - \delta \alpha_i), X(\alpha + \delta \alpha_i))}{2\delta \alpha_i} \]
Discrete adjoint method

Mathematical game (1/4)

- Mathematical game in $\mathbb{R}^n$ to understand adjoint method
- given $(f, b_i) \in \mathbb{R}^n$ ($i \in \{1, n_d\}$), given $A \in \mathcal{M}(\mathbb{R}^n)$

$$x_i \cdot f \quad A x_i = b_i \quad i \in \{1, n_d\}$$

- Solution solving one linear system instead of $n_d$ linear systems ???
• Linear algebra reminder: the inverse of the transpose is the transpose of the inverse

\[ M^T (M^{-1})^T = (M^{-1}M)^T = I^T = I \]
\[ (M^{-1})^T M^T = (MM^{-1})^T = I^T = I \]

• The notation \( M^{-T} \) is suitable for \( (M^T)^{-1} / (M^{-1})^T \)
Mathematical game in $\mathbb{R}^n$ to understand adjoint method

given $(f, b_i) \in \mathbb{R}^n$ ($i \in \{1, n_d\}$), given $A \in \mathcal{M}(\mathbb{R}^n)$

Calculate the values of $f.x_i, A x_i = b_i, i \in \{1, n_d\}$

$f.x_i = f.(A^{-1}b_i) = ((A^{-1})^T f).b_i = (A^{-T} f).b_i$ efficient solution

Solve $A^T \lambda = f$ Calculate $\lambda.b_i, i \in \{1, n_d\}$
Mathematical game in $\mathbb{R}^n$ to understand adjoint method

given $(f_j, b_i) \in \mathbb{R}^n \ (i \in \{1, n_d\} \ j \in \{1, n_f\})$, given $A \in \mathcal{M}(\mathbb{R}^n)$

Calculate the values of $x_i.f_j \quad A x_i = b_i \quad i \in \{1, n_d\}$

- Solution solving $n_d$ linear systems
- Solution solving $n_f$ linear systems
Several ways of deriving the equations of discrete adjoint method. The following also helps understanding continuous adjoint

Following equalities hold \( \forall \lambda_k \in \mathbb{R}^{n_a} \)

\[
\lambda_k^T \frac{\partial R}{\partial W} dW + \lambda_k^T \left( \frac{\partial R}{\partial X} dX \right) = 0
\]

\[
\frac{dJ_k(\alpha)}{d\alpha_i} = \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha_i} + \frac{\partial J_k}{\partial W} \frac{dW}{d\alpha_i} + \lambda_k^T \frac{\partial R}{\partial W} dW + \lambda_k^T \left( \frac{\partial R}{\partial X} dX \right)
\]

\[
\frac{dJ_k(\alpha)}{d\alpha_i} = \left( \frac{\partial J_k}{\partial W} + \lambda_k^T \frac{\partial R}{\partial W} \right) \frac{dW}{d\alpha_i} + \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha_i} + \lambda_k^T \left( \frac{\partial R}{\partial X} dX \right)
\]
Vector $\lambda_k$ defined in order to cancel the factor of the flow sensitivity $\frac{dW}{d\alpha_i}$... the adjoint equation. $\lambda_k$ actually appears to be linked to functions $J_k$

$$\frac{\partial J_k}{\partial W} + \lambda_k^T \frac{\partial R}{\partial W} = 0$$

Calculation of derivatives

$$\forall i \in [1, n_d] \quad \frac{dJ_k(\alpha)}{d\alpha_i} = \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha_i} + \lambda_k^T \left( \frac{\partial R}{\partial X} \frac{dX}{d\alpha_i} \right)$$

$$\nabla_\alpha J_k(\alpha) = \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha} + \lambda_k^T \left( \frac{\partial R}{\partial X} \frac{dX}{d\alpha} \right)$$

Method with $n_f$ and not $n_d$ linear systems to solve
Other ways to derive the discrete adjoint equation

- Introduce a Lagrangian
- Manipulate direct differentiation gradient expression (like in the mathematical game)

From direct method gradient expression

\[ \nabla_\alpha J_k(\alpha) = \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha} + \frac{\partial J_k}{\partial W} \frac{dW}{d\alpha} \]

\[ \nabla_\alpha J_k(\alpha) = \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha} - \frac{\partial J_k}{\partial W} \left( \frac{dR}{dW} \right)^{-1} \frac{dR}{dX} \frac{dX}{d\alpha} \]

\[ \nabla_\alpha J_k(\alpha) = \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha} - \left( \frac{\partial J_k}{\partial W} \left( \frac{dR}{dW} \right)^{-1} \right) \frac{dR}{dX} \frac{dX}{d\alpha} \]

- Define \( \lambda_k \) column vector
Discrete adjoint parameter method (4/5)

From direct method gradient expression

\[
\nabla_{\alpha} J_k(\alpha) = \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha} - \left( \frac{\partial J_k}{\partial W} \left( \frac{dR}{dW} \right)^{-1} \right) \frac{dR}{dX} \frac{dX}{d\alpha}
\]

Define \( \lambda_k \)

\[
\lambda_k^T = -\frac{\partial J_k}{\partial W} \left( \frac{dR}{dW} \right)^{-1} \quad \text{or} \quad \lambda_k^T (\frac{dR}{dW}) = -\frac{\partial J_k}{\partial W} \quad \text{or} \quad \left( \frac{dR}{dW} \right)^T \lambda_k = -\frac{\partial J_k}{\partial W}^T
\]

Expression of sensitivity

\[
\nabla_{\alpha} J_k(\alpha) = \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha} + \lambda_k^T \frac{dR}{dX} \frac{dX}{d\alpha}
\]
CFD teams tend to mimic the solution of steady state flow although flow equations are non-linear whereas direct/adjoint equation are linear.

Storing the jacobian of the scheme and sending to direct solver has been done but is rare and is not tractable for large cases.

Iterative resolution is much more common. Newton/relaxation algorithm

\[
\left( \frac{\partial R}{\partial W} \right)^{(APP)}^T \left( \lambda_k^{(l+1)} - \lambda_k^{(l)} \right) = - \left( \left( \frac{\partial R}{\partial W} \right)^T \lambda_k^{(l)} + \left( \frac{\partial J_k}{\partial W} \right)^T \right)
\]
Iterative solution of direct and adjoint equation (2/3)

- Common Newton/relaxation algorithm for adjoint
  \[
  \left( \frac{\partial R}{\partial W} \right)^{(APP)} T \left( \chi^{(l+1)} - \chi^{(l)} \right) = - \left( \left( \frac{\partial R}{\partial W} \right)^T \lambda_k^{(l)} + \frac{\partial J_k}{\partial W} \right)^T
  \]

- Common Newton/relaxation algorithm for direct
  \[
  \left( \frac{\partial R}{\partial W} \right)^{(APP)} \left( \frac{dW}{d\alpha_i} \right)^{(l+1)} - \left( \frac{dW}{d\alpha_i} \right)^{(l)} = - \left( \left( \frac{\partial R}{\partial W} \right) \frac{dW}{d\alpha_i} + \frac{\partial R}{\partial X} \frac{dX}{d\alpha_i} \right)
  \]

- Defining an approximate Jacobians \( \left( \frac{\partial R}{\partial W} \right)^{(APP)} \) is an old subject in compressible CFD (definition of implicit stages for backward-Euler schemes...)
  - upwind approximate linearization of convective flux
  - neglecting cross derivatives in linearization of viscous fluxes...

- Possibly adapting implicit stages and mutigrid algorithm (flow solver to adjoint solver)
Discrete adjoint method

Iterative solution of direct and adjoint equation (3/3)

- Common Newton/relaxation algorithm for adjoint

\[
\left( \frac{\partial R}{\partial W} \right)^{(APP)^T} \left( \lambda_k^{(i+1)} - \lambda_k^{(i)} \right) = - \left( \left( \frac{\partial R}{\partial W} \right)^T \lambda_k^{(i)} + \frac{\partial J_k}{\partial W} \right)^T
\]

- Accuracy of adjoint vector only depends on \( \left( \frac{\partial R}{\partial W} \right) \). Only minor simplifications are allowed at this stage to preserve an acceptable accuracy

- Convergence towards solution of the linear system depends on \( \left( \frac{\partial R}{\partial W} \right) \), \( \left( \frac{\partial R}{\partial W} \right)^{(APP)} \), multigrid (if active), other operations like smoothing (if active)
Checking adjoint method... much more difficult than checking direct differentiation method. If

$$\frac{dJ_k}{d\alpha_i} \leftrightarrow \frac{J_k(W(\alpha + \delta\alpha_i), X(\alpha + \delta\alpha_i)) - J_k(W(\alpha - \delta\alpha_i), X(\alpha + \delta\alpha_i))}{2\delta\alpha_i}$$

no easy checking procedure

In the iterative resolution method, the gradient accuracy depends on the $(\frac{\partial R}{\partial W})^T \lambda_k^{(l)}$ operation

If direct mode is coded, duality checks between direct and adjoint code are useful. $(U, V)$ two column vectors of $\mathbb{R}^{na}$

$$U^T(\frac{\partial R}{\partial W})V = \left(U^T(\frac{\partial R}{\partial W})\right)_{adj-code} \cdot V = U^T \left((\frac{\partial R}{\partial W})V\right)_{lin-code}$$

Valid for individual fluxes routine. Valid for part of the interfaces (border, joins...)

J. Peter (ONERA DMFN)
Discrete adjoint method mesh method (1/3)

- Vector $\lambda_k$ defined by
  \[ \frac{\partial J_k}{\partial W} + \lambda_k^T \frac{\partial R}{\partial W} = 0 \]

- Calculation of derivatives
  \[ \forall i \in [1, n_f] \quad \frac{dJ_k(\alpha)}{d\alpha_i} = \frac{\partial J_k}{\partial X} \frac{dX}{d\alpha_i} + \lambda_k^T \left( \frac{\partial R}{\partial X} \frac{dX}{d\alpha_i} \right) \]
  \[ \forall i \in [1, n_f] \quad \frac{dJ_k(\alpha)}{d\alpha_i} = \left( \frac{\partial J_k}{\partial X} + \lambda_k^T \frac{\partial R}{\partial X} \right) \frac{dX}{d\alpha_i} \]

- Obvious mathematical factorization. Huge practical importance.
Solve for adjoint vectors

CFD gradient computation code computes “only”

\[
\frac{dJ_k}{dX} = \frac{\partial J_k}{\partial X} + \lambda_k^T \frac{\partial R}{\partial X}
\]

The functional outputs sensitivities \(dJ_k(\alpha)/d\alpha_i\) are calculated later by a mesh/geometrical tool

Pros : CFD has no knowledge of parametrization. Huge memory savings
[Nielsen, Park 2005] Try several parametrization. Check \((dJ_k/dS)\) with engineers

Cons : Matrix \((\partial R/\partial X)\) has to be explicitly computed (instead of \(\frac{\partial R}{\partial X} \frac{dX}{d\alpha_i}\) computable by finite differences) Hard work...
Solve for adjoint vectors. Compute “only”

\[
\frac{dJ_k}{dX} = \frac{\partial J_k}{\partial X} + \chi_k^T \frac{\partial R}{\partial X}
\]

Cons: Matrix \((\partial R/\partial X)\) has to be explicitly computed (instead of \(\frac{\partial R}{\partial X} \frac{dX}{d\alpha_i}\) computable by finite differences) Hard work...

How to calculate \((dJ_k/dS)\)?

- Explicit link between \(X\) and \(S\)

\[
\frac{dJ_k}{d\alpha_i} = \left[ \frac{dJ_k}{dX} \frac{dX}{dS} \right] \frac{dS}{d\alpha_i}
\]

- Implicit link between \(X\) and \(S\) [Nielsen, Park 2005]
Outline

1. Introduction
2. Discrete adjoint method
3. Continuous adjoint method
4. Discrete vs Continuous adjoint
5. Conclusions
Bibliography

- Mathematical references [Pironneau 73, 74]
- Mathematical aeronautical reference [Jameson 88]

- Simplest introduction [Giles, Pierce 99]
  *An introduction to the adjoint approach to design* ERCOFTAC Workshop on Adjoint Methods, Toulouse 1999.
Continuous Adjoint for toy problems (1/6)

- From [Giles, Pierce 99] section (3.2)
- Toy problems without design parameters

Solve

\[ \frac{du}{dx} - \epsilon \frac{d^2 u}{dx^2} = f \quad \text{on} \quad [0, 1] \quad u(0) = u(1) = 0 \]

before calculating

\[ J = (u, g) = \int_0^1 u \ g \ dx \]

Adjoint problem? Define (if it exists)

\[ L^* \lambda = g \quad \text{on} \quad [0, 1] \quad \text{plus boundary conditions} \]

such that

\[ J = (\lambda, f) = \int_0^1 \lambda \ f dx \]
Continuous Adjoint for toy problems (2/6)

- Direct: solve
  \[
  \frac{d u}{d x} - \epsilon \frac{d^2 u}{d x^2} = f \quad \text{on } [0, 1] \quad u(0) = u(1) = 0
  \]
  before calculating
  \[
  J = (u, g) = \int_0^1 u \cdot g \, dx
  \]

- Adjoint problem (if it exists):
  \[
  L^* \lambda = g \quad \text{on } [0, 1]
  \]
  plus boundary conditions such that
  \[
  J = (\lambda, f) = \int_0^1 \lambda \cdot f \, dx
  \]

- Defining equation \( L^* \)
Continuous Adjoint for toy problems (3/6)

- Defining equation $L^*$

\[
(\lambda, f) = \int_0^1 \lambda \, f \, dx = \int_0^1 \lambda \left( \frac{du}{dx} - \epsilon \frac{d^2u}{dx^2} \right) \, dx
\]

\[
(\lambda, f) = -\int_0^1 \frac{d\lambda}{dx} \, u \, dx + [\lambda \, u]_0^1 + \epsilon \int_0^1 \frac{d\lambda}{dx} \frac{du}{dx} \, dx - \epsilon \left[ \lambda \, \frac{du}{dx} \right]_0^1
\]

\[
(\lambda, f) = -\int_0^1 \frac{d\lambda}{dx} \, u \, dx + [\lambda \, u]_0^1 - \epsilon \int_0^1 \frac{d^2\lambda}{dx^2} \, du \, dx - \epsilon \left[ \lambda \, \frac{du}{dx} \right]_0^1 - \epsilon \left[ \frac{d\lambda}{dx} \, u \right]_0^1
\]

- Finally

\[
(\lambda, f) = \int_0^1 \left( \frac{d\lambda}{dx} - \epsilon \frac{d^2\lambda}{dx^2} \right) \, u \, dx + [\lambda \, u]_0^1 + \epsilon \left[ \frac{d\lambda}{dx} \, u \right]_0^1 - \epsilon \left[ \lambda \, \frac{du}{dx} \right]_0^1
\]

- Suitable adjoint equation. Solving

\[- \frac{d\lambda}{dx} - \epsilon \frac{d^2\lambda}{dx^2} = g \text{ on } [0, 1] \quad \lambda(0) = \lambda(1) = 0\]

ensures $(\lambda, f) = (u, g)$
In order to calculate $J = (u, g) = \int_\Omega u \, g \, d\Omega$, solve for $u$

$$\text{div}(k \, \text{grad}(u)) = f \quad \text{on} \quad \Omega \quad u = 0 \quad \text{on} \partial \Omega$$

In order to calculate $J$ as $(\lambda, f) = \int_\Omega \lambda \, f \, d\Omega$, solve for $\lambda$

$$\text{div}(k \, \text{grad}(\lambda)) = g \quad \text{on} \quad \Omega \quad \lambda = 0 \quad \text{on} \partial \Omega$$

Definition of adjoint operator comes from

$$(\lambda, f) = \int_\Omega u \, \text{div}(k \, \text{grad}(\lambda)) \, d\Omega - \int_{\partial \Omega} k \, u \, (\text{grad}(\lambda).n) \, dS + \int_{\partial \Omega} k \, \lambda \, (\text{grad}(u).n) \, dS$$
Continuous Adjoint for toy problems (5/6)

- **Direct:** solve
  \[
  \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = f \quad \text{on} \quad [0, L] \times [0, T] \quad u(0, .) = u(L, .) = 0 \quad u(., 0) = 0
  \]
  before calculating
  \[
  J = (u, g) = \int_0^L \int_0^T u g \, dxdt
  \]

- **Adjoint:** solve
  \[
  -\frac{\partial \lambda}{\partial t} - \frac{\partial^2 \lambda}{\partial x^2} = g \quad \text{on} \quad [0, L] \times [0, T] \quad \lambda(0, .) = \lambda(L, .) = 0 \quad \lambda(., T) = 0
  \]
  before calculating \( J \) as
  \[
  (\lambda, f) = \int_0^L \int_0^T \lambda \, f \, dxdt
  \]
Continuous Adjoint for toy problems (6/6)

- Time derivative $\frac{\partial u}{\partial t}$ gets $-\frac{\partial \lambda}{\partial t}$
  Backward time integration for unsteady adjoint

- Convection term $\frac{\partial u}{\partial x}$ gets $-\frac{\partial \lambda}{\partial x}$
  “Backward propagation” in adjoint steady state solutions

- Diffusion term $\frac{\partial^2 u}{\partial x^2}$ gets $\frac{\partial^2 \lambda}{\partial x^2}$
Continuous Adjoint for 2D Euler equations (1/11)

2D Euler equations

\[
\frac{\partial w}{\partial t} + \frac{\partial f(w)}{\partial x} + \frac{\partial g(w)}{\partial y} = 0
\]

avec

\[
w = \begin{pmatrix}
\rho \\
\rho u \\
\rho v \\
\rho E
\end{pmatrix} \quad f(w) = \begin{pmatrix}
\rho u \\
\rho u^2 + p \\
\rho uv \\
\rho Hu
\end{pmatrix} \quad g(w) = \begin{pmatrix}
\rho v \\
\rho uv \\
\rho v^2 + p \\
\rho Hv
\end{pmatrix}
\]

\[p = (\gamma - 1)\rho(E - \frac{u^2 + v^2}{2}), \quad \rho H = \rho E + p\]
Continuous Adjoint for 2D Euler equations (2/11)

Figure: Coordinate transformation for airfoil-fitted structured mesh

Coordinate transformation $\Gamma$, $C^1$ diffeomorphism $D_{\xi \eta} = [\xi_{min}, \xi_{max}] \times [\eta_{min}, \eta_{max}]$ en $D_w$.

$\Gamma \left\{ \begin{array}{c}
D_{\xi \eta} \\
(\xi, \eta) \\
\rightarrow \\
D_{xy} \\
\rightarrow \\
(x, y)
\end{array} \right.$
Continuous Adjoint for 2D Euler equations (3/11)

Figure: Normal surface vectors
Continuous Adjoint for 2D Euler equations (4/11)

- 2D Euler equations generalized coordinates

\[
K = \left( \frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi} \frac{\partial x}{\partial \eta} \right)
\]

\[
\begin{pmatrix} U \\ V \end{pmatrix} = \frac{1}{K} \left( \begin{pmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial x}{\partial \eta} \\ -\frac{\partial y}{\partial \xi} & \frac{\partial x}{\partial \xi} \end{pmatrix} \right) \begin{pmatrix} u \\ v \end{pmatrix}
\]

\[
W = K \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}
\]

\[
F(W) = K \begin{pmatrix} \rho U \\ \rho Uu + p \frac{\partial \xi}{\partial \xi} \\ \rho Uv + p \frac{\partial \xi}{\partial \eta} \\ \rho UH \end{pmatrix}
\]

\[
G(W) = K \begin{pmatrix} \rho V \\ \rho Vu + p \frac{\partial \eta}{\partial \xi} \\ \rho Vv + p \frac{\partial \eta}{\partial \eta} \\ \rho VH \end{pmatrix}
\]

\[
\frac{\partial W}{\partial t} + \frac{\partial F(W)}{\partial \xi} + \frac{\partial G(W)}{\partial \eta} = 0
\]
Continuous Adjoint for 2D Euler equations (5/11)

- Steady state equation can also be rewritten as

\[
\frac{\partial}{\partial \xi} \left( f \frac{\partial y}{\partial \eta} - g \frac{\partial x}{\partial \eta} \right) + \frac{\partial}{\partial \eta} \left( -f \frac{\partial y}{\partial \xi} + g \frac{\partial x}{\partial \xi} \right) = 0 \quad \text{sur} \quad D_{\xi \eta}
\]

- Jacobians per mesh directions:

\[
a(w) = \frac{df(w)}{dw} \quad b(w) = \frac{dg(w)}{dw}
\]

\[
a_1(w, \xi, \eta) = \left( a(w) \frac{\partial y}{\partial \eta} - b(w) \frac{\partial x}{\partial \eta} \right) \quad a_2(w, \xi, \eta) = \left( -a(w) \frac{\partial y}{\partial \xi} + b(w) \frac{\partial x}{\partial \xi} \right)
\]
Continuous Adjoint for 2D Euler equations (6/11)

- Coordinate transformation now depending on a design parameter $\alpha$ (for the sake of simplicity scalar)

$$
\Gamma \left\{ \begin{array}{c}
D_{\xi\eta} D_\alpha \\
(\xi, \eta)(\alpha)
\end{array} \right\} \rightarrow D_w \\
\rightarrow (x(\xi, \eta, \alpha), y(\xi, \eta, \alpha))
$$

- $D_w$ changes with $\alpha$ but not $D_{\xi\eta}$
- Equation for $dW/d\alpha_i$ ?
Continuous Adjoint for 2D Euler equations (7/11)

- Variations induced by $d\alpha$ change

\[
\begin{align*}
  f(w) & \quad \rightarrow \quad f(w) + \frac{df}{dw} \frac{dw}{d\alpha_i} d\alpha_i \\
  \frac{\partial x}{\partial \eta} & \quad \rightarrow \quad \frac{\partial x}{\partial \eta} + \frac{\partial^2 x}{\partial \eta \partial \alpha_i} d\alpha_i
\end{align*}
\]

- Fluid dynamics equations on the fixed domain $D_{\xi \eta}$

\[
\forall \quad \alpha \in D_\alpha \quad \frac{\partial}{\partial \xi} \left( f \frac{\partial y}{\partial \eta} - g \frac{\partial x}{\partial \eta} \right) + \frac{\partial}{\partial \eta} \left( -f \frac{\partial y}{\partial \xi} + g \frac{\partial x}{\partial \xi} \right) = 0 \quad \text{on} \quad D_{\xi \eta}
\]

- Differentiate w.r.t. $\alpha$
Continuous adjoint method

Continuous Adjoint for 2D Euler equations (8/11)

- Continuous direct differentiation equation

\[
\frac{\partial}{\partial \xi} \left( a_1(w, \xi, \eta) \frac{dw}{d\alpha} \right) + \frac{\partial}{\partial \eta} \left( a_2(w, \xi, \eta) \frac{dw}{d\alpha} \right) + \\
\frac{\partial}{\partial \xi} \left( f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha} \right) + \frac{\partial}{\partial \eta} \left( -f(w) \frac{\partial^2 y}{\partial \xi \partial \alpha} + g(w) \frac{\partial^2 x}{\partial \xi \partial \alpha} \right) = 0
\]

- Objective function (fixed domain \( D_{\xi\eta} \))

\[
\mathcal{J}(\alpha) = \int_{\xi_{\min}} J_1(w) d\eta + \int_{D_{\xi\eta}} J_2(w) d\xi d\eta
\]

- Derivative of the objective function (fixed domain \( D_{\xi\eta} \))

\[
\frac{d\mathcal{J}(\alpha)}{d\alpha} = \int_{\xi_{\min}} \frac{dJ_1(w)}{dw} \frac{dw}{d\alpha} d\eta + \int_{D_{\xi\eta}} \frac{dJ_2(w)}{dw} \frac{dw}{d\alpha} d\xi d\eta
\]
Continuous Adjoint for 2D Euler equations (9/11)

- continuous direct differentiation equation is multiplied by $\psi$, $C^1$, periodic in $\eta_{min}, \eta_{max}$

\[ \forall \psi \in C^1(D_{\xi \eta})^4 \int_{D_{\xi \eta}} \psi^T \left( \frac{\partial}{\partial \xi} \left( a_1(w, \xi, \eta) \frac{dw}{d\alpha} \right) + \frac{\partial}{\partial \eta} \left( a_2(w, \xi, \eta) \frac{dw}{d\alpha} \right) \right) d\xi d\eta + \int_{D_{\xi \eta}} \psi^T \left( f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha} \right) + \frac{\partial}{\partial \eta} \left( -f(w) \frac{\partial^2 y}{\partial \xi \partial \alpha} + g(w) \frac{\partial^2 x}{\partial \xi \partial \alpha} \right) \right) d\xi d\eta = 0 \]

- Integration by parts

\[ - \int_{D_{\xi \eta}} \frac{\partial \psi^T}{\partial \xi} a_1(w, \xi, \eta) \frac{dw}{d\alpha} d\xi d\eta - \int_{D_{\xi \eta}} \frac{\partial \psi^T}{\partial \eta} a_2(w, \xi, \eta) \frac{dw}{d\alpha} d\xi d\eta + \]

\[ - \int_{D_{\xi \eta}} \frac{\partial \psi^T}{\partial \xi} \left( f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha} \right) d\xi d\eta \]

\[ - \int_{D_{\xi \eta}} \frac{\partial \psi^T}{\partial \eta} \left( -f(w) \frac{\partial^2 y}{\partial \xi \partial \alpha} + g(w) \frac{\partial^2 x}{\partial \xi \partial \alpha} \right) d\xi d\eta \]

\[ + \int_{\xi_{min}} \psi^T a_1(w, \xi, \eta) \frac{dw}{d\alpha} d\eta + \int_{\xi_{min}} \psi^T \left( f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha} \right) d\eta = 0. \]
Continuous Adjoint for 2D Euler equations (10/11)

- Gradient of objective function for all $\psi$ function of $C_1^1(D_{\xi\eta})^4$

\[
\frac{dJ(\alpha)}{d\alpha} = \int_{\xi_{\text{min}}} \frac{dJ_1(w)}{dw} \frac{dw}{d\alpha} d\eta + \int_{D_{\xi\eta}} \frac{dJ_2(w)}{dw} \frac{dw}{d\alpha} d\xi d\eta
\]

- \[
\int_{D_{\xi\eta}} \frac{\partial \psi}{\partial \xi} a_1(w, \xi, \eta) \frac{dw}{d\alpha} d\xi d\eta - \int_{D_{\xi\eta}} \frac{\partial \psi}{\partial \eta} a_2(w, \xi, \eta) \frac{dw}{d\alpha} d\xi d\eta +
\]

- \[
\int_{D_{\xi\eta}} \frac{\partial \psi}{\partial \xi} \left( f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha} \right) d\xi d\eta
\]

- \[
\int_{D_{\xi\eta}} \frac{\partial \psi}{\partial \eta} \left( -f(w) \frac{\partial^2 y}{\partial \xi \partial \alpha} + g(w) \frac{\partial^2 x}{\partial \xi \partial \alpha} \right) d\xi d\eta
\]

- \[
\int_{\xi_{\text{min}}} \psi^T a_1(w, \xi, \eta) \frac{dw}{d\alpha} d\eta + \int_{\xi_{\text{min}}} \psi^T \left( f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha} \right) d\eta
\]

- $\psi$ chosen so as to cancel all flow sensitivity terms

\[
\begin{cases}
\frac{dJ_2(w)}{dw} - \frac{\partial \psi}{\partial \xi} a_1(w, \xi, \eta) - \frac{\partial \psi}{\partial \eta} a_2(w, \xi, \eta) = 0 & \text{over } D_{\xi,\eta} \\
\psi^T a_1(w, \xi, \eta) + \frac{dJ_1(w)}{dw} = 0 & \text{on } \xi_{\text{min}}
\end{cases}
\]
Final form of objective gradient ($\psi$ being the solution of continuous adjoint equation)

$$\frac{dJ(\alpha)}{d\alpha_i} = \int_{\xi_{\text{min}}} \psi^T \left( f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha_i} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha_i} \right) d\eta$$

$$- \int_{D_{\xi \eta}} \frac{\partial \psi^T}{\partial \xi} \left( f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha_i} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha_i} \right) d\xi d\eta$$

$$- \int_{D_{\xi \eta}} \frac{\partial \psi^T}{\partial \eta} \left( -f(w) \frac{\partial^2 y}{\partial \xi \partial \alpha_i} + g(w) \frac{\partial^2 x}{\partial \xi \partial \alpha_i} \right) d\xi d\eta$$

Just as for discrete adjoint, one adjoint field for one function of interest and design parameters

Partial differential equation which derivation exceeds level of maths ordinarily used by engineers

Equation to be discretized to get numerical values
Some intuitions about adjoint vector? (1/7)

Could I get some intuition about adjoint vector?

Try again with continuous adjoint!

- Rewrite flow equation locally, neglecting metric derivatives terms

\[
\frac{\partial}{\partial \xi} \left( f \frac{\partial y}{\partial \eta} - g \frac{\partial x}{\partial \eta} \right) + \frac{\partial}{\partial \eta} \left( -f \frac{\partial y}{\partial \xi} + g \frac{\partial x}{\partial \xi} \right) = 0 \quad \text{sur} \quad D_{\xi \eta}
\]

\[
a(w) = \frac{df(w)}{dw} \quad b(w) = \frac{dg(w)}{dw}
\]

\[
a_1(w, \xi, \eta) = \left( a(w) \frac{\partial y}{\partial \eta} - b(w) \frac{\partial x}{\partial \eta} \right) \quad a_2(w, \xi, \eta) = \left( -a(w) \frac{\partial y}{\partial \xi} + b(w) \frac{\partial x}{\partial \xi} \right)
\]
Some intuitions about adjoint vector? (1/7)

Could I get some intuition about adjoint vector?

Try again with continuous adjoint!

Rewrite flow equation locally, neglecting metric derivatives terms

$$\frac{\partial}{\partial \xi} \left( f \frac{\partial y}{\partial \eta} - g \frac{\partial x}{\partial \eta} \right) + \frac{\partial}{\partial \eta} \left( -f \frac{\partial y}{\partial \xi} + g \frac{\partial x}{\partial \xi} \right) = 0 \quad \text{sur} \quad D_{\xi\eta}$$

$$a(w) = \frac{df(w)}{dw} \quad b(w) = \frac{dg(w)}{dw}$$

$$a_1(w, \xi, \eta) = \left( a(w) \frac{\partial y}{\partial \eta} - b(w) \frac{\partial x}{\partial \eta} \right) \quad a_2(w, \xi, \eta) = \left( -a(w) \frac{\partial y}{\partial \xi} + b(w) \frac{\partial x}{\partial \xi} \right)$$
Could I get some intuition about adjoint vector?
Trying again based on continuous adjoint

Rewrite flow equation locally, neglecting metric derivatives terms

\[ a_1(w, \xi, \eta) \frac{\partial w}{\partial \xi} + a_2(w, \xi, \eta) \frac{\partial w}{\partial \eta} = 0 \]

Reminder adjoint equation

\[ \frac{dJ_2(w)}{dw} = - \frac{\partial \psi^T}{\partial \xi} a_1(w, \xi, \eta) - \frac{\partial \psi^T}{\partial \eta} a_2(w, \xi, \eta) = 0 \]

Change of sign, transposed jacobians, source term.

Hyperbolic system. Same conditions for existence of simple wave solutions
\[ \psi(\xi, \eta) = \psi(a\xi + b\eta)V, \text{ propagation par convection. Number of solutions for subsonic/supersonic flow...} \]
Continuous adjoint method

Some intuitions about adjoint vector? (3/7)

- Supersonic inviscid flow $M_\infty = 1.5$ $\text{AoA} = 1^\circ$

**Figure:** 513 × 513 mesh
Some intuitions about adjoint vector ? (5/7)

- Supersonic inviscid flow $M_\infty = 1.5$ AoA = $1^\circ$

Figure: iso-lines of Mach number
Some intuitions about adjoint vector? (6/7)

- Supersonic inviscid flow $M_\infty = 1.5$ $\text{AoA} = 1^\circ$

**Figure:** First component of adjoint vector for $C_Dp$
Some intuitions about adjoint vector? (6/7)

- Supersonic inviscid flow $M_\infty = 1.5 \, \text{AoA} = 1^\circ$

Figure: First component of adjoint vector for $CD_p$ (close view)
Continuous adjoint method

Some intuitions about adjoint vector? (7/7)

- Supersonic inviscid flow $M_\infty = 1.5$ AoA = 1°

Figure: Fourth component of adjoint vector for $CD_p$
Some intuitions about adjoint vector? (7/7)

- Supersonic inviscid flow $M_\infty = 1.5$ $\alpha = 1^\circ$

Figure: Fourth component of adjoint vector for $C_Dp$ (close view)
Outline

1. Introduction
2. Discrete adjoint method
3. Continuous adjoint method
4. Discrete vs Continuous adjoint
5. Conclusions
Discrete vs Continuous adjoint

Discrete adjoint

- **Assets**
  - calculates what you want = sensitivity of your code
  - can deal with all types of functions
  - code can be partly built by AD (automatic differentiation)
  - higher order derivatives simple (not too complex) in a discrete framework

- **Drawbacks**
  - no understanding of underlying physics (Euler flows...)
  - numerical consistency with a set of pde? Dissipative scheme for this set of pde?
Discrete adjoint

- **Assets**
  - get physical understanding of underlying equations (with all following restrictions)
  - codes a dissipative discretization of underlying equation
  - the code is shorter and simpler than the one of discrete adjoint

- **Drawbacks**
  - does not calculate the sensitivity of your direct (steady state) code
  - no reason that continuous adjoint equations would exist for all types of initial pde
  - can not deal with far-field functions
Coexistence of continuous and discrete adjoint

• Coexistence comes from the fact that their assets are balanced
• Continuous more suitable for theoretical mechanics
• Probably discrete more suitable for practical applications
More material can be found in *Numerical sensitivity analysis for aerodynamic optimization: A survey of approaches. Computers and Fluids* 39 J.P. & RP Dwight 2010

- Second order derivatives, frozen turbulence and other approximations, discretization of the continuous adjoint equation...

Twenty-six years after [Jameson 88] famous article...

- All large CFD code in aeronautics have an adjoint module
- Some robustness issues to be solved
- Compatibility with some complex options of direct code possibly missing
- Integration in automated local shape optimization requires adjoint enhanced robustness and CAD/parametrization issue to be solved
- Numerous successful adjoint-based local optimizations and goal-oriented adaptations