Tangent/Adjoint Basics

Jay Sitaraman

Date: 8 Jun 202

Tangents/ Adjoints

- **Gradients from high-fidelity CFD codes are useful for gradient based optimization**
  - Naïve way of computing gradients is finite difference around the current state
  - For steady state calculation, this is a still feasible approach, since number of non-linear iterations required to reach a new steady state with a perturbed design variable is quite a bit smaller than that require for the initial steady state calculation
  - For time dependent calculations, finite difference is the only robust way, at least for real problems.
  - Tangents/Adjoints need differentiation of the code base such that the objectives as well as the sensitivities are computed in one shot
- **Tangent method is preferred if there are several objectives and few design variables**
- **Adjoint method is preferrable when there are few objectives (usually just one) and large number of design variables.**

Multi-variable Calculus Recap

- **Total Derivatives and Partial derivatives**
  - Total derivatives include implicit dependence
  - Partial derivatives include only explicit dependence

Illustrative example

$$F(x,y) = 3y^3 + x$$
$$x = y^2$$

$$\frac{\partial F}{\partial x} = 1$$

$$\frac{\partial F}{\partial y} = 9y^2$$

$$\frac{dF}{dy} = \frac{\partial F}{\partial y} + \frac{\partial F}{\partial x}\frac{dx}{dy}$$

$$\frac{dF}{dy} = 9y^2 + 2y$$

Partial derivative

Total derivative

Sensitivity Analysis Problem

- **Examples of standard cost functions could be Lift, Drag, L/D, Figure of Merit, Surface heating etc..**
- **Design variables are generally shapes, operating conditions, boundary conditions, model constants etc..**

- $L(x, q, D)$ **= cost function, usually a scalar**
  - $x = grid\ (kN, 1), k = number\ of\ dimensions$
  - $q = \ flow\ field\ (fN, 1), f = number\ of\ fields$
  - $D = \ design\ variables\ (\mathrm{d}, 1)$
- $\frac{dL}{dD}$ **= sensitivities to design variables, row vector of size** $[1, \boldsymbol{d}\ ]$
- **An optimizer is usually the outer loop and determines new design variables based on current obj function value and its sensitivity to design variables**

Tangent Math

- **Cost function:** $L(x, q, D)$
- $\frac{dL}{dD} = \frac{\partial L}{\partial D} + \frac{\partial L}{\partial x}\frac{\boldsymbol{dx}}{\boldsymbol{dD}} + \frac{\partial L}{\partial q}\frac{\boldsymbol{dq}}{\boldsymbol{dD}}$

These are unknown

  - Chain rule of partial sensitivities
  - Matrix dimensionality : $[(1,d)] = [(1,d)] + [(1,kN)][(kN,d)] + [(1,fN)][(fN,d)]$
  - First term is explicit contribution of the design variables to the cost function, e.g., if the design variable is a boundary condition, you will have explicit dependence (more on this in the example)
  - Second term is a product of a row-vector times a matrix
    - Row-vector: dependence of cost function on grid locations of the degrees of freedom (mostly sparse, except for the boundary grid points that contribute to the force integration)
    - Matrix : dependence on volume grid locations with design variables
      - if you have algebraic grid deformation this can be found analytically
      - Can finite-difference if grid generation is quick
      - Can solve another adjoint problem if volume grid is solution of a PDE (linear elasticity/spring analogy) etc
  - Third term is a product of row-vector and a matrix
    - Row-vector: dependence of cost function on flow field (again mostly sparse, only depends on the fields at the boundary for most cases)
    - Matrix: unknown and need to be solved for

Tangent Math (continued)

- **Solution residuals must remain zero (ideally), no matter what the design variables are.. This is a constraint**
- $\boldsymbol{constraints}$: $R(x, q, D) = 0,\ [(\boldsymbol{fN}, \mathbf{1})] = \mathbf{0}$
- $\frac{dR}{dD} = 0, [(N, d)] = 0$
- $\frac{dR}{dD} = \frac{\partial R}{\partial D} + \frac{\partial R}{\partial x}\frac{\boldsymbol{dx}}{\boldsymbol{dD}} + \frac{\partial R}{\partial q}\frac{\boldsymbol{dq}}{\boldsymbol{dD}},\ \ [(\boldsymbol{fN}, \boldsymbol{d})] = [(\boldsymbol{fN}, \boldsymbol{d})] + [(\boldsymbol{fN}, \boldsymbol{kN})][(\boldsymbol{kN}, \boldsymbol{d})] + [(\boldsymbol{fN}, \boldsymbol{fN})][(\boldsymbol{kN}, \boldsymbol{d})]$
  - First term is explicit contribution of the design variables to the residuals, e.g., if the design variable is a boundary condition
  - Second term is a product of two matrices
    - Matrix1 : dependence on volume grid locations with design variables (same term as previous slide)
    - Matrix2 : dependence of residual on grid locations, Residual -> metrics -> grid locations
  - Third term is a product of two matrices
    - Matrix1 : Jacobian of the residual vector to field variables
    - Matrix2 : Unknown from the previous slide

Tangent problem continued..

- **Tangent problem**

  1) $\frac{dR}{dD} = \frac{\partial R}{\partial D} + \frac{\partial R}{\partial x}\frac{dx}{dD} + \frac{\partial R}{\partial q}\frac{dq}{dD} = 0$ (constraint problem)

  2) $\frac{\partial q}{\partial D} = -\left[\frac{\partial R}{\partial q}\right]^{-1}\left[\frac{\partial R}{\partial D} + \frac{\partial R}{\partial x}\frac{dx}{dD}\right]$ $(\boldsymbol{solve\ for\ solution\ tangent})$

  3) $\frac{dL}{dD} = \frac{\partial L}{\partial D} + \frac{\partial L}{\partial x}\frac{dx}{dD} + \frac{\partial L}{\partial q}\frac{dq}{dD}$ $(\boldsymbol{original\ sensitivity\ equation})$

  4) $\frac{dL}{dD} = \frac{\partial L}{\partial D} + \frac{\partial L}{\partial x}\frac{dx}{dD} - \frac{\partial L}{\partial q}\left[\frac{\partial R}{\partial q}\right]^{-1}\left[\frac{\partial R}{\partial D} + \frac{\partial R}{\partial x}\frac{dx}{dD}\right]$ $(\boldsymbol{substitute\ for}\ \frac{dq}{dD}$ from 2)

- **Equation two is the same linear problem that most CFD codes solve every iteration, can use sparse matrix methods, But the Jacobian has to be exact, (not approximate). Can use the approximate Jacobian and find the full Jacobian mat-vec product using residual subtraction.**
- **Tangent is feasible for simple problems where the sensitivity of the mesh w.r.t design variables can be computed fast such that the products $[\frac{\partial L}{\partial x}\frac{dx}{dD}]$ and $[\frac{\partial R}{\partial x}\frac{dx}{dD}]$ can be computed on the fly**
- **Also, the inversion problem (last term of equation above) requires as many inversions as there are design variables,**
- **Linear problem is :** $\left[\frac{\partial R}{\partial q}\right]\left[\frac{\partial q}{\partial D}\right] = -\left[\frac{\partial R}{\partial D} + \frac{\partial R}{\partial x}\frac{dx}{dD}\right],\ [(\boldsymbol{f}N, \boldsymbol{f}N)][(\boldsymbol{f}N, d)] = [(\boldsymbol{f}N, d)]$

Adjoint problem

- $\frac{dL}{dD} = \frac{\partial L}{\partial D} + \frac{\partial L}{\partial x}\frac{\partial x}{\partial D} - \frac{\partial L}{\partial q}\left[\frac{\partial R}{\partial q}\right]^{-1}\left[\frac{\partial R}{\partial D} + \frac{\partial R}{\partial x}\frac{\partial x}{\partial D}\right]$ $(\boldsymbol{Tangent\ problem\ from\ last\ slide}$
- $\left[\frac{dL}{dD}\right]^T = \frac{\partial \mathrm{L}}{\partial \mathrm{D}}^{\mathrm{T}} + \frac{\partial \mathrm{x}}{\partial \mathrm{D}}^{T}\left[\frac{\partial \mathrm{L}}{\partial \mathrm{x}}\right]^T - \left[\frac{\partial \mathrm{R}}{\partial \mathrm{D}} + \frac{\partial \mathrm{R}}{\partial \mathrm{x}}\frac{\partial \mathrm{x}}{\partial \mathrm{D}}\right]^T\left[\frac{\partial \mathrm{R}}{\partial \mathrm{q}}\right]^{-T}\frac{\partial \mathrm{L}}{\partial \mathrm{q}}^{\mathrm{T}}$ $(\boldsymbol{Transpose\ of\ the\ equation\ above})$
  - $[(d,1)] = [(d,1)] + [(d,kN)][(kN,1)] + [(d,fN)][(fN,fN)]^{-1}[(fN,1)]$
- **Adjoint problem**
  - $\left[\frac{\partial \mathrm{R}}{\partial \mathrm{q}}\right]^T \Lambda_q = \left[\frac{\partial L}{\partial q}\right]^T$

    $[(fN,fN)][(fN,1)] = [(fN,1)]$
- **With the adjoint formulation, just one linear problem needs to be solved and the entire sensitivity can then be arranged with matrix vector products.**
- **One analysis solution followed by one adjoint solution can yield a method for computing sensitivities to arbitrary number of design variables at the same cost.**
- **In the formulation above, we did however assume that the products $\frac{\partial \mathrm{x}}{\partial \mathrm{D}}^{T}\left[\frac{\partial \mathrm{L}}{\partial \mathrm{x}}\right]^T$ and $\frac{\partial \mathrm{R}}{\partial \mathrm{x}}\frac{\partial \mathrm{x}}{\partial \mathrm{D}}$ can be arranged easily on the fly. For algebraic mesh deformation (i.e., with no iterative solution necessary), this is usually the case. But most real problems require an elastic deformation methodology that solves a set of equations (linear elasticity, spring analogy, RBF.. etc)**

Adjoint for a general steady state shape design problem

$\mathrm{L}(x, q, D)$ Cost function

$\mathrm{G}(x, x_s) = 0$ Grid deformation residual

$\mathrm{R}(x, q, D) = 0$ Flow field residual

x = grid
q= flow variables
D= design variables

$$\left[\frac{dL}{dD}\right]^T = \left[\frac{\partial L}{\partial D}\right]^T + \left[\left(\frac{dx}{dD}\right)^T \quad \left(\frac{dq}{dD}\right)^T\right] \begin{bmatrix} \frac{\partial L}{\partial x}^T \\ \\ \frac{\partial L}{\partial q}^T \end{bmatrix} \tag{1}$$

$$\begin{bmatrix} \left(\frac{\partial G}{\partial x}\right) & 0 \\ \\ \left(\frac{\partial R}{\partial x}\right) & \left(\frac{\partial R}{\partial q}\right) \end{bmatrix} \begin{bmatrix} \left(\frac{dx}{dD}\right) \\ \\ \left(\frac{dq}{dD}\right) \end{bmatrix} = \begin{bmatrix} -\left(\frac{\partial G}{\partial x_s}\right)\left(\frac{dx_s}{dD}\right) \\ \\ -\left(\frac{\partial R}{\partial D}\right) \end{bmatrix} \tag{2}$$

$$\left[\left(\frac{dx}{dD}\right)^T \quad \left(\frac{dq}{dD}\right)^T\right] = \left[-\left(\frac{\partial R}{\partial D}\right)^T \quad -\left(\frac{dx_s}{dD}\right)^T\left(\frac{\partial G}{\partial x_s}\right)^T\right] \begin{bmatrix} \left(\frac{\partial G}{\partial x}\right)^T & \left(\frac{\partial R}{\partial x}\right)^T \\ \\ 0 & \left(\frac{\partial R}{\partial q}\right)^T \end{bmatrix}^{-1} \tag{3}$$

$$\left[\frac{dL}{dD}\right]^T = \left[\frac{\partial L}{\partial D}\right]^T + \left[-\left(\frac{\partial R}{\partial D}\right)^T \quad -\left(\frac{dx_s}{dD}\right)^T\left(\frac{\partial G}{\partial x_s}\right)^T\right] \begin{bmatrix} \left(\frac{\partial G}{\partial x}\right)^T & \left(\frac{\partial R}{\partial x}\right)^T \\ \\ 0 & \left(\frac{\partial R}{\partial q}\right)^T \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial L}{\partial x}^T \\ \\ \frac{\partial L}{\partial q}^T \end{bmatrix} \tag{4}$$

$$\left[\frac{\partial R}{\partial q}\right]^T \Lambda_q = \left[\frac{\partial L}{\partial q}\right]^T$$ (Flow adjoint)

$$\left[\frac{\partial G}{\partial x}\right]^T \Lambda_x = \left[\frac{\partial L}{\partial x}\right]^T - \left[\frac{\partial R}{\partial q}\right]^T \Lambda_q$$ (Grid adjoint)

$$\frac{\partial G}{\partial x_s}^T = -I$$ (Grid equations sensitivity to surface grid is just identity)

$$\frac{dL}{dD}^T = \frac{\partial L}{\partial D}^T - \frac{\partial R}{\partial D}^T \Lambda_q + \frac{dx_s}{dD}^T \Lambda_x$$ (total sensitivity)

Steps

1. Deform surface mesh using design variables
2. Generate/Deform volume mesh
3. Solve analysis problem (Non-linear)
4. Solve Flow adjoint problem (Linear)
5. Solve Grid adjoint problem (Linear)
6. Assemble adjoints to create sensitivity vector

Classroom example of adjoint method

- **Heat equation**

$$x \in [0,1], y \in [0,1]$$

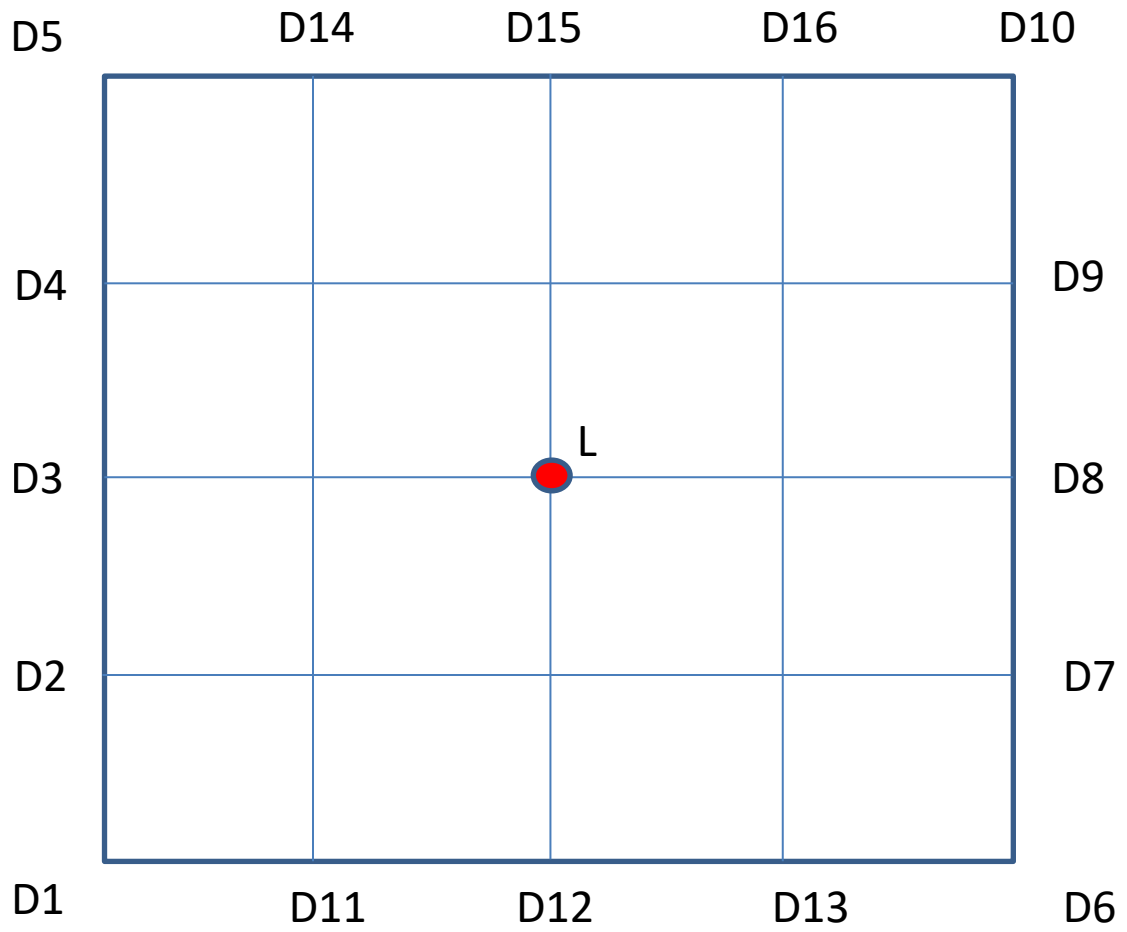
$$\nabla^2 q(x,y) = 0$$

$$q = D \quad x, y \in \partial\Omega$$

$$L = q(0.5, 0.5)$$

$$R(q,D) : q_{i,j} - 0.25(q_{i+1,j} + q_{i-1,j} + q_{i,j-1} + q_{i,j+1}) = 0$$

What is the sensitivity w.r.t boundary conditions of the Laplace solution at the mid point of the domain ?

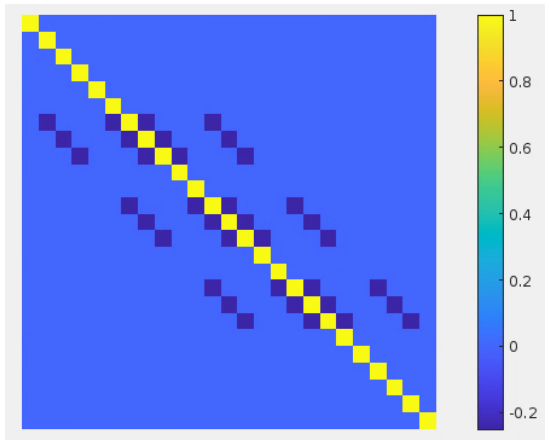


Heat equation example contd..

$$R(q, D) = [A]q - B(D)$$

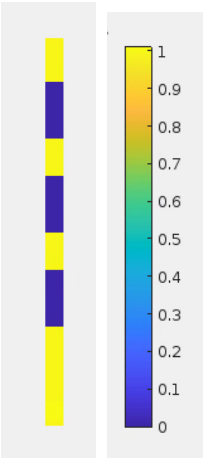
$$\frac{\partial R}{\partial q} = [A]$$

A=



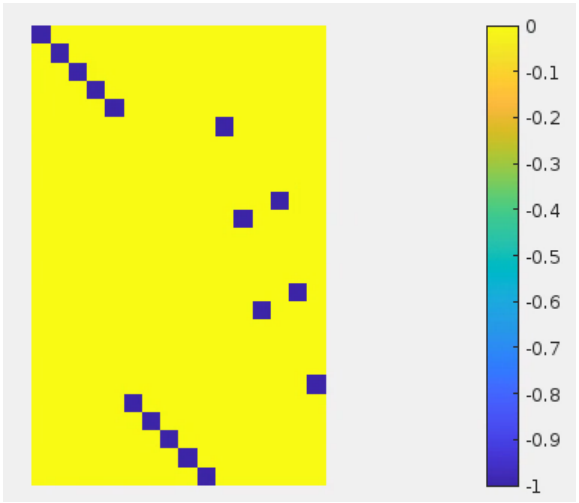
25 x 25

B=



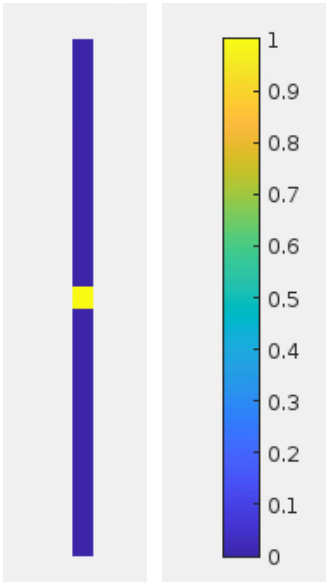
25 x 1

$$\frac{\partial R}{\partial D} =$$



25 x 16

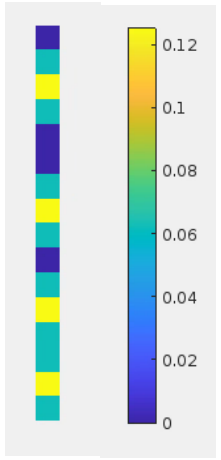
$$\frac{\partial L}{\partial q} =$$

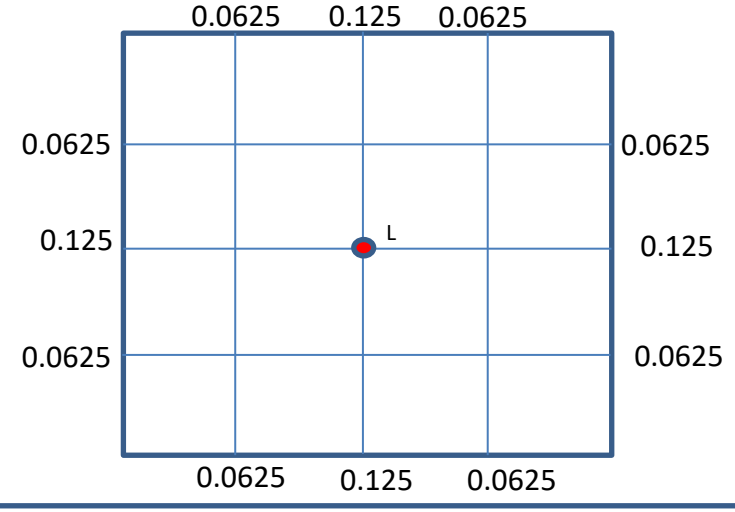


25 x1

$$\frac{dL}{dD}^T = -\left[\frac{\partial R}{\partial D}\right]^T \left[\left(\frac{\partial R}{\partial q}\right)^T\right]^{-1} \left[\frac{\partial L}{\partial q}\right]$$

$=$





https://github.com/jsitaraman/adjoint_heat.git