Building an Adjoint code (discrete approach)

January 14th 2014, Politecnico di Milano

Jan Pralits
DICCA, Università di Genova
jan.pralits@unige.it

Example: building the adjoint code

Let start with a linear system

$Ax = b$ (A and b are known)

A simple program is written

$x = A^{-1}b$

This is a linear operation with

- input b
- output x

Example: building the adjoint code

Direct

Initialize var1

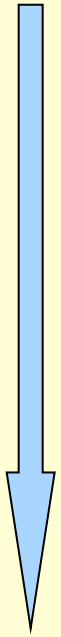
Sub1(var1,var2)

Sub2(var2,var3)

Sub3(var3,var4)

Sub4(var5,var6)

Solution=var6



The program that solves the direct equation (marching) is a sequence of linear operations which are executed in a given order.

I

Sub1(var1,var2)

var2=**M** • var1

End Sub1

input --------- output ---------

Example: building the adjoint code

Direct

Initialize var1

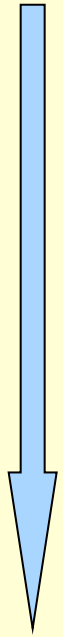
Sub1(var1,var2)

Sub2(var2,var3)

Sub3(var3,var4)

Sub4(var5,var6)

Solution=var6



The program that solves the direct equation (marching) is a sequence of linear operations which are executed in a given order. If we want to solve the adjoint equation we have to derive the adjoint of our direct code. This can be done by writing down the adjoint of each direct subroutine (which performs a linear operation between the input and the output) and then calling them in a reversed sequence.

input --------- output ---------

Example: building the adjoint code

Direct

Initialize var1

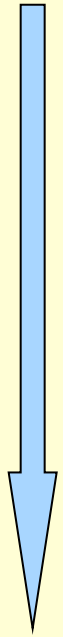
Sub1(var1,var2)

Sub2(var2,var3)

Sub3(var3,var4)

Sub4(var5,var6)

Solution=var6



Adjoint

Solution = avar1

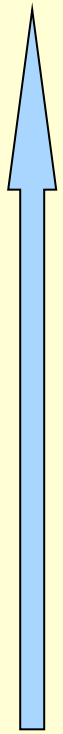
AdjSub1(avar1,avar2)

AdjSub2(avar2,avar3)

AdjSub3(avar3,avar4)

AdjSub4(avar5,avar6)

Initialize avar6



input --------- output ---------

A simple example: the adjoint of a given subroutine

```
I
Sub1(var1,var2)

    var2=M • var1

End Sub1
```

```
AdjSub1(avar1,avar2)

    avar1=M^t • avar2

End AdjSub1
```

Each subroutine performs a linear operation between the input and the output and can be represented through the action of a matrix **M**.

In a real code, however, this operation can be implemented through a complicated sequence of operations and loops. So care must be taken in deriving the correct calling sequence of operations in the adjoint subroutine which gives the correct results.

A simple example: testing the adjoint subroutine

Considering that each subroutine performs a linear operation we can use the definition of the adjoint

$$< \boldsymbol{M}u , v > = < u , \boldsymbol{M}^{+} v >$$

to test our adjoint subroutine. In fact giving two random values to var1 and avar2 and calling the direct and adjoint subroutine the following identity must be satisfied to *machine precision*

$$< var2 , avar2 > = < var1 , avar1 >$$