

# Hands-on derivation of tangent linear and adjoint codes

**Angela Benedetti**

with contributions from:

**Marta Janisková and Yannick Tremolet**

## Road map

- Simple exercise of adjoint derivation
- “Manual” derivation of tangent linear for the Lorenz three-variable model (note that the first two equations were derived this morning during the lecture and can be found in the handouts).
- Derivation of the adjoint code (if time is an issue we will only start this task)
- Derivation of tangent linear and adjoint codes using an automatic differentiation software online (TAPENADE).  
<http://www-sop.inria.fr/tropics/> for general info  
<http://tapenade.inria.fr:8080/tapenade/index.jsp> for the actual deal
- Questions

Details on the Lorenz code and its use in variational data assimilation can be found in Huang and Yang (1996)

Find the adjoint statement of the following non-linear statement:

$$x = x^2 + y^2$$



## Naming conventions

- Remember that in the tangent linear routines of the ECMWF Integrated Forecasting System, the variables WITHOUT subscripts are the tangent linear variables, i.e. the derivatives, and in the adjoint routines they are the corresponding adjoint variables (gradients). The trajectory is usually indicated by attaching to the variables the suffix “5”.
- In this exercise, we WILL NOT adopt this convention in order to be able to compare more directly with codes derived using the automatic differentiation code TAPENADE. All tangent linear variables will have a suffix “d”, as well as all adjoint variables. The variables without any suffix are the nonlinear model variables.
- However, just to further confuse matters, you will note that in TAPENADE, all adjoint variables have a suffix “b”, while all tangent linear variables have the suffix “d”.
- By replacing the suffix “d” in your manually-derived adjoint code with “b”, you should be able to check your adjoint code directly with the result of the automatic differentiation without **trouble** (however, for that, some degree of luck will have to be involved ;) ).

## Lorenz model code (FORTRAN)

```
SUBROUTINE model(x,dt,nstep)
REAL,INTENT(INOUT) :: x(3)
REAL,INTENT(IN)    :: dt    ! constant
INTEGER, INTENT(IN):: nstep
  DO i = 1,nstep
    CALL lorenz(x,dxdt)
    CALL step  (x,dxdt,dt)
  ENDDO
END SUBROUTINE model

! -----
SUBROUTINE lorenz(x,dxdt)
REAL,INTENT(IN)  :: x(3)
REAL,INTENT(OUT):: dxdt(3)
REAL :: p, r      ! constants
  dxdt(1) = -p*x(1)+p*x(2)
  dxdt(2) = x(1)*(r-x(3))-x(2)
  dxdt(3) = x(1)*x(2)-b*x(3)
END SUBROUTINE lorenz

! -----
SUBROUTINE step(x,dxdt,dt)
REAL,INTENT(INOUT):: x(3)
REAL,INTENT(IN)   :: dxdt(3),dt
  DO i = 1,3
    x(i) = x(i)+dt*dxdt(i)
  ENDDO
END SUBROUTINE step
```

# Steps to use the automatic differentiation software

1. Work in groups of 2-3 people
2. Make sure you have access to the internet from your desktops
3. Open a terminal window
4. Type ftp <ftp.ecmwf.int>
5. Login: benedetti
6. Password: ang31a
7. cd Lorenz
8. mget \* (you will be acquiring three files: model.f95 lorenz.f95 and step.f95)
9. Open an internet browser and go to:  
<http://tapenade.inria.fr:8080/tapenade/index.jsp>
10. Upload files model.f95, lorenz.f95, step.f95 as “source”
11. Enter “model” as top routine (leave the rest blank)
12. Differentiate in: “Tangent mode” for tangent linear and “Reverse” for adjoint code
13. You can look at the resulting codes directly on the screen by clicking on the specific routine or download them onto your desktop
14. There will be slight differences with respect to your manually-derived codes, ask if you are confused.
15. Have fun!