1. Structures of the GRAPES-CUACE aerosol module

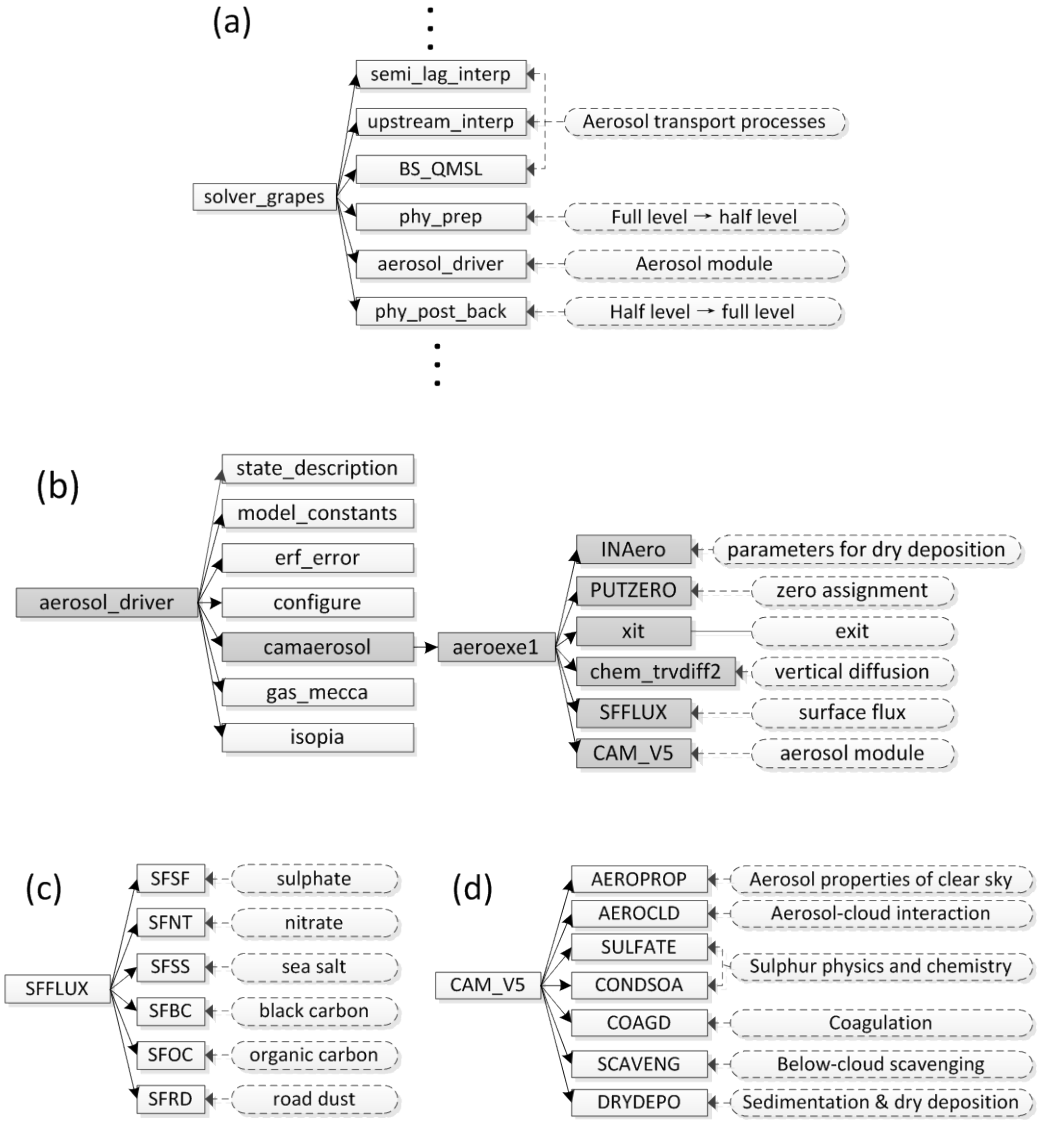

Figure S1 Structure of GRAPES-CUACE aerosol model. Dashed boxes are descriptions of each subroutine. (a) Aerosol transport processes and the aerosol module main interface; (b) Shaded are interface subroutines (aerosol_driver, module_ae_cam and aeroexe1) and CAM; (c)Surface fluxes calculation module; (d) The aerosol physical and chemical processes section.

2. Structures of theCUACE aerosol adjoint model

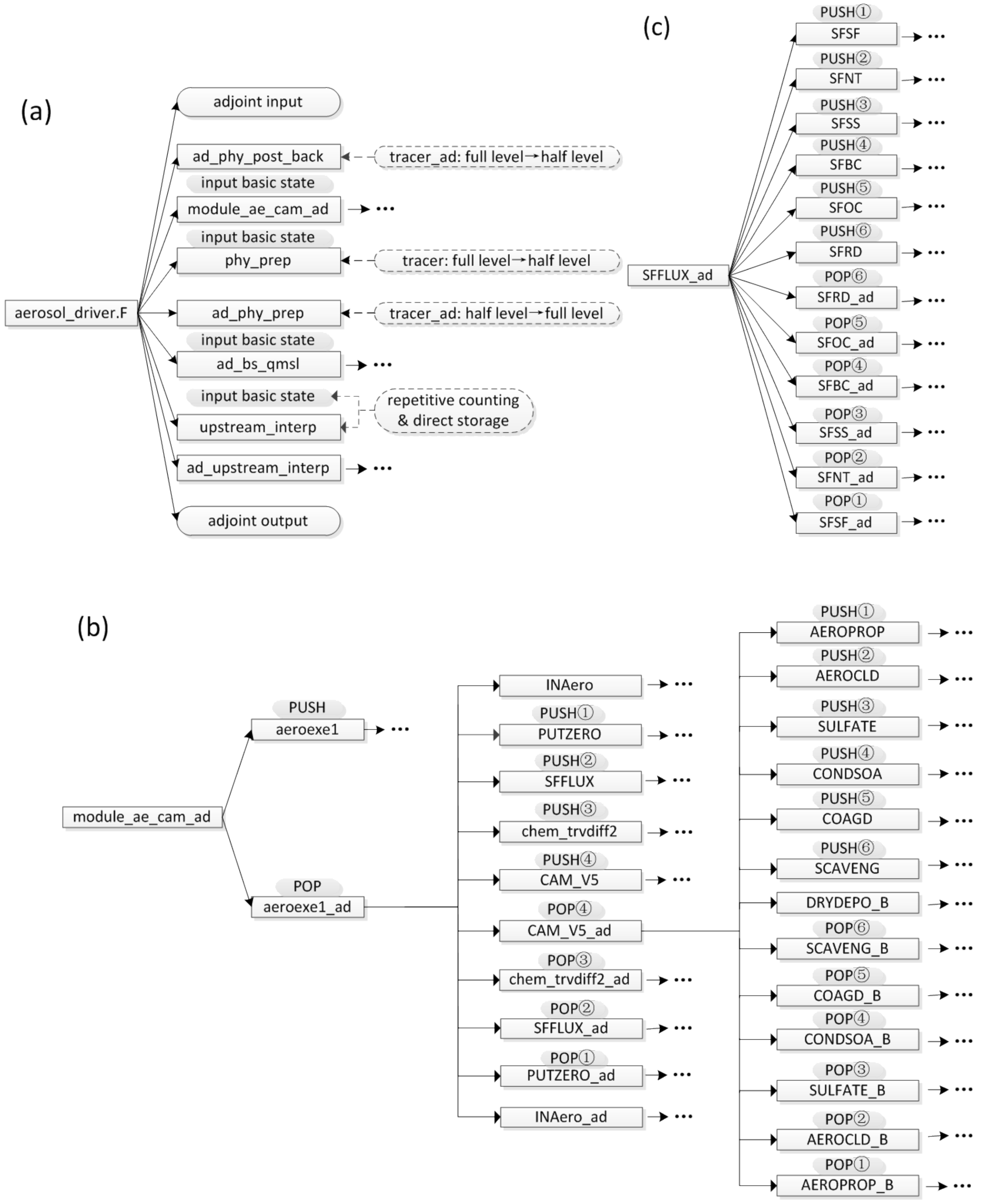

Figure S2 Structure of the adjoint of GRAPES-CUACE aerosol model. (a): The adjoint structure of Fig. 1 (a); (b)The adjoint structure of Fig. 1 (b) and (d); (c): The adjoint structure of Fig. 1 (c).

In the GRAPES-CUACE aerosol model, ‘aerosol_driver’ is the main interface subroutine that connect GRAPES-Meso and CUACE. Both ‘aerosol_driver’ and the aerosol transport process subroutines are called by ‘solver_grapes’ (Figure 1(a)).

However, in developing the adjoint of CUACE aerosol model, ‘aerosol_driver’ acts as the main program. Aerosol transport processes, full and half level interpolation subroutines (‘ad_phy_prep’ and ‘ad_phy_post’) and the aerosol physics and chemical processes are all subroutines of ‘aerosol_driver’ (Figure 2 (a)).

The uneuilibrated variables are saved in checkpoint files at the beginning of each external time step during the forward integration. While for saving intermediate uneuilibrated variables, recalculation and stack storage (PUSH & POP) schemes are adopted. This type of two-level checkpointing strategy is similar to that used in the adjoint of GEOS-Chem and has been shown to optimally balance storage, memory and CPU requirements (Sandu et al., 2005; Henze et al., 2007).

3. Forward and adjoint codes example

Followings are two pieces of code from coagd.F (section 3.1; coagulation process) and coagd_ad.F (section 3.2; the adjoint of coagulation processes). In section 3.2, recalculation and stack storage schemes for intermediate uneuilibrated variables are shown.

3.1. Forward code from coagd.F:

```
DO I=1,ICOB
        DO J=I,ICOB
          DO L=1+MAE,ILEV
            DO IL=IL1,IL2
C
C      * DIFFUSION COEFFICIENTS
C
DIFFX=PDIFF(IL,L,I)
DIFFY=PDIFF(IL,L,J)
C
C      * AIR'S DYNAMIC VISCOSITY
C
              AMU=145.8*1.E-8*THROW(IL,L+1)**1.5/
     1                              (THROW(IL,L+1)+110.4)
DSUM=2.*(RHSIZE(IL,L,I)+RHSIZE(IL,L,J))
C
C        BROWNIAN COAGULATION COEFFICIENT        [V1, V2, VR - PARTICLE MASS, KG]
C
CBAR =SQRT(CBAR12(IL,L,I)+CBAR12(IL,L,J))
GMEAN=SQRT(GX(IL,L,I)+GX(IL,L,J))

BETA(IL,L,I,J)=6.2832*(DIFFX+DIFFY)*DSUM/(DSUM/(DSUM
     1                         +2.*GMEAN)+8.*(DIFFX+DIFFY)/
     2                                    (CBAR*DSUM*STICK))
```

```
C
C        ADD GRAVITATIONAL COAGULATION
C
              BETA(IL,L,I,J)=BETA(IL,L,I,J)+0.7854*DSUM**2
     1                                *ABS(PDEPV(IL,L,I)-PDEPV(IL,L,J))
BETA(IL,L,J,I)=BETA(IL,L,I,J)
            END DO
          END DO
        END DO
END DO
```

3.2. Corresponding adjoint code from coagd_ad.F:

```
C
C      * DIFFUSION COEFFICIENTS
C
diffx = pdiff(il, l, i)
diffy = pdiff(il, l, j)
C
C      * AIR'S DYNAMIC VISCOSITY
C
dsum = 2.*(rhsize(il, l, i)+rhsize(il, l, j))
                CALL PUSHREAL8(cbar)
C
C        BROWNIAN COAGULATION COEFFICIENT        [V1, V2, VR - PARTICLE MASS, KG]
C
cbar = SQRT(cbar12(il, l, i) + cbar12(il, l, j))
                CALL PUSHREAL8(gmean)
gmean = SQRT(gx(il, l, i) + gx(il, l, j))
C
beta(il, l, i, j) = 6.2832*(diffx+diffy)*dsum/(dsum/(
     +            dsum+2.*gmean)+8.*(diffx+diffy)/(cbar*dsum*stick))
                IF (pdepv(il, l, i) - pdepv(il, l, j) .GE. 0.) THEN
                  CALL PUSHREAL8(abs0)
abs0 = pdepv(il, l, i) - pdepv(il, l, j)
                  CALL PUSHCONTROL1B(0)
                ELSE
                  CALL PUSHREAL8(abs0)
abs0 = -(pdepv(il, l, i)-pdepv(il, l, j))
                  CALL PUSHCONTROL1B(1)
                END IF
C
C        ADD GRAVITATIONAL COAGULATION
```

```
C
beta(il, l, i, j) = beta(il, l, i, j) + 0.7854*dsum**2*
        +                    abs0
tmp = beta(il, l, i, j)
beta(il, l, j, i) = tmp
ENDDO
ENDDO
ENDDO
               CALL PUSHINTEGER4(ad_from)
              ENDDO

DO i=icob,1,-1
               CALL POPINTEGER4(ad_from)
               DO j=icob,ad_from,-1
                  DO l=ilev,1+mae,-1
                     DO il=il2,il1,-1
tmpb = betab(il, l, j, i)
betab(il, l, j, i) = 0.0
betab(il, l, i, j) = betab(il, l, i, j) + tmpb
dsum = 2.*(rhsize(il, l, i)+rhsize(il, l, j))
dsumb = abs0*0.7854*2*dsum*betab(il, l, i, j)
abs0b = 0.7854*dsum**2*betab(il, l, i, j)
                        CALL POPCONTROL1B(branch)
                        IF (branch .EQ. 0) THEN
                           CALL POPREAL8(abs0)
pdepvb(il, l, i) = pdepvb(il, l, i) + abs0b
pdepvb(il, l, j) = pdepvb(il, l, j) - abs0b
                        ELSE
                           CALL POPREAL8(abs0)
pdepvb(il, l, j) = pdepvb(il, l, j) + abs0b
pdepvb(il, l, i) = pdepvb(il, l, i) - abs0b
                        END IF
diffx = pdiff(il, l, i)
diffy = pdiff(il, l, j)
temp8 = stick*cbar*dsum
temp7 = (diffx+diffy)/temp8
temp6 = dsum + 2.*gmean
temp5 = dsum/temp6 + 8.*temp7
temp5b = 6.2832*betab(il, l, i, j)/temp5
temp5b0 = -((diffx+diffy)*dsum*temp5b/temp5)
temp6b = -(dsum*temp5b0/temp6**2)
temp7b = 8.*temp5b0/temp8
temp8b = -(temp7*temp7b)
diffxb = temp7b + dsum*temp5b
```

```
diffyb = temp7b + dsum*temp5b
dsumb = dsumb + stick*cbar*temp8b + temp6b + temp5b0/
     +               temp6 + (diffx+diffy)*temp5b
gmeanb = 2.*temp6b
cbarb = dsum*stick*temp8b
betab(il, l, i, j) = 0.0
              CALL POPREAL8(gmean)
              IF (gx(il, l, i) + gx(il, l, j) .EQ. 0.0) THEN
temp5b1 = 0.0
              ELSE
temp5b1 = gmeanb/(2.0*SQRT(gx(il, l, i)+gx(il, l, j)))
              END IF
gxb(il, l, i) = gxb(il, l, i) + temp5b1
gxb(il, l, j) = gxb(il, l, j) + temp5b1
              CALL POPREAL8(cbar)
              IF (cbar12(il, l, i) + cbar12(il, l, j) .EQ. 0.0) THEN
temp5b2 = 0.0
              ELSE
temp5b2 = cbarb/(2.0*SQRT(cbar12(il, l, i)+cbar12(il,
     +                l, j)))
              END IF
cbar12b(il, l, i) = cbar12b(il, l, i) + temp5b2
cbar12b(il, l, j) = cbar12b(il, l, j) + temp5b2
rhsizeb(il, l, i) = rhsizeb(il, l, i) + 2.*dsumb
rhsizeb(il, l, j) = rhsizeb(il, l, j) + 2.*dsumb
pdiffb(il, l, j) = pdiffb(il, l, j) + diffyb
pdiffb(il, l, i) = pdiffb(il, l, i) + diffxb
ENDDO
ENDDO
ENDDO
ENDDO
```

References:

Sandu, A., Daescu, D., Carmichael, G. R., and Chai, T,:Adjoint sensitivity analysis of regional air quality models, J., Comput. Phys., 204, 222-252, 2005a.

Henze, D., Hakami, A., and Seinfeld, J.: Development of the adjoint of GEOS-Chem, Atmos. Chem. Phys., 7, 2413-2433, 2007.