Heterogeneous chemistry in WRF-Chem

Quick version: In order to implement heterogeneous chemistry reactions in WRF-Chem using chem_opt = 202, you need to modify the following files and then manually recompile the model:

../WRFChem/WRFChem4.2/chem/KPP/mechanisms/mozart_mosaic_4bin_aq:
Mozart_mosaic_4bin_aq.eqn
Mozart_mosaic_4bin_aq.def

The inc files are found in: …/WRFChem/WRFChem4.2/chem/KPP/inc/mozart_mosaic_4bin_aq
Specifically, the following files need to be updated:
kpp_mechd_ibu_mozart_mosaic_4bin_aq.inc
extra_args_to_update_rconst_mozart_mosaic_4bin_aq.inc
extra_args_update_rconst_mozart_mosaic_4bin_aq.inc
kpp_mechd_l_mozart_mosaic_4bin_aq.inc
extra_decls_update_rconst_mozart_mosaic_4bin_aq.inc

Lastly, the KPP uses the gen.c file found in
…/WRFChem/WRFChem4.2/chem/KPP/kpp/kpp-2.1/src

After changing updating these files it is necessary to compile the model manually as described in the WRFotron Users Guide: https://wrfchem-leeds.github.io/WRFotron/compilation.html

---

The heterogeneous chemistry reactions are implemented in MOZART MOSAIC 4BIN AQ (chem_opt = 202) in a similar way to T1 MOZCART (chem_opt = 132).

First, will add desired heterogeneous reactions into the Mozart_mosaic_4bin_aq.eqn file which is found in the following directory: ../WRFChem/WRFChem4.2/chem/KPP/mechanisms/mozart_mosaic_4bin_aq

An additional “heterogeneous reaction” section was added to the end of the file.

```
// Heterogeneous Reactions

{331:7949} HO2          = 0.5 H2O2                         : usr_HO2_aer( aero_srf_area, aero_diam,
temp )                             ;

{332:7939} N2O5          = 2 HNO3                          : usr_N2O5_aer( aero_srf_area, aero_diam,
temp )                             ;

 {333:7941} NO2          = 0.5 HONO + 0.5 HNO3                      : usr_NO2_aer( aero_srf_area,
aero_diam, temp )                              ;
```

In order for WRF-Chem to calculate the reaction rates for these reactions we need to define these reactions (usr_HO2_aer, usr_N2O5_aer, usr_NO2_aer) in the Mozart_mosaic_4bin_aq.def file. We also need to calculate aero_srf_area and aero_diam variables in this file.

The types are first defined:

```
REAL(KIND=dp) FUNCTION usr_N2O5_aer( aero_srf_area, aero_diam, temp )

! heterogeneous uptake on aerosols: N2O5 -> 2 HNO3

   REAL(KIND=dp), INTENT(IN) :: aero_srf_area(:)        ! aerosol surface area
   REAL(KIND=dp), INTENT(IN) :: aero_diam(:)            ! aerosol diameter
   REAL(KIND=dp), INTENT(IN) :: temp                    ! temperature (K)

   INTEGER :: n
   REAL(KIND=dp), parameter :: dg = .1_dp
   REAL(KIND=dp), parameter :: gamma_n2o5 = .1_dp
   REAL(KIND=dp) :: c_n2o5, term

   n = size( aero_srf_area )

   c_n2o5 = 1.40e3_dp * sqrt( temp )
   term = 4._dp/(c_n2o5*gamma_n2o5)

   usr_N2O5_aer = &
    sum( aero_srf_area(1:n)/(.5_dp*aero_diam(1:n)/dg + term) )

END FUNCTION usr_N2O5_aer


REAL(KIND=dp) FUNCTION usr_NO2_aer( aero_srf_area, aero_diam, temp )
! heterogeneous uptake on aerosols: NO2 -> 0.5 HONO + 0.5 HNO3

   REAL(KIND=dp), INTENT(IN) :: aero_srf_area(:)        ! aerosol surface area
   REAL(KIND=dp), INTENT(IN) :: aero_diam(:)            ! aerosol diameter
   REAL(KIND=dp), INTENT(IN) :: temp                    ! temperature (K)

   INTEGER :: n
   REAL(KIND=dp), parameter :: dg = .1_dp
   REAL(KIND=dp), parameter :: gamma_no2 = 1.e-5_dp
   REAL(KIND=dp) :: c_no2, term

   n = size( aero_srf_area )
   c_no2 = 2.15e3_dp * sqrt( temp )
   term = 4._dp/(c_no2*gamma_no2)

   usr_NO2_aer = &
    sum( aero_srf_area(1:n)/(.5_dp*aero_diam(1:n)/dg + term) )

END FUNCTION usr_NO2_aer

REAL(KIND=dp) FUNCTION usr_HO2_aer( aero_srf_area, aero_diam, temp )
! heterogeneous uptake on aerosols: HO2 -> 0.5 H2O2

   REAL(KIND=dp), INTENT(IN) :: aero_srf_area(:)        ! aerosol surface area
   REAL(KIND=dp), INTENT(IN) :: aero_diam(:)            ! aerosol diameter
   REAL(KIND=dp), INTENT(IN) :: temp                    ! temperature (K)

   INTEGER :: n
   REAL(KIND=dp), parameter :: dg = .1_dp
   REAL(KIND=dp), parameter :: gamma_ho2 = .2_dp
   REAL(KIND=dp) :: c_ho2, term

   n = size( aero_srf_area )

   c_ho2 = 2.53e3_dp * sqrt( temp )
   term = 4._dp/(c_ho2*gamma_ho2)

   usr_HO2_aer = &
    sum( aero_srf_area(1:n)/(.5_dp*aero_diam(1:n)/dg + term) )

END FUNCTION usr_HO2_aer
```

At the end of the .def file there is a subroutine that calculates aerosol surface diameter and area, beginning with:

```
   SUBROUTINE aero_surfarea( aero_srf_area, aero_diam, rh, temp, &
                      so4_a01, oc_a01, bc_a01, &
                      so4_a02, oc_a02, bc_a02, &
                      so4_a03, oc_a03, bc_a03, &
                      so4_a04, oc_a04, bc_a04 )
```

a01, a02, a03, and a04 are MOSAIC variables calculated by the model for the 4 size bins. The MOSAIC variables are speciated as: sulfate (so4), organic carbon (oc), and black carbon (bc) and others. More

MOSIAC variables can be added to this list, but these make up the bulk of PM and is kept to this list for simplicity. These 12 variables are calculated by the subroutine and placed in an array called “aero_srf_area” . There is a similar array for aerosol diameter called “aero_diam” which accounts for hygroscopic growth and aerosol growth.

The mean radius for each bin was updated to the mean radius/diameter for that bin number which is why it is the same for each bin regardless of chemical composition. Additional updates to the code were included to account for these additional variables needed to include all bins.

Because we have these additional local variables within the aerosol surface area subroutine (i.e. so4_a02), confusingly called the same thing as the variables in MOSAIC, we need to define these variables in the inc files.

The inc files are found in: …/WRFChem/WRFChem4.2/chem/KPP/inc/mozart_mosaic_4bin_aq
Specifically, the following files need to be updated:
kpp_mechd_ibu_mozart_mosaic_4bin_aq.inc
extra_args_to_update_rconst_mozart_mosaic_4bin_aq.inc
extra_args_update_rconst_mozart_mosaic_4bin_aq.inc
kpp_mechd_l_mozart_mosaic_4bin_aq.inc
extra_decls_update_rconst_mozart_mosaic_4bin_aq.inc

Unit conversions are required to convert from a mass concentration (μg / kg dry air) to a surface area (cm^2/cm^3 air) in kpp_mechd_ibu_mozart_mosaic_4bin_aq.inc

Lastly, the KPP uses the gen.c file found in
…/WRFChem/WRFChem4.2/chem/KPP/kpp/kpp-2.1/src
To produce the .f90 files in
/nobackup/chmltf/WRFChem/WRFChem4.2/chem/KPP/mechanisms/mozart_mosaic_4bin_aq

Specifically, the file mozart_mosaic_4bin_aq_Update_Rconst.f90 needs to be modified to run the aerosol surface area subroutine. Therefore we added the following to gen.c where similar code is used for T1 MOZCART.

```
 if( !strcmp( rootFileName,"mozart_mosaic_4bin_aq" ) ) {
   NewLines(1);
   bprintf( "   real(dp) :: aero_srf_area(12)\n");
   bprintf( "   real(dp) :: aero_diam(12)\n");
   NewLines(1);
   bprintf( "   call aero_surfarea( aero_srf_area, aero_diam, rh, temp, &\n");
   bprintf( "                    so4_a01, oc_a01, bc_a01, & \n");
   bprintf( "                    so4_a02, oc_a02, bc_a02, & \n");
   bprintf( "                    so4_a03, oc_a03, bc_a03, & \n");
```

```
bprintf( "                    so4_a04, oc_a04, bc_a04 )\n");

bprintf( "     sulf_srf_area_a01 = aero_srf_area(1) \n");
bprintf( "     oc_srf_area_a01   = aero_srf_area(2) \n");
bprintf( "     bc_srf_area_a01   = aero_srf_area(3) \n");
bprintf( "     sulf_srf_area_a02 = aero_srf_area(4) \n");
bprintf( "     oc_srf_area_a02   = aero_srf_area(5) \n");
bprintf( "     bc_srf_area_a02   = aero_srf_area(6) \n");
bprintf( "     sulf_srf_area_a03 = aero_srf_area(7) \n");
bprintf( "     oc_srf_area_a03   = aero_srf_area(8) \n");
bprintf( "     bc_srf_area_a03   = aero_srf_area(9) \n");
bprintf( "     sulf_srf_area_a04 = aero_srf_area(10) \n");
bprintf( "     oc_srf_area_a04   = aero_srf_area(11) \n");
bprintf( "     bc_srf_area_a04   = aero_srf_area(12) \n");
```