Documentation for MF2005_GWM
August 17, 2007

The MODFLOW Ground Water Management (GWM) Process is available within MODFLOW-2000 as the code MF2K_GWM and is fully documented in Ahlfeld et al. (2005). This note describes MF2005_GWM, a code which couples the GWM Process with MODFLOW-2005 (Harbaugh, 2005). Input to MF2005_GWM is nearly identical to that for MF2K_GWM. The one required change to input is discussed below along with changes in sample problems. The remainder of this document describes programming issues related to the conversion from MF2K to MF2005 and the integration of GWM into MODFLOW.

Changes to Input to GWM

The capabilities and input requirements for GWM are fully described in Ahlfeld et al. (2005) with modifications posted to the USGS Web page. These remain unchanged with one exception: the means of handling the GLOBAL file. In MF2K_GWM, a global file was required. It was specified in the MODFLOW Name file and was used to record GWM output. MODFLOW-2005 no longer includes a GLOBAL file, instead directing all output to the LIST file. To retain a separate file for output specific to the GWM process, MF2005_GWM creates a separate output file. The user can name this file or allow GWM to assign the file a default name. The naming of this file occurs in the GWM file. Hence, the GWM input instructions in Ahlfeld et al. (2005), on page 34, are revised for the GWM file as follows:

Input Items:

0. #Text

1. A file name for all output from the GWM Process may be assigned here. If no filename is specified here, a default name of GWM.OUT, written to the directory in which program execution occurs, is used. If a filename is specified, it must be the first keyword read.
   OUT Fname

2. ... **DECVAR** ...

3. ... **OBJFNC** ...

4. ... **VARCON** ...

5. ... **SUMCON** ...
   ... **HEDCON** ...
   ... **STRMCON** ...

6. ... **SOLN** ...

Below is an example of the use of the OUT keyword in the DEWATER sample problem. The OUT keyword is the first keyword in the file. Its presence will cause GWM to write GWM Process output to the file DEWATER.GWMOUT. If the line with the OUT keyword were absent from this file then the output would be written to a file named GWM.OUT.

```
#DEWATER Sample Problem, GWM file
#August 12, 2007
OUT     dewater.gwmout
DECVAR  ..\data\dewater.decvar
OBJFNC  ..\data\dewater.objfnc
VARCON  ..\data\dewater.varcon
HEDCON  ..\data\dewater.hedcon
SOLN    ..\data\dewater.soln
```

Changes to Sample Problems

The sample problems originally distributed with MF2K_GWM have been modified to operate with MF2005_GWM. All problems have been modified to remove GLOBAL from the MODFLOW Name file and place the keyword OUT, with corresponding name of the output file, in the GWM file. The SEAWATER problem distributed with MF2K_GWM included an SEN file which set the value of recharge, over-riding the value specified in the RCH file, to demonstrate the ability of GWM to read parameters. The SEN Process is not available in MF2005, however, parameters can be read using the PVAL Package. As a result, the SEAWATER problem includes a PVAL file replacing the SEN file. The output from the sample problems distributed with MF2005_GWM can still be compared with those printed in Ahlfeld et al. (2005) except that the information output to the GLOBAL file from the GWF process is no longer present in the output file of the MF2005 version.

Programming Notes

The GWM Process consists of a BAS package, the RMS package for solving the optimization problem and six other packages for defining decision variables, objective function and constraints. To couple the GWM and GWF Processes, changes to the main program of MODFLOW are required.

A major change in the programing of MODFLOW from the MF2000 version to the MF2005 version is the use of the FORTRAN-90 module structure for storing and sharing data. The components of the GWM process in MF2K_GWM were originally coded using this data structure so that the incorporation of GWM into MF2005 is fairly straightforward. Nevertheless, minor changes have been made to most of the GWM packages to take advantage of the full use of module structures by MF2005. These changes make the new GWM packages incompatible with MF2K. As a result a new set of GWM packages have been defined, each with the number designation 2, so that, for example, the package GWM1OBJ1 becomes GWM1OBJ2. To avoid confusion, this change was made to all GWM packages, even if no internal changes were made to the code. In each case, the version date listed at the beginning of each package has been changed; however, the version date in each subroutine has been unchanged from that found in version 1, unless the code within the subroutine has been changed. Major

changes have been made to the GWM BAS Package and minor changes to all other packages.

Changes to GWM BAS package

To accommodate the new features of MF2005 a new version of the BAS package for GWM was created. This version is named GWM1BAS2 and includes modified versions of the AR and RW routines, eliminates the RPP subroutine and introduces three subroutines and one function. The changes to the AR and RW routines and the functions of the new subroutines are described here.

GWM1BAS2AR allocates and reads all information needed by GWM. Changes from GWM1BAS1AR include:

1) accomodations for the elimination of the GLOBAL file from MF2005,
2) addition of code to look for OUT as the first keyword in the GWM file. If present, a name is read and assigned to the GWM output file. If not present this output goes to a file with the default name GWM.OUT,
3) addition of calls to IGETUNIT to obtain a unit number for the GWM.OUT file. IGETUNIT was available in MF2000, but is not present in MF2005. A copy of this subroutine is placed in GWM1BAS2SUBS.for.

GWM1BAS2RW performs rewind operations on all input files. It also rewinds the LIST file and writes a new header to this file. It is constructed from the rewind subroutine found in the MF2000 Parameter Estimation Process, PES1BAS6RW.

GWM1BAS2DABAS7 performs selected deallocation of memory that had been allocated in the BAS7 package. Memory that is originally allocated in SGWF2BAS7ARDIS, SGWF2BAS7ARMZ or SGWF2BAS7I is deallocated in GWM1BAS2DABAS7.

GWM1BAS2RRF is a modifed version of the GWF1BAS7AR routine with several important differences.

1) most of the allocation in GWF1BAS7AR is not repeated as these arrays are not deallocated.
2) A new subroutine, GWM1BAS2OPEN, is called to check that all files remain open.
3) If the GWF process simulation is steady state then the solution from the prior simulation run can be used as an initial estimate of the solution, significantly speeding solution time. For this case, unless it is the last simulation, STRT in the Basic Package File (BAS6) is not read and the HNEW array is not reset.

GWM1BAS2OPEN is derived from SGWF2BAS7OPEN and makes sure that all files listed in the MF2005 Name file are still open. If any file is not open, it is re-opened in this subroutine. This check is needed because some packages (e.g. PVAL) close their input files after reading is completed.

Minor Changes to GWM Packages

GWM1DCV2 – subroutines GWF2DCV2FM and GWF2DCV2BD changed to eliminate most items in argument list with appropriate USE statements added. Since both of these subroutines are considered part of the GWF Process, the process number has also been changed to be consistent with the new process number for MF2005.

GWM1DCC2 – no change to code.

GWM1HDC2 – subroutine GWM1HDC2OS changed to eliminate most items in argument list with appropriate USE statements added.

GWM1OBJ2 – no change to code.

GWM1RMS2 – subroutine GWM1RMS2OS changed to eliminate most items in argument list with appropriate USE statements added.

GWM1SMC2 – no change to code.

GWM1STC2 – subroutine GWM1STC2OS changed to eliminate most items in argument list with appropriate USE statements added.

Modifications to the MF2005 Main Program to Create MF2005-GWM

The major addition to the MF2005 main program required by GWM is the introduction of a looping structure around the commands for executing a GWF Process simulation. At the completion of each GWF Process run, before the next run is initiated, MF2005_GWM re-reads most input. In preparation for re-reading input, MF2005_GWM rewinds all input files using the subroutine GWM1BAS2RW. The space utilized by each of the GWF packages is deallocated using the appropriate GWF DA routines. In addition, selected portions of GWF BAS data are deallocated using the subroutine GWM1BAS2DABAS7. The re-reading of the input and associated re-allocation of memory is accomplished by calling the allocate and read (AR) subroutines for each package. Only some of the GWF BAS data is re-read. This re-reading is accomplished in subroutine GWM1BAS2RRF. Included in this subroutine is a call to GWM1BAS2OPEN, which performs a check to insure that all files in the MF2005 Name file remain open. A detailed description of the steps required to create the MF2005_GWM main program is given at the end of this document.

References

Ahlfeld, D.P., Barlow, P.M., and Mulligan, A.E., 2005, GWM-A ground-water management process for the U.S. Geological Survey modular ground-water model (MODFLOW-2000): U.S. Geological Survey Open-File Report 2005-1072, 124 p.

Harbaugh, A.W., 2005, MODFLOW-2005, the U.S. Geological Survey modular ground-water model—the Ground-Water Flow Process: U.S. Geological Survey Techniques and Methods 6-A16, variously paginated.

APPENDIX: Creating MF2005_GWM Main Program

1: Insert USE statements for GWM subroutines with other USE statements

```
C GWM: GWM SUBROUTINES CALLED FROM MF2005 MAIN PROGRAM
      USE GWM1BAS2SUBS, ONLY: GWM1BAS2AR,GWM1BAS2RW,GWM1BAS2RRF,
     &                        GWM1BAS2DABAS7
      USE GWM1RMS1SUBS, ONLY: GWM1RMS1PL,GWM1RMS1PP,GWM1RMS1FP,
     &                        GWM1RMS1FM,GWM1RMS1AP,GWM1RMS1OT,
     &                        GWM1RMS1OS
      USE GWM1DCV1, ONLY: GWF2DCV1FM,GWF2DCV1BD
      USE GWM1HDC1, ONLY: GWM1HDC1OS
      USE GWM1STC1, ONLY: GWM1STC1OS
```

2: Replace Version definition

```
      PARAMETER (VERSION='MF2005_GWM 1.0 081207,FROM MF2005V1.3.01')
```

3: Insert local GWM variables among declaration statements

```
C GWM: LOCAL VARIABLES CREATED FOR GWM
      LOGICAL FIRSTSIM,LASTSIM,FINISH,MFCNVRG,GWMCNVRG
      INTEGER IPERT,NPERT
      REAL HCLOSET
      CHARACTER*200 FNAMEN
```

4: Insert code to save name of name file

```
C GWM: SAVE NAME OF NAME FILE FOR LATER USE
      FNAMEN=FNAME
```

5: Move calls to solver AR subroutines and OBS2…AR subroutines to immediately after the call to GWF2BAS7AR.

6: After OBS2…AR calls, insert code for GWM…AR call.

```
C GWM: READ GWM PROCESS INFORMATION
      IF(IUNIT(56).GT.0)THEN ! DEFINE PROPER HCLOSE
        IF(IUNIT(9).GT.0)  HCLOSET = HCLOSE    !SIP
        IF(IUNIT(10).GT.0) HCLOSET = HCLOSEDE4 !DE4
        IF(IUNIT(13).GT.0) HCLOSET = HCLOSEPCG !PCG
        IF(IUNIT(42).GT.0) HCLOSET = HCLOSEGMG !GMG
        CALL GWM1BAS2AR(IUNIT(56),IOUT,NROW,NCOL,NLAY,
     &                  NPER,PERLEN,HCLOSET,VERSION,MFVNAM)
      ENDIF
```

7: Immediately following, insert code for controlling GWM loops

```
C GWM: INSERT GWM LOOPING CONTROLS
      FIRSTSIM = .TRUE.
      GWMCNVRG = .FALSE.
      FINISH   = .FALSE.
C
C-----PREPARE ITERATION LOOP CONTROLS
      IF(IUNIT(56).EQ.0)THEN           ! GWM NOT ACTIVE; LOOP FOR ONE PASS
        LASTSIM = .TRUE.
      ELSEIF(IUNIT(56).GT.0)THEN       ! GWM ACTIVE; LOOP FOR GWM ITERATIONS
        LASTSIM = .FALSE.
        MFCNVRG = .TRUE.
      ENDIF
C
C-----BEGIN ITERATION LOOP FOR GWM PROCESS
```

```
      DO 300 WHILE (.NOT. FINISH)
C
C-------PREPARE PERTURBATION LOOP CONTROLS
        IF(IUNIT(56).EQ.0)THEN         ! GWM NOT ACTIVE; LOOP FOR ONE PASS
          IPERT = 1
          NPERT = 1
        ELSEIF(IUNIT(56).GT.0)THEN     ! GWM ACTIVE; SET LOOP CONTROLS
          CALL GWM1RMS1PL(IPERT,NPERT,FIRSTSIM,LASTSIM)
        ENDIF
C
C-------BEGIN PERTURBATION LOOP FOR GWM PROCESS
        DO 200 WHILE (IPERT .LE. NPERT)
C
C-------PREPARE CALCULATION OF SIMULATION RESPONSE
        IF(IUNIT(56).GT.0)
     &      CALL GWM1RMS1PP(IOUT,MFCNVRG,IPERT,FIRSTSIM,LASTSIM)
```

8: Immediately following, insert code to rewind files and re-read BAS data before returning to all remaining GWF…AR calls.

```
C GWM: PREPARE FLOW PROCESS FILES FOR NEXT SIMULATION
      IF(IUNIT(56).GT.0 .AND. .NOT.FIRSTSIM)THEN
C-------REWIND FLOW PROCESS MODEL FILES AND RE-READ BAS FILE
        CALL GWM1BAS2RW(INUNIT,FNAME,CUNIT,NIUNIT,IOUT,VERSION,MFVNAM)
        CALL GWM1BAS2RRF(INUNIT,CUNIT,VERSION,24,31,32,
     1                IGRID,12,HEADNG,26,MFVNAM,FNAMEN,LASTSIM)
      ENDIF
```

9: Writing to screen at each time step interferes with GWM screen output, so, comment out the lines just before loop 30, the iteration loop.

```
C GWM: too much information to screen    WRITE(*,25)KPER,KSTP
C   25     FORMAT(' Solving:  Stress period: ',i5,4x,
C     &       'Time step: ',i5,4x,'Ground-Water Flow Eqn.')
```

10: Add GWM managed flows to FM block within GWF iteration loop

```
C GWM: ADD GWM MANAGED FLOWS
            IF(IUNIT(56).GT.0) CALL GWF2DCV1FM(KKPER)
```

11: Insert just before 33 CONTINUE to act on failed GWF convergence

```
C GWM: IF GWM ACTIVE AND CONVERGENCE HAS FAILED SKIP TO PERTURBATION LOOP END
            IF(IUNIT(56).GT.0)THEN
              MFCNVRG = .FALSE.
              CALL GWM1RMS1FP(MFCNVRG,IPERT,NPERT,FIRSTSIM,LASTSIM)
              FIRSTSIM = .FALSE.
              GO TO 200
            ENDIF
```

12: Add GWM managed flows to budget calculations within BD block

```
C GWM: RECORD BUDGET FOR GWM MANAGED FLOW VARIABLES
          IF(IUNIT(56).GT.0)CALL GWF2DCV1BD(KKSTP,KKPER)
```

13: Insert between 90 CONTINUE and 100 CONTINUE

```
C GWM: ASSIGN SYSTEM STATE TO STATE ARRAY AT END OF STRESS PERIOD
          IF(IUNIT(56).GT.0)THEN
            CALL GWM1RMS1OS(KKPER) ! CHECK MANAGED WELLS FOR DEWATER
            CALL GWM1HDC1OS(KKPER)
            IF(IUNIT(18).GT.0)CALL GWM1STC1OS(KKPER)
          ENDIF
```

14: Insert immediately after 100 CONTINUE to complete simulation

```
        IF(IUNIT(56).EQ.0)THEN ! GWM NOT ACTIVE
          IPERT = 2            ! INCREMENT IPERT TO TERMINATE PERTURBATION LOOP
        ELSEIF(IUNIT(56).GT.0)THEN ! ANALYZE SIMULATION RESPONSE FOR GWM
          MFCNVRG = .TRUE.     ! SIMULATION HAS CONVERGED
          CALL GWM1RMS1FP(MFCNVRG,IPERT,NPERT,FIRSTSIM,LASTSIM)
          FIRSTSIM = .FALSE.   ! SIMULATION COMPLETED, NO LONGER FIRSTSIM
        ENDIF
```

15: To avoid too much elapsed time output change call to GLO1BAS6ET

```
      IF(LASTSIM) CALL GLO1BAS6ET(IOUT,IBDT,1)
```

16: Retain GWF deallocations for all packages except BAS and solvers and OBS process packages. Insert code to perform limited deallocation of BAS package.

```
      IF(.NOT. LASTSIM)  CALL GWM1BAS2DABAS7(IGRID)!DEALLOCATE SOME BAS7 MEMORY
```

17: Insert code to conclude looping, solve optimization problem and test for GWM convergence.

```
C
C GWM: END OF GWM PERTURBATION LOOP
  200   ENDDO
C
C-------IF NOT THE LAST SIMULATION, SOLVE GWM PROBLEM AT THIS ITERATION
        IF(.NOT. LASTSIM)THEN           ! GWM PROCESS MUST BE ACTIVE
          CALL GWM1RMS1FM               ! FORMULATE THE GWM PROBLEM
          CALL GWM1RMS1AP(GWMCNVRG)     ! SOLVE THE GWM PROBLEM
          IF(.NOT.GWMCNVRG)CALL GWM1RMS1OT(-1)! WRITE GWM SOLUTION OUTPUT
        ENDIF
C
C-------IF GWM PROBLEM HAS CONVERGED, ONE MORE SIMULATION RUN IS NEEDED
        IF(GWMCNVRG .AND. .NOT. LASTSIM)THEN
          CALL GWM1RMS1OT(0)            ! WRITE GWM SOLUTION OUTPUT
          LASTSIM = .TRUE.              ! EXECUTE ONE LAST SIMULATION
C
C-------IF GWM HAS COMPLETED ITS LAST SIMULATION OR GWM IS NOT ACTIVE
        ELSEIF(LASTSIM)THEN
          FINISH = .TRUE.               ! TERMINATE ITERATION LOOP
        ENDIF
C
C GWM: END OF GWM ITERATION LOOP
  300 ENDDO
```

18: Insert code to perform final deallocations before ending program.

```
C
C GWM: DEALLOCATE REMAINING PACKAGES: SOLVER, OBS2 AND BAS7
      IF(IUNIT(9).GT.0) CALL SIP7DA(IGRID)
      IF(IUNIT(10).GT.0) CALL DE47DA(IGRID)
      IF(IUNIT(13).GT.0) CALL PCG7DA(IGRID)
      IF(IUNIT(42).GT.0) CALL GMG7DA(IGRID)
      CALL OBS2BAS7DA(IUNIT(28),IGRID)
      IF(IUNIT(33).GT.0) CALL OBS2DRN7DA(IGRID)
      IF(IUNIT(34).GT.0) CALL OBS2RIV7DA(IGRID)
      IF(IUNIT(35).GT.0) CALL OBS2GHB7DA(IGRID)
      IF(IUNIT(38).GT.0) CALL OBS2CHD7DA(IGRID)
      CALL GWF2BAS7DA(IGRID)
```