Groundwater

Methods Note/

A Generic Closed-Loop Contaminant Treatment System Package for MODFLOW 6

by Jeremy T. White[1], John E. Ewing[2], Greg Ruskauff[3], and Helal Rashid[3]

Abstract

We present a contaminant treatment system (CTS) package for MODFLOW 6 that facilitates the simulation of pump-and-treat systems for groundwater remediation. Using the "nonintrusive" MODFLOW 6 application programming interface (API) capability, the CTS package can balance flows between extraction and injection wells within the outer flow solution loop and applies blended concentration/mass treatment efficiency within the outer transport solution loop. The former can be important when the requested extraction rate cannot be satisfied by the current simulated groundwater system conditions, while the latter can be important for simulating incomplete/imperfect treatment schemes. Furthermore, the CTS package allows users to temporally vary all aspects of a simulated CTS system, including the configuration and location of injection and extraction wells, and the CTS efficiency. This flexibility combined with the API-based implementation provide a generic and general CTS package that can be applied across the wide range of MODFLOW 6 simulation options and that evolves in step with MODFLOW 6 code modifications and advancements without needing to update the CTS package itself.

Introduction

Simulation of the performance and efficacy of closed-loop groundwater contaminant treatment systems (CTS) is a primary function of groundwater flow and mass-transport modeling. In this setting, practitioners design and construct a forward simulation of a dissolved-phased plume and then simulate the implementation of a CTS through the addition of new boundary conditions to represent the extraction and injection wells that endeavor to extract dissolved-phase from the groundwater system, apply treatment to remove (some) mass, and then inject the treated water back into the groundwater system. In these types of simulations, practitioners must take care to ensure the simulated CTS is conceptually in harmony with the actual system design that is to be implemented so that a representative estimate of system performance is provided (see Bedekar et al. (2016) for information and background on CTS simulations). Important considerations for CTS simulations include:

- conceptualization of the extractor and injector wells as "linked" sink-to-source boundary conditions in the flow and transport model such that the simulated volume of water extracted matches the simulated injected volume (Figure 1);
- balancing the CTS inflows and outflows in the case where the inflows to the CTS are effected by system state conditions that may vary during the simulation or as part of a higher-level analysis like uncertainty quantification, data assimilation, or management optimization;
- representing the treatment efficiency of the CTS to remove some, but possibly not all, dissolved-phase mass. That is, an imperfect treatment system. In some settings, mass removal is assumed to be 100% as there is no linkage between extracted and injected concentrations. However, treatment efficiency may not be 100% and some extracted mass may be recirculated via injection wells (Figure 1).

[1]Corresponding author: INTERA Inc., Boulder, CO; jwhite@intera.com

[2]INTERA Inc., Austin, TX, USA

[3]INTERA Inc., Richland, WA, USA

Article impact statement: We demonstrate how the MODFLOW 6 API can be used to implement a nonintrusive contaminant treatment system package.

Received March 2022, accepted July 2022.
© 2022 National Ground Water Association.
doi: 10.1111/gwat.13239

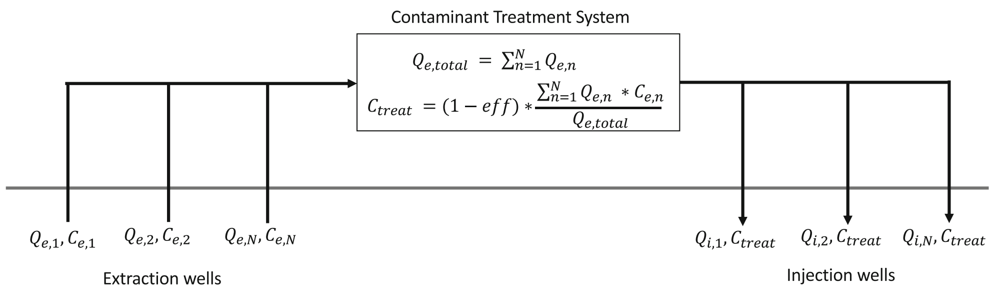

Figure 1. Conceptual diagram of a CTS. $Q_{e,n}$ and $C_{e,n}$ represent the extraction rate and concentration extracted groundwater for the extraction well n. $Q_{e,total}$ is the total extraction rate for the CTS, summed across all extraction wells. C_{treat} is the mixed, or "blended" concentration from the extraction wells times the CTS efficiency (*eff*), which represents the treatment process. $Q_{i,n}$ is the injection rate of the injection well n. Conceptually, if the CTS efficiency is less than 1.0, then C_{treat} is greater than 0.0. Based on figure 7 of Bedekar et al. (2016).

Additionally, to capture the usage of real world CTSs, each of the above considerations may change in time. That is, the locations and functions of extraction and injection wells may change, the flow rates of individual extraction wells ($Q_{e,1}$ to $Q_{e,n}$ quantities in Figure 1) and/or injection wells (the $Q_{i,1}$ to $Q_{i,n}$ quantities in Figure 1) may change, the aggregate CTS flow rate may change ($Q_{e,total}$ quantity in Figure 1), and the treatment efficiency (*eff* quantity in Figure 1) may change. Furthermore, if the model is unable to satisfy the requested extraction rate, then the injection rate must be adjusted to simulate a closed-loop CTS—the sum of the $Q_{e,n}$ values should match the sum of the $Q_{i,n}$ values in Figure 1. Otherwise, the volume of injected water may be larger than the volume of extracted water, resulting in an incorrect groundwater flow field, and, ultimately, a corrupted simulation of the overall CTS performance.

Recently, MODFLOW 6 Langevin et al. (2017), Hughes et al. (2017) and Langevin et al. (2022) has been put forward as an advanced, state-of-the-art groundwater flow and transport simulation engine. As described in Hughes et al. (2022), MODFLOW 6 can be driven through an application programming interface (API) to facilitate a high degree of practitioner control over the forward solution process. As described in Hughes et al. (2022), using MODFLOW 6 through the API means the practitioner takes control of the advancement of time in the simulation and the solution process itself. This happens by writing a driver program (herein, a python script) and calling a shared/dynamic-link library version MODFLOW 6, rather than the statically compiled MODFLOW 6 executable (both the MODFLOW 6 shared library and statically compiled executables are available at https://github.com/MODFLOW-USGS/executables for Windows, Mac, and Linux).

To meet the needs of simulating closed-loop CTSs within MODFLOW 6, we have implemented a CTS package in python that uses the MODFLOW 6 API functionality. The conceptual design of CTS package presented herein is based largely on the CTS simulation concepts presented in Bedekar et al. (2016). The MODFLOW 6 CTS package is designed to function for the case where the flow model and transport model are executed separately to support efficient multi-species simulation, and includes support for:

- multiple CTS instances (that is multiple groupings of injection and extraction wells),
- time-varying spatial configuration of each CTS instance,
- extractor-injector flow rate balancing (e.g., closed-loop CTSs) within the outer flow model solution loop,
- time-varying treatment system efficiency applied within the outer transport model solution loop by aggregating extraction well inflows and concentrations and then reintroducing remaining dissolved-phase mass at injection wells.

In this way, we provide a general CTS system implementation for MODFLOW 6, and one that is nonintrusive with respect to the MODFLOW 6 code base.

We note that a CTS package is available in MT3D-USGS Bedekar et al. (2016) that includes several of the above important CTS considerations, as well as additional functionality not included in the implementation presented herein. However, the CTS implementation in MT3D-USGS does not include flow-balancing and does not support unstructured grid models, both of which may be important in some settings. The ability to balance inflows and outflows for a given CTS system can be especially important for formal management optimization analyses where well locations and rates are adjusted programmatically in ways that may result in less-than-requested extraction rates.

The remainder of this work describes the implementation and general usage details of the MODFLOW 6 CTS package, and also demonstrates certain usage aspects on a simple synthetic problem.

Approach

The MODFLOW 6 API described in Hughes et al. (2022) exposes many aspects of the execution of a MODFLOW 6 simulation. Users can control the advancement of the simulation time and solver iterations, and can modify MODFLOW 6 inputs and outputs "in memory" between solver iterations and time advancement. In this way, the MODFLOW 6 API allows for the creation of additional packages for MODFLOW 6 without the need to write any MODFLOW 6 source code. Hughes et al. (2022) for more information related to the API functionality of MODFLOW 6.

Conceptually, a CTS instance receives groundwater inflow and dissolved-phase mass from the extraction wells. The simulated concentration and simulated inflow to the CTS instance for each extractor is aggregated to form a "blended" CTS concentration and total inflow into the CTS instance. If the total inflow from the extractors into a CTS instance is less than the requested CTS outflow flux to the injectors (as defined in the MODFLOW 6 WEL and/or MAW input files), then the flux assigned to the injectors is modified so that the total CTS instance outflow equals the total CTS instance inflow. This modification to the flow model injection well boundary conditions happens within the outer flow model solution loop so that the solution process continues until the total inflow and total outflow for the CTS instance are in equilibrium with the flow model solution. Note that the rate for each injector is modified in proportions equal to the requested injector rates. That is, the relative magnitude between injectors is maintained at the requested proportions as specified in the model input files.

If an optional treatment efficiency is supplied, it represents the fraction of dissolved-phase mass in the blended CTS inflow removed by treatment system. That is, if a treatment efficiency less than 1.0 (representing 100% treatment efficiency) is provided, the blended concentration is then reduced by multiplying the blended CTS input concentration by the efficiency factor. The resulting concentration is then specified as the transport-model injection well boundary condition concentration. The treatment efficiency process is applied within the transport model outer solution loop such that the transport model solution process continues until the simulated extracted and injected concentration is in equilibrium with the transport model solution.

The CTS package is implemented as a python class that interacts and drives the MODFLOW 6 flow and transport models. Conceptually, the class is designed to function with separate flow and transport simulations to support a more granular CTS modeling workflow. This approach uses the Flow Model Interface (FMI) package to link the flow solution to the transport solution.

The CTS package requires an input file; this file has been designed to mimic the MODFLOW 6 input style. The input file is divided into a series of "period data" blocks, where one block is defined for each user-defined CTS instance for each stress period, where a CTS instance is a grouping of extraction and injection wells that function as a single CTS instance. Each CTS instance can be a mixture of standard specified-flux (e.g., WEL) types and multi-aquifer well (MAW) types. Each period-data block can also include an optional efficiency value to simulate an incomplete removal of dissolved-phase mass by the simulated treatment system. The optional efficiency value is supplied for each CTS period-data block; treatment efficiency ranges from 0.0 to 1.0. Through the use of these period-data blocks, the definition and function of each CTS instance can be changed at the stress period level. An example CTS input file is included in the Appendix.

Example Function Calls

Here we present a few example python calls to demonstrate the MODFLOW 6 CTS usage pattern. First the `Mf6Cts` class object must be created (e.g., "instantiated" in the object-oriented parlance):

```
mf6 = Mf6Cts("model.cts","mf6lib.so",
  "flow_model","transport_model")
```

where "model.cts" is the CTS package input file stored in the `transport_model` directory, "flow_model" is the directory containing the MODFLOW 6 flow model input files and "transport_model" is the directory containing the MODFLOW 6 transport model input files, and "mf6lib.so" is the precompiled MODFLOW 6 shared library file that is present in both the flow and transport model directories. Once the `Mf6Cts` class is instantiated (the `mf6` variable), users need to solve the flow solution (that includes flow balancing for each CTS instance listed in "model.cts"):

```
mf6.solve_gwf()
```

After the flow model solution has completed, the `Mf6Cts` class will have written CSV-style CTS flow summary files in the "flow_model" directory that summarize the CTS node-level and system-level flow model performance. At this point, and depending on the configuration of the FMI package, the flow-model binary output files may need to be transferred from the `flow_model` directory to the `transport_model` directory for use by the transport model.

Now users can solve the transport model:

```
mf6.solve_gwt()
```

After this solution is complete, users will find CSV-style CTS summary files in the `transport_model` directory that summarize the CTS node-level and system-level transport model performance. If MAW-type boundary conditions are included in a CTS instance, then a MAW-specific CSV-style summary file is also written to the `transport_model` directory; this CSV file summarizes the performance of each node in the MAW boundary types.

The CTS package can also be used through a command line interface (CLI):

```
python mf6cts.py config.py
```

where `mf6cts.py` is the python source file containing the `Mf6Cts` class and `config.py` is a configuration file that is valid python source file containing the requisite

`Mf6Cts` arguments as python variables, as well as a listing of the flow model budget files needed for the transport model solution as a python `list` container. An example configuration file is provided in the Appendix.

Example Application

To demonstrate the use of the MODFLOW 6 CTS package, we developed a synthetic groundwater flow and transport simulation. The model is 330 m by 330 m horizontally and has a total vertical thickness of 10 m, represented by a single layer. The model domain is horizontally discretized into 33 rows and columns. The simulation is for 10 transient 180-d stress periods. A regional left-to-right gradient is simulated with general head boundary (GHB) instances along the first and last columns; the left GHB stage was set to 10 m (model top), while the right GHB stage was set to 5 m. Additional model details are as follows:

- specific storage = 0.00001/m
- specific yield = 0.01
- porosity = 0.05
- longitudinal dispersivity = 1.0 m
- transverse dispersivity = 0.1 m

Two different horizontal hydraulic conductivity (HK) fields were tested in an effort to demonstrate the importance of the flow-balancing functionality. One model run used a homogeneous HK value of 10 m/d. The other model run used a heterogeneous HK field realized via spectral simulation using an exponential variogram with a range equal to five times the cell dimension and a sill of 10 m/d (Figure 2D). All other model inputs were treated as spatially homogeneous and isotropic.

Dissolved-phase mass was introduced into the model domain by specifying a concentration of 100.0 mg/L for inflows from the left upgradient GHB instances for all stress periods.

Four injection wells and two extraction wells were placed in the model domain as shown in Figure 2. These wells were grouped into two CTS instances such that the northern extraction well and the northwest injection well-formed one CTS instance and the southern extraction well and three remaining injection wells formed a second CTS instance.

For this demonstration, the CTS systems were both given an efficiency of 0.7, meaning both CTS systems removed 70% of the dissolved-phase mass. Both CTS systems are active for stress periods 2 through 10.

For the homogeneous HK model run, each CTS instance was assigned a flow rate of 100 m^3/d for stress periods 2 through 10. This results in the first CTS instance injecting and extracting 100 m^3/d for it is single injection and extraction well. For the second CTS instance, the rate for each of the three injection wells is 33.3 m^3/d. For the heterogeneous HK model run, each injection well and extraction well in both CTS instances were varied for each stress period by scaling the homogeneous-case requested rates between values of zero and four times the homogeneous HK rate. This introduces spatial and temporal variability in the operation of the simulated CTS instances

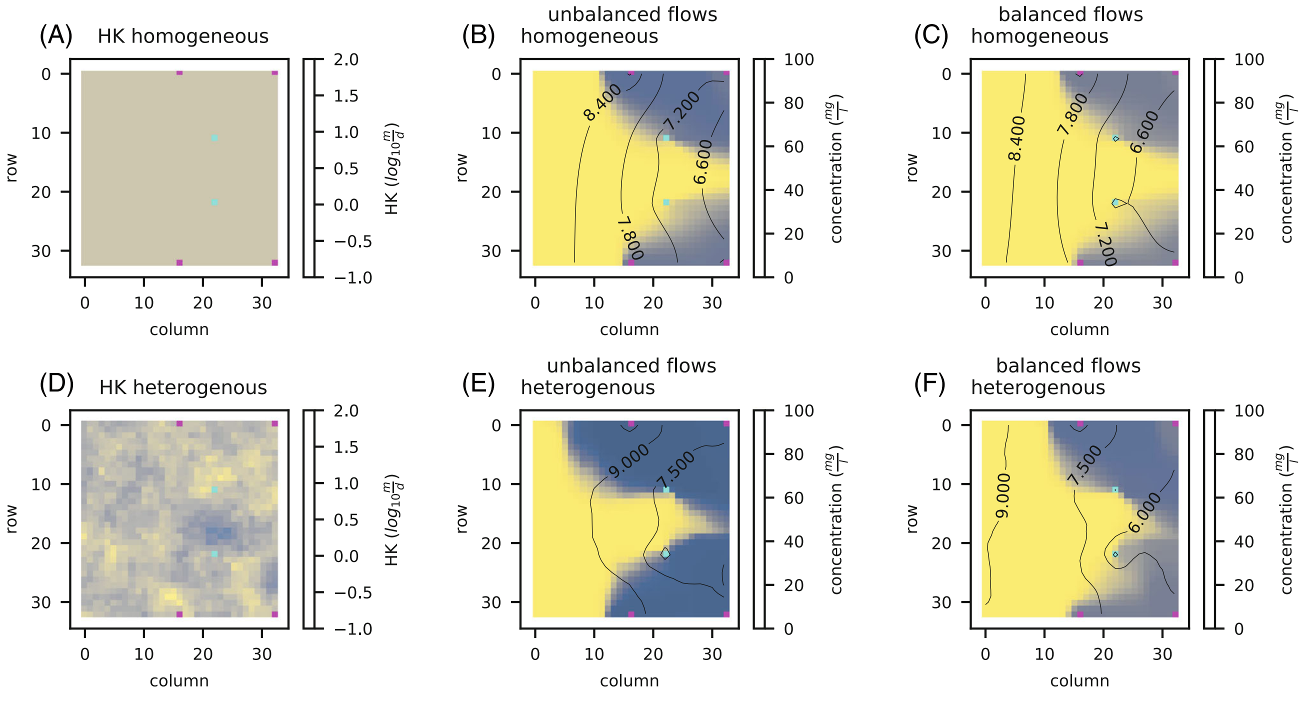

Figure 2. Comparison of the effect of flow-balancing for the case of a homogeneous hydraulic conductivity field (A to C) and heterogeneous hydraulic conductivity field (D to F). Injection wells are shown in magenta, extraction wells are shown in cyan. The flow-balancing has a more pronounced effect on the heterogeneous hydraulic conductivity. Groundwater levels (contours) and dissolved-phase distributions (color flood) for stress period 10 shown on B, C, E, and F.

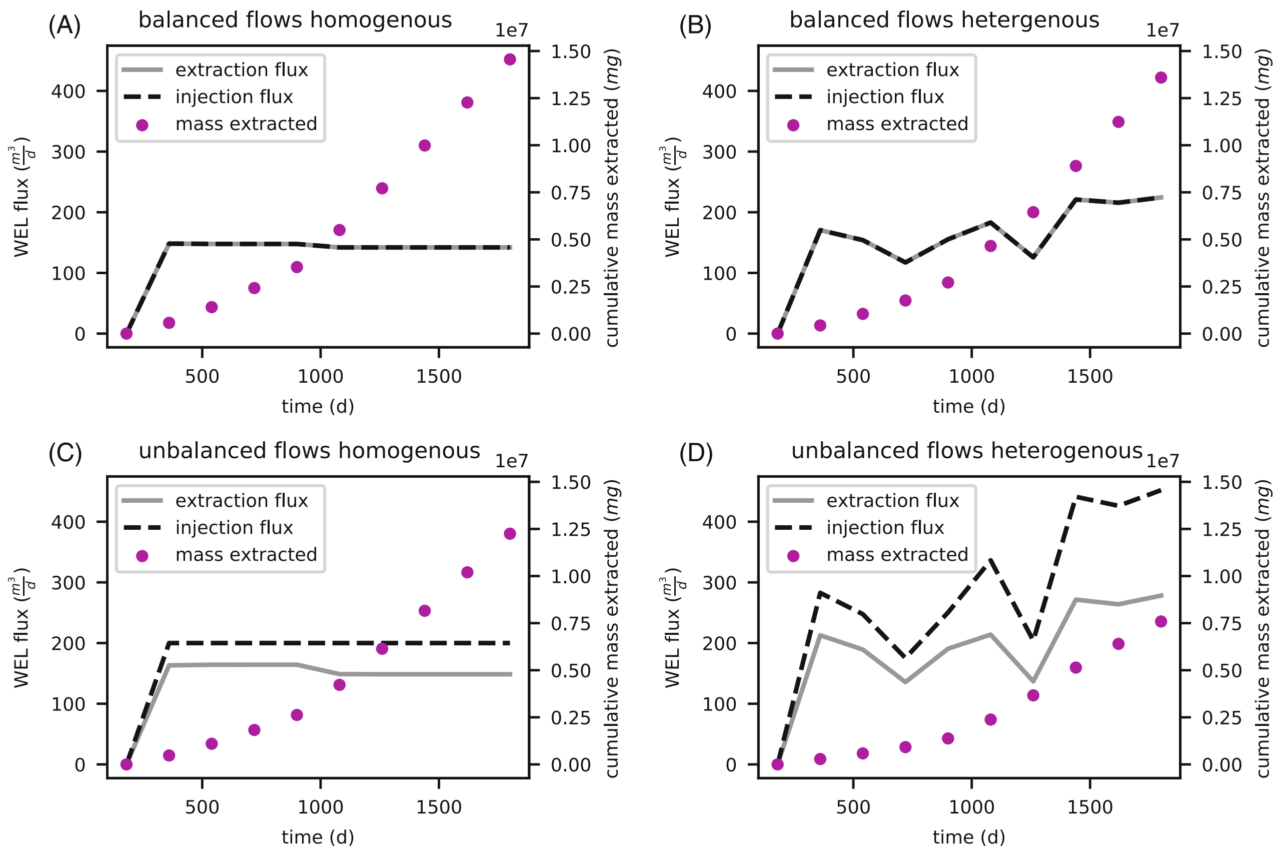

Figure 3. Comparison of the simulated WEL extraction and injection rates for the entire model, and the simulated mass removed by extraction wells. The importance of the flow-balancing is seen for both the homogeneous and heterogeneous cases.

to stress-test the CTS implementation and also better mimic real world CTS operations. For both the homogeneous and heterogeneous cases, an unbalanced flow model run was also completed for comparative purposes.

Results

The results of the homogeneous and heterogeneous cases demonstrate that the MODFLOW 6 CTS package is functioning as expected. The global volumetric water budget for the WEL package inflow and outflows are numerically identical across all stress periods (Figure 3A and 3B) and the reported WEL package groundwater volumetric and dissolved-phase mass fluxes are in numerical agreement between the global water budget summaries reported by MODFLOW 6 and the CTS node and system summaries reported by the CTS implementation.

As expected, the flow-balancing functionality is shown to be quite important for simulating CTS performance, especially in the presence of HK heterogeneity and variable injection and extraction rates (Figure 2B, 2C, 2E, and 2F, and Figure 3C and 3D). The difference between the balanced and unbalanced CTS instances results in substantially different simulated groundwater levels, ultimately leading to differences in the efficacy of the dissolved-phase mass and simulated performance of the CTS instances. This is especially pronounced in the heterogeneous model run, where the unbalanced CTS instances extract approximately 6E+6 mg while the flow-balanced heterogeneous run CTS instances extract approximately 1.2E+7 mg.

Discussion and Conclusion

We expect the MODFLOW 6 CTS implementation to find use in the increasing groundwater modeling industry focus on decision-support modeling analyses such as uncertainty analysis and management optimization under uncertainty. These are settings where requested extraction and injection rates may be varied in programmatic and unexpected ways such that the requested extraction rates may not be met by the simulation results.

It is important to note that while the computational overhead of driving MODFLOW 6 through the API is generally negligible compared to executing statically compiled MODFLOW 6 (Hughes et al. 2022), the flow-balancing functionality can introduce additional nonlinearity in the flow-model solution, such that additional nonlinear (i.e., "outer") iterations may be required. This is especially true if the injection and extraction wells are located in close proximity to one another such that their hydraulic influences may interact.

The CTS implementation in MT3D-USGS (Bedekar et al. 2016) includes additional functionality to account for external sources and sinks from a CTS instance, as well as other options to simulate the dissolved-phase mass removal process, whereas the MODFLOW 6 CTS implementation presented herein currently only uses treatment efficiency and does not currently support external sources and sinks. Additionally, as currently implemented, the MODFLOW 6 CTS package does not allow for multiple injection/extraction wells to be colocated in the same model cell. Furthermore, the CTS package does not currently account for any "storage" of extracted or treated water. However these limitations could be overcome to meet specific simulation goals by simply modifying the python source file; no modifications to MODFLOW 6 are required. Similarly, as MODFLOW 6 continues to evolve and progress, the CTS implementation we have developed should continue to function as expected, as long as no changes to the MODFLOW 6 API are made. We see this characteristic of the MODFLOW 6 CTS implementation as very powerful and providing a more "future proof" solution, compared to a more traditional MODFLOW package concept, that would either require inclusion into the MODFLOW 6 project or would require maintaining a duplicate MODFLOW 6 code base.

Acknowledgments

We would like to acknowledge two reviewers for constructive feedback. The data files, scripts, and software used for the analyses herein are available as a code repository at https://github.com/INTERA-Inc/mf6cts. Included in the code repository are several variants of the example model presented herein, which are used for continuous integration testing; two example jupyter notebooks are included as well. To use the MODFLOW 6 API through python, the following dependencies are required (in addition to the operating-system-specific MODFLOW 6 shared library file):

- `numpy` https://numpy.org/
- `bmipy` https://pypi.org/project/bmipy/
- `xmipy` https://pypi.org/project/xmipy/
- `modflow6api` https://pypi.org/project/modflowapi/

We note that except for `numpy`, all of these dependencies are included in the code repository, so that users only need to install `numpy`.

Authors' Note

The authors do not have any conflicts of interest or financial disclosures to report.

Appendix

MODFLOW 6 CTS Package Users Guide

The `Mf6Cts` class is designed to work with separate flow-then-transport modeling scheme. As such, the flow model and transport models should be in separate directories. Additionally, the `FMI` process is expected to be used, and, if the `MAW` package is used in the flow model, then the `MWT` package is expected in the transport model. Users are expected to make the flow-model binary output files needed by the FMI available.

The CTS input file follows the standard MODFLOW 6 input format style. Through the use of period-data blocks, a CTS instance can be (re-)defined for each stress period in the model. Each period-data block must identify the CTS instance that is being defined. In this way, numerous different CTS instances can be (re-)defined at the stress period level. An optional efficiency can be specified; efficiency ranges from 0.0 (no dissolved-phase mass is removed) to 1.0 (all dissolved-phase mass is removed). If not specified, efficiency is assumed 1.0—the CTS instance removes all mass.

Within each period block, information related to the components of the CTS instance must be defined. These include the component package type (e.g., "wel," "maw," and so on), the component instance of the package type (as defined in the flow model nam file, e.g., "wel_1," "maw_cts," and so on), whether the component is an extractor ("out") or an injector ("in"), and the indexing information for the component. For non-MAW type components, if the model is a structured grid, then the indexing information is the layer, row and column location of the component; if the model is unstructured, then the indexing information should be node-number of the component. For MAW-type components, the indexing information is the "wellno" value in the MAW package.

Below is an example CTS input file:

```
begin period 2 cts 1 efficiency 0.361
wel wel_0 out 3 21 21
wel wel_0 out 3 21 23
maw maw_0 out 2
wel wel_0 in 1 1 1
maw maw_0 in 1
end period 2 cts 1
begin period 2 cts 2 efficiency 0.909
wel wel_0 out 3 20 3
wel wel_0 in 1 33 1
wel wel_0 in 1 33 33
wel wel_0 in 1 1 33
end period 2 cts 2
begin period 5 cts 1 efficiency 0.925
maw maw_0 out 2
maw maw_0 in 1
end period 3 cts 1
begin period 7 cts 2 efficiency 0.222
wel wel_0 out 3 20 3
wel wel_0 in 1 33 1
wel wel_0 in 1 33 33
wel wel_0 in 1 1 33
end period 3 cts 2
```

In this example file, we see that for stress period 2, CTS instance "1" has three extraction wells and one injection well, while CTS instance “2” has one extraction well and three injection wells. Using the MODFLOW 6 convention, CTS instance “1” continues from stress period 2 to stress period 5 as it is defined in stress period 2. Similarly, CTS instance 2 continues from stress period 2 to stress period 7 as it was defined in stress period 2. At stress period 5, CTS instance “1” is reconfigured to be a single injector and single extractor system of just MAW-type wells. At stress period 7, CTS instance “2” changes its efficiency value but contains the same injection and extraction wells.

The `Mf6Cts` class can also be driven from the command line. For example:

```
python mf6cts.py <config_file.py>
```

where `<config_file.py>` is the name of a configuration file that is a simple python source file listing the required arguments:

- `cts_filename`: the name of the cts file
- `lib_name`: the name of the MF6 shared library file
- `transport_dir`: the directory holding the transport model files
- `flow_dir`: the directory holding the flow model files
- `is_structured`: a boolean flag indicating if the model is a structured grid (a value of `1` indicates a structured grid)
- `flow_output_files`: a python list of the flow model output binary files that the `FMI` package is expecting
- `solve_gwf`: (optional) a boolean flag to solve the flow model, default is “True”
- `transfer_flow_output_files`: (optional) a boolean flag to transfer the flow-model output files, default is “True”
- `solve_gwt`: (optional) a boolean flag to solve the transport model, default is “True”

An example configuration file:

```
cts_filename='model.cts'
lib_name='libmf6.so'
transport_dir='fivespot_maw_t_api'
flow_dir='fivespot_maw_api'
is_structured=True
flow_output_files=['gwf.hds',
  'gwf.bud','gwf.maw.bud']
solve_gwf=True
transfer_flow_output_files=True
solve_gwt=True
```

Note the last three lines represent optional arguments.

The MODFLOW 6 CTS package writes several comma-separated-value (CSV) files summarizing the performance of the CTS package for the both the flow and transport models and from both the CTS node and CTS instance perspective. The flow model summary CSV files are written in the flow model directory (e.g., `flow_dir`) and are named “gwf_cts_flow_node_-summary.csv” and “gwf_cts_flow_system_summary.csv”; these files summarize the flow-model extraction and injection aspects of CTS instance performance and include information about the requested and actual rates and cumulative volumes extracted and/or injected for each CTS instance and corresponding nodes across all flow solution stress periods and time steps.

Flow-model node summary file data columns:

- stress_period: stress period number
- time_step: time step number
- ctime: cumulative simulation time
- dt: length of current stress-period/time step
- cts_system: cts system instance number
- package: flow model package type
- instance: flow model package name
- inout: “in” (injector) or “out” (extractor)
- index: the indexing number for this node. Varies depending on package type. For `MAW`, this is the “wellno” value. For `WEL`, this is the node number
- requested_rate: the requested rate in the flow model input file for the current stress period/time step
- actual_rate: the actual rate used in the flow solution for the current stress period/time step
- requested_cum_vol: the requested cumulative volume for the current stress period/time step
- actual_cum_vol: the actual cumulative volume for the current stress period/time step

Flow-model system summary file data columns:

- stress_period: stress period number
- time_step: time step number
- ctime: cumulative simulation time
- dt: length of current stress-period/time step
- cts_system: cts system instance number
- num_injectors: number of injection wells in this CTS system
- num_extractors: number of extraction wells in this CTS system
- requested_rate: the requested rate in the flow model input file for the current stress period/time step
- actual_rate: the actual rate used in the flow solution for the current stress period/time step
- requested_cum_vol: the requested cumulative volume for the current stress period/time step
- actual_cum_vol: the actual cumulative volume for the current stress period/time step

The transport-model summary CSV files are written in the transport model directory (e.g., `transport_dir`) and are named “gwt_cts_node_summary.csv” and “gwt_cts_system_summary.csv”; these files summarize the transport-model extraction and injection aspects of CTS instance performance and include information about the extracted and injected dissolved-phase mass as both a rate and cumulative mass for each stress period and time step, as well as cumulative mass across the entire solution period. These files also contain information about the blended injected concentration if an efficiency less than 1.0 is used.

Transport-model node summary file data columns:

- stress_period: stress period number
- time_step: time step number
- ctime: cumulative simulation time
- dt: length of current stress-period/time step
- cts_system: cts system instance number
- package: flow model package type
- instance: flow model package name
- index: the indexing number for this node. Varies depending on package type. For `MAW`, this is the "wellno" value. For `WEL`, this is the node number
- flow_rate: the actual flow rate for the current stress period/time step
- cum_vol: the actual cumulative volume for the current stress period/time step
- concen: the concentration at this node for the current stress period/time step
- mass_rate: the mass flux rate for this node for the current stress period/time step
- mass: the mass extracted/injected for this node for the current stress period/time step
- cum_mass: the cumulative mass injected/extracted for this node for the current stress/time step

Transport-model system summary file data columns:

- stress_period: stress period number
- time_step: time step number
- ctime: cumulative simulation time
- dt: length of current stress-period/time step
- cts_system: cts system instance number
- num_injectors: number of injection wells in this CTS system
- num_extractors: number of extraction wells in this CTS system
- flow_rate: the actual flow rate for the current stress period/time step
- cum_vol: the actual cumulative volume for the current stress period/time step
- mass_removed: the mass extracted for the current stress period/time step
- cum_mass_removed: the cumulative mass extracted
- mass_treated: the mass "treated" for the current stress period/time step by the application of the "efficiency" factor
- cum_mass_treated: the cumulative mass treated
- mass_injected: the mass injected for the current stress period/time step
- cum_mass_injected: the cumulative mass injected
- concen_injected: the concentration of the injected water
- requested_efficiency: the requested treatment efficiency for the current stress period/time step

If MAW-type boundary conditions are included in CTS instance, then a MAW-specific summary file is written in the transport model directory named "gwt_maw_node_summary.csv." This file summarizes the dissolved-phase transport aspects of individual nodes that comprise each MAW boundary condition, including flow rate, concentration, cumulative volume and cumulative mass for each node in each MAW boundary condition that is included in a CTS instance.

Transport-model `MAW`-specific node summary file data columns:

- stress_period: stress period number
- time_step: time step number
- ctime: cumulative simulation time
- dt: length of current stress-period/time step
- cts_system: cts system instance number
- package: flow model package type
- instance: flow model package name
- maw_wellno: the `MAW` "wellno" value
- index: the node number for this `MAW` node
- flow_rate: the actual flow rate for the current stress period/time step
- cum_vol: the actual cumulative volume for the current stress period/time step
- concen: the concentration at this node for the current stress period/time step
- mass_rate: the mass flux rate for this node for the current stress period/time step
- mass: the mass extracted/injected for this node for the current stress period/time step
- cum_mass: the cumulative mass injected/extracted for this node for the current stress/time step

References

Bedekar, V., E. Morway, C. Langevin, and M. Tonkin. 2016. MT3D-USGS version 1: A U.S. Geological Survey release of MT3DMS updated with new and expanded transport capabilities for use with MODFLOW. *U.S. Geological Survey Techniques and Methods* 6-A53: 69.

Hughes, J.D., M.J. Russcher, C.D. Langevin, E.D. Morway, and R.R. McDonald. 2022. The modflow application programming interface for simulation control and software interoperability. *Environmental Modelling and Software* 148: 105257.

Hughes, J., C. Langevin, and E. Banta. 2017. Documentation for the MODFLOW 6 framework. *U.S. Geological Survey Techniques and Methods* 6-A57: 40.

Langevin, C., A. Provost, S. Panday, and J. Hughes. 2022. Documentation for the MODFLOW 6 groundwater transport model. *U.S. Geological Survey Techniques and Methods* 6-A61: 56.

Langevin, C., J. Hughes, E. Banta, R. Niswonger, S. Panday, and A. Provost. 2017. Documentation for the MODFLOW 6 groundwater flow model. *U.S. Geological Survey Techniques and Methods* 6-A55: 197.