4.0 THE HYDROBIOGEOCHEM PROGRAM STRUCTURE

4.1 The General Solution Strategy of HYDROBIOGEOCHEM

HYDROBIOGEOCHEM is designed to solve a system of equations describing hydrologic transport and biogeochemical reactions in a reactive multicomponent system. The transport equations are derived from the continuity of mass and the law of flux. The biogeochemical equations are material balances coupled with the mass action and kinetic rate equations. The transport equations, along with initial and boundary conditions, and the equations for the biogeochemical reactions govern the migration and chemical transformation of multicomponent species in saturated-unsaturated media. The major transport processes are advection, dispersion/diffusion, and source/sinks. The major chemical processes are aqueous complexation, adsorption, ion-exchange, precipitation/dissolution, redox, and acid-base reactions. The major microbiological processes are biodegradation and microbial respiration.

Two basic types of equations are necessary for modeling the transport of reactive multicomponent species in the subsurface environment. Transport is described by a set of partial or ordinary differential equations. The biochemical reactions are described by a set of nonlinear algebraic equations. The governing transport equations along with the initial conditions are given by Equations (3.1.1) through (3.1.23). The chemical reaction equations are Equations (3.1.24) through (3.1.48). The steps to solve the governing equations are as follows:

1. Equilibrate the system.

2. Estimate the reaction terms in Equations (3.1.1), (3.1.10), and (3.1.17).

3. Solve Equations (3.1.1) through (3.1.23) to obtain estimate of mobile species concentrations, T_j's, kinetic x_i's, and b_i's.

4. Solve Equations (3.1.24) through (3.1.48) for all other species concentrations.

5. Revise estimate of the reaction terms in Equations (3.1.1), (3.1.10), and (3.1.17).

6. Solve Equations (3.1.1) through (3.1.23) to obtain new estimate of T_j's, kinetic x_i's, and b_i's.

7. Compare the newly obtained values with the prior estimates from Step 3.

8. If the difference is within the error tolerance, proceed to the next time step. If the difference is greater than the tolerance, repeat Steps 2 through 7 until the system converges.

4.2 Description of HYDROBIOGEOCHEM Subroutines

HYDROBIOGEOCHEM consists of a short MAIN routine, 125 subroutines, and a function. The MAIN module is used to specify the sizes of all arrays, read data file names, and open data files. The control and coordination activities are performed by subroutine GM2D. The linkage between the hydrologic transport model and biogeochemical reaction model is performed by the subroutine OCSPIT. The subroutines called by GM2D can be classified into four major categories:

- 31 subroutines are used to perform hydrologic transport.
- 25 subroutines and the function are used to compute the Lagrangian concentrations.
- 55 subroutines are used to simulate chemical equilibrium and kinetics.
- 13 subroutines are to perform utility functions.

Figure 4.1 shows the structure of the program. The subroutines are described below.

Subroutine GM2D: Subroutine GM2D controls the entire sequence of operations, a function generally performed by the MAIN program. It is preferable, however, to keep a short MAIN module and utilize variable storage allocation. This makes it possible to deal with a site-specific problem without making changes in array dimensions throughout all subroutines.

Subroutine GM2D will calculate

- The steady-state solution (KSS = 0 and NTI = 0 in Data Set 2 of Appendix A).
- A transient solution using the steady-state solution as the initial conditions (KSS = 0, NTI > 0).
- A transient solution using user-supplied initial conditions (KSS = 1, NTI > 0).

Subroutine GM2D calls:

- Subroutine DATAHT to read and print input data required for hydrologic transport calculations.
- Subroutine CONECT to arrange node connections in ascending order, to locate the diagonal entry for each row and to compute the number of entries in each row.
- Subroutine READR to read the liquid density field.

- Subroutine DATACS to read and print input data required for biogeochemical calculations.

- Subroutine AFABTA to compute the upstream weighting factor.

- Subroutine NODVAL to compute values of the moisture content, bulk density, initial cation exchange capacity, capacitances, and surface area of adsorbent sites at nodes.

- Subroutine VELDC to calculate the average darcy velocity at each node over the current time step and the dispersion coefficient for each element.

- Subroutine EFCTVK to compute the velocity of the fluid with respect to the solid, and the components of the tracking velocity and the constant K which are independent of species concentration.

- Subroutine SSBV to compute source/sink and boundary values.

- Subroutine OCSPIT to obtain individual chemical and microbial species distributions and total dissolved concentrations, total sorbed concentrations, and total precipitated concentrations of all components at all nodes.

- Subroutine TWDVK to compute the concentration dependent velocity and concentration dependent first order rate constant.

- Subroutine ADVWRK to prepare working arrays for Lagrangian integration.

- Subroutine LGRNSTP to perform the Lagrangian Step.

- Subroutine ELRNSTP to perform the Eulerian Step for the hydrologic transport calculations.

- Subroutine SFLOW to compute the net rate of chemical flow through open boundaries.

- Subroutine CALKD to compute equivalent K_d values.

- Subroutine PRINTT to print the results.

- Subroutine PRITER to print the intermediate results between hydrologic transport and biogeochemical iterations.

- Subroutine STORE to store the results for plotting.

Subroutines for Utility Functions:

Subroutine DATAHT: is called by GM2D to read the system geometry and hydrologic transport input data from Data Sets 2 through 15 as described in Appendix A. It also prints all the input information to a tabular output file. DATAHT calls subroutine SURF to identify the boundary segments and boundary nodes, and subroutines READR and READN to automatically generate real and integer numbers, respectively, and subroutine LNDGEN to generate the relationships between node number and equation number when pointwise iteration solution strategies are used.

- **Subroutine SURF**: is called by DATAHT to identify the boundary sides, sequence the boundary nodes, calculate the boundary side length, and compute the directional cosine of the surface sides. The mapping from boundary nodes to global nodes is stored in NPBB(I), where NPBB(I) is the global node number of the I-th boundary node. The element number associated with the boundary sides is stored in MBES. The length and directional cosines for each side are stored in DLB and DCOSXB and DCOSZB, respectively. The local and global nodal numbers of two nodes of each side are stored in ISB. SURF calls Subroutine SLBDY to calculate the shortest element side connected to each boundary node, SLSCAL. The information contained in NPBB, MBES, ISB, DLB, DCOSXB, DCOSZB, and SLSCAL along with the number of boundary nodes and the number of boundary sides, are returned to subroutine DATAHT for use by other subroutines.

- **Subroutine READR**: is called by DATAHT to generate real numbers for Data Sets 7 and 11 described in Appendix A. Automatic generation of regularly patterned data is built into this subroutine. READR is also called by GM2D to generate real numbers for Data Set 16.

- **Subroutine READN**: is called by DATAHT to generate integers for Data Sets 9, 13, 14, and 15 described in Appendix A.

- **Subroutine LNDGEN**: is called by DATAHT to preprocess the pointer arrays needed to assemble the global matrix in compressed form when pointwise solution methods are used. These pointer arrays are generated based on the element connectivity IE(M,J), which is the global node number of the J-th node of element M. Pointer array LRN(J,N) is the global node number of the J-th node connected to the global node N. Pointer array LRL(K,N) is the global element number of the K-th element connected to the global node N, and array NLRL(N) contains the number of elements connected to global node N.

Subroutine CONECT: is called by GM2D to arrange nodal connections in ascending order, to locate the diagonal entry for each row, and to compute the number of entries in each row.

Subroutine DATACS: is called by GM2D to read the biogeochemical system input data from Data Sets 17 through 28 as described in Appendix A and prints all input information.

Subroutine SFLOW: is called by GM2D to compute mass fluxes through all open boundary nodes.

Subroutine CALKD: is called by GM2D to calculate the equivalent partition coefficient K_d for all components.

Subroutine PRINTT: is called by GM2D to print the transport variables. These include the total analytical concentrations, total dissolved concentrations, total sorbed concentrations, total precipitated concentrations of all components, and the negative logarithm of concentration for all component species, and the concentration of all kinetic chemical product species or microbial species.

Subroutine PRITER: is called by GM2D to print the intermediate total analytical concentrations of all components between hydrologic transport and geochemical reaction iterations.

Subroutine STORE: is called by subroutine GM2D to store the transport variables on Logical Unit 12 in binary form. This data is intended for plotting and other post processing use and includes region geometry and transport variables.

Subroutines for Lagrangian Concentration Calculations:

Subroutine ADVWRK: is called by GM2D to prepare all the working arrays to be used in particle tracking in LGRNSTP. This subroutine is called only when the transient simulation is required.

Subroutine LGRNSTP: is called by GM2D to implement the Lagrangian step to obtain the Lagrangian value of the total analytical concentrations for the component/species being subjected to trasnport. This subroutine calls ADVBC to implement the upstream and downstream boundary conditions and GNTRAK to perform the backward particle tracking and interpolation.

Subroutine ADVBC: is called by LGRNSTP to implement the variable and Dirichlet boundary conditions upstream and downstream of the particle track. For a Dirichlet boundary, the Lagrangian concentration is specified. For variable boundaries, the fictitious particle associated with the boundary node must come from the interior nodes if the flow is directed out of the region or from the incoming fluid if the flow is directed into the region. Hence for the downstream (or outflow) variable boundary nodes, the Lagrangian concentration for the boundary node is computed in subroutine GNTRAK when particle tracking within the interior is performed and the implementation for such a boundary segment is bypassed. For upstream (or inflow) variable boundary nodes, the incoming fluid concentration is specified if the flow is directed into the region. The Lagrangian concentration is then calculated according to Eq. (3.2.55), repeated

here:

$$T_{vi}^{*} = \frac{\int_{B_v} N_i \mathbf{n} \cdot \mathbf{V} C_{in} dB}{\int_{B_v} N_i \mathbf{n} \cdot \rho_\ell \mathbf{V} \frac{C'}{T'} T' dB} \qquad (4.2.1)$$

This subroutine calls

- **Subroutine Q2ADVB** to implement the variable boundary condition for a given boundary side, applying mass lumping. It evaluates the following integrations for a boundary segment:

$$RIQ(i) = \int_{B_v} N_i \mathbf{n} \cdot \mathbf{V} C_{in} dB \qquad (4.2.2)$$

$$RLQ(i) = \int_{B_v} N_i \mathbf{n} \cdot \rho_\ell \mathbf{V} \frac{C'}{T'} T' dB \qquad (4.2.3)$$

where

RIQ(i) = the rate of chemical passing through node i.
RLQ(i) = the rate of liquid passing through node i.

Subroutine GNTRAK: is called by LGRNSTP to perform the backward particle tracking and interpolation. It is designed to get the Lagrangian concentrations of all the particles sitting on the global nodes at the current time step. In the subroutine, each particle is tracked one element by one element until either the tracking time is completely consumed or the particle encounters a specified boundary side. This subroutine calls:

- **Subroutine REPLAS:** to set values of x, y, vx, and vy at point P.
- **Subroutine ELENOD:** to determine the shape (triangular or quadrilateral) for a specified element.
- **Subroutine WRKARY:** to prepare working arrays for particle tracking in an element or subelement.
- **Subroutine ELETRK:** to track a particle using the "in-element" approach. This

subroutine is described in further detail below.

- **Subroutine FIXCHK:** to process particle tracking after the particle encounters a boundary. This subroutine calls:

    - **Subroutine ALGBDY** to achieve particle tracking along an unspecified boundary. Tracking is executed one boundary side by one boundary side based on the nodal velocity component along the side being considered. The tracking will not be stopped until either the tracking time is completely consumed or the particle encounters a specified boundary side. ALGBDY calls ELENOD to determine an element's shape.

- **Subroutine CKCOIN:** to examine whether a given node corresponds with node N1 or N2.

- **Subroutine CKCNEL:** to determine the elements that are connected to the element side with nodes N1 and N2 as its ends.

- **Subroutine INTRP:** to compute the Lagrangian concentration by interpolation. This subroutine calls:

    - **Subroutine BASE1** to determine the shape function values associated with a specified point based on the given two-dimensional global coordinates. BASE1 calls **Subroutine XSI2D** to compute the local coordinate of an element based on the given original global coordinates.

Subroutine ELETRK: is called by LGRNSTP to track a particle using the "in-element" approach. In order to increase the accuracy of particle tracking, regular refinement is used to divide the element into as many subelements as the user specifies, and the tracking is executed one subelement by one subelement. Tracking is not stopped until either the tracking time is completely consumed or the particle encounters a side of the element. This subroutine calls subroutines REPLAS, CKCNEL, WRKARY, and CKCOIN described above and :

- **Subroutine VALBDL:** to determine the velocity at a point by interpolation.

- **Subroutine MMLOC:** to locate the starting point for the "in-element" particle tracking in a specified element. MMLOC calls REPLAS to set values at point P and calls **Subroutine ONLINE** to assure that point (XP,YP) is on the line segment with nodes J1 and J2 as its end nodes.

- **Subroutine TRACK2:** to determine (1) if the particle will pass through the working subelement and (2) where the ending point is if the particle passes through the working element. This subroutine is employed when the starting point is not on any of the working subelement nodes. For accuracy, the average

velocity of both the source point and the target point is first considered in locating the target point. However, if this average velocity approach is not able to deal with very complex velocity fields, the single velocity of the source point is used to determine the location of the target point. This subroutine calls:

- **Subroutine LOCQ2D** to determine the location and velocity at the target point, q, as well as the available time for further tracking when the target side is specified. This subroutine calls **Subroutine CHNGSGN** to change the sign of velocity arrays and **Subroutine SURE2D** to locate the target point for a tracking path when the single velocity approach is employed.

- **Subroutine IBWCHK:** to determine the two global nodes whose associated element side is where the target point rests at the end of the "in-element" particle tracking in a specified element.

- **Subroutine TRACK1:** to determine (1) if the particle will pass through the working subelement and (2) where the ending point is if the particle passes through the working element. This subroutine is employed when the starting point is one of the working subelement nodes. As in Subroutine TRACK2, the average velocity of both the source point and the target point is considered first for accuracy in locating the target point. If this average velocity approach is not able to deal with very complex velocity fields, the single velocity of the source point is used to determine the location of the target point. This subroutine calls Subroutine LOCQ2D.

Subroutines for Hydrologic Transport Calculations:

Subroutine AFABTA: is called by GM2D to compute the optimum weighting factors for all sides of an element. The results are stored in array WETAB(J,M), where M = 1, 2, ..., NEL and J = 1, 2, 3, 4 for quadrilateral elements and J = 1, 2, 3 for triangular elements.

Subroutine NODVAL: is called by GM2D to convert the values of the cation ion exchange capacities and the surface area and capacitance of adsorbing sites, the moisture content, and the bulk density of the materials associated with each element into values at the nodal points. It examines the size and properties associated with each element connected to each node and determines a weighted average value for these properties at each node.

Subroutine VELDC: is called by GM2D to calculate the average darcy velocity (specific discharge) at each node over the current time step and the dispersion coefficient for each element.

$$\mathbf{AKDC(3,4,M)} = \rho_\ell \theta \mathbf{D} \qquad \mathbf{(4.2.4)}$$

Subroutine EFCTVK: is called by GM2D to compute the velocity of the fluid with respect to the solid (pore water velocity) and the components of the tracking velocity and the first order rate constant K which are independent of species concentration at each node:

$$\mathbf{EFCTV(NP,2)} = \rho_\ell \mathbf{V} \tag{4.2.5}$$

$$\mathbf{EFCTK(NP)} = -\frac{\partial \rho_\ell \theta}{\partial t} \tag{4.2.6}$$

Subroutine SSBV: is called by GM2D to compute the source/sink and boundary values for the species subject to transport. SSBV linearly interpolates the input source/sink and boundary condition profiles to values at the current time. SSBV calls **Subroutine INTERP** to perform the interpolation.

Subroutine TWDVK: is called by GM2D to compute the values at each node for the concentration dependent velocity and concentration dependent first order rate constant for each species subject to transport:

$$\begin{aligned} \mathbf{VTRK(NP,2)} &= \mathbf{EFCTV(NP,2)}\,\frac{C'}{T'} - \mathbf{AKDC}\cdot\nabla\frac{C'}{T'} \\ &= \left[\rho_\ell \mathbf{V}\frac{C'}{T'} - \rho_\ell \theta\, \mathbf{D}\cdot\nabla\frac{C'}{T'}\right] \end{aligned} \tag{4.2.7}$$

$$\begin{aligned} \mathbf{RATK(NP)} &= \mathbf{EFCTK(NP)}\frac{C'}{T'} + \mathbf{EFCTV(NP,2)}\cdot\nabla\frac{C'}{T'} - \nabla\cdot\left[\mathbf{AKDC}\cdot\nabla\frac{C'}{T'}\right] - \mathbf{EFCTK(NP)} \\ &= \left[\frac{\partial \rho_\ell \theta}{\partial t} + \rho_\ell \mathbf{V}\nabla\cdot\frac{C'_j}{T'_j} - \frac{C'_j}{T'_j}\frac{\partial \rho_\ell \theta}{\partial t} - \nabla\cdot\left(\rho_\ell \theta \mathbf{D}\cdot\nabla\frac{C'_j}{T'_j}\right)\right] \end{aligned} \tag{4.2.8}$$

This subroutine calls

- **Subroutine GRADFN**: to calculate the gradient of (C/T). GRADF is a general subroutine to calculate the gradient of any specified function using the finite element method. GRADFN calls **Subroutine Q4D** to calculate the element matrix for quadrilateral elements and **Subroutine Q3D** to calculate the element matrix for triangular elements. Q4D calls **Subroutine SHAPE** to evaluate the base and weighting functions and derivatives of the base functions at a Gaussian point.

- **Subroutine DDOTGFN**: to calculate $\rho_\ell\, \theta\, \mathbf{D} \bullet \nabla$ (C/T). DDOTGFN is a general

subroutine to calculate $\rho_\ell \theta \mathbf{D} \bullet$ the gradient of any specified function using the finite element method. DDOTGFN calls Q4D and Q3D.

- **Subroutine DIVVEC**: to calculate $\nabla \bullet \{ \rho_\ell \theta \mathbf{D} \bullet \nabla (C/T)\}$. DIVVEC is a general subroutine to calculate the divergence of any specified vector function using the finite element method. DDOTGF calls **Subroutine Q4DIV** to to calculate the element matrix for quadrilateral elements and **Subroutine Q3DIV** to to calculate the element matrix for triangular elements.

Subroutine ELRNSTP: is called by GM2D to implement the Eulerian step of the hydrologic transport calculations. This subroutine calls ASEMBL to compose the matrix equations, BC to implement the boundary conditions, and SOLVE, PISS, PPCG, ILUCG, MICPCG, or SSORCG to solve the matrix equations.

Subroutine ASEMBL: is called by subroutine ELRNSTP to compose the matrix equations. It sums over all element matrices to form the compressed global matrix CMTRX and the global load vector RLD. This subroutine calls:

- **Subroutines Q4 and Q3**: to compute the element matrices for quadrilateral and triangular elements, respectively, given by

$$A^e_{\alpha\beta} = \int_{R_e} N^e_\alpha (\rho_\ell \theta) N^e_\beta dR \quad , \tag{4.2.9}$$

$$D^e_{\alpha\beta} = \int_{R_e} \left(\nabla N^e_\alpha \right) \cdot \rho_\ell \theta \mathbf{D} \frac{C'}{T'} \cdot \left(\nabla N^e_\beta \right) dR \tag{4.2.10}$$

$$C^e_{\alpha\beta} = \int_{R_e} N^e_\alpha \left[\frac{\partial \rho_\ell \theta}{\partial t} + \rho_\ell \mathbf{V} \nabla \cdot \frac{C'}{T'} - \frac{C'}{T'} \frac{\partial \rho_\ell \theta}{\partial t} - \nabla \cdot \left(\rho_\ell \theta \mathbf{D} \cdot \nabla \frac{C'}{T'} \right) \right] N^e_\beta dR \tag{4.2.11}$$

$$Q^e_{\alpha\beta} = \int_{R_e} N^e_\alpha Q\rho^* \frac{C'}{T'} N^e_\beta dR \tag{4.2.12}$$

Subroutines Q4 and Q3 also calculate the element load vectors given by

$$R^e_\alpha = \int_{R_e} N^e_\alpha R^c dR \tag{4.2.13}$$

and

$$S_\alpha^e = \int_{R_e} N_\alpha^e Q C_{in} dR \qquad (4.2.14)$$

Subroutine Q4 calls subroutine SHAPE to evaluate the base and weighting functions and derivatives of the base functions at a Gaussian point.

Subroutine BC: is called by subroutine ELRNSTP to incorporate the Dirichlet and variable-boundary conditions into the the compressed global matrix CMTRX and the global load vector RLD. For a Dirichlet boundary condition, an identity algebraic equation is generated for each Dirichlet nodal point. Any other equation having this nodal variable is modified accordingly to simplify computations. Subroutine BC implements the variable-boundary conditions by calling subroutine Q2VB:

- **Subroutine Q2VB**: is called by subroutine BC to compute the contributions of a variable-boundary segment to the global matrix equation. It computes a two-by-two matrix, QB, and a load vector, QR, for each variable-boundary segment as follows:

$$QB(I,J) = \int_{B_e} N_\alpha^e \mathbf{n} \cdot \left(\rho_\ell \mathbf{V} \frac{C'}{T'} - \rho_\ell \theta \mathbf{D} \cdot \nabla \frac{C'}{T'} \right) dB \quad , \text{if } (\mathbf{n} \cdot \mathbf{V}) < 0 \quad , \qquad (4.2.15)$$

$$QB(I,J) = \int_{B_e} N_\alpha^e \left[\mathbf{n} \cdot \left(\rho_\ell \mathbf{V} \frac{C'}{T'} - \rho_\ell \theta \mathbf{D} \cdot \nabla \frac{C'}{T'} \right) - \mathbf{n} \cdot \rho_\ell \mathbf{V} \frac{C'}{T'} \right] dB \quad , \text{if } (\mathbf{n} \cdot \mathbf{V}) \geq 0 \quad , \qquad (4.2.16)$$

$$QR(I) = - \int_{B_e} N_\alpha^e (\mathbf{n} \cdot \mathbf{V}) C_{in} dB \quad , \text{if } (\mathbf{n} \cdot \mathbf{V}) < 0 \quad , \qquad (4.2.17)$$

and

$$QR(I) = 0 \quad , \text{if } (\mathbf{n} \cdot \mathbf{V}) \geq 0 \quad . \qquad (4.2.18)$$

Subroutine SOLVE: is called by subroutine ELRNSTP to solve a matrix equation of the type

$$[C]\{x\} = \{y\} \quad , \qquad (4.2.19)$$

where

[C] = coefficient matrix.
{x} = unknown vector to be determined.
{y} = known load vector.

The subroutine returns the solution {y} and stores it in {y}. SOLVE uses a standard-banded,

Gaussian direct elimination procedure.

Subroutine PISS: is called by subroutine ELRNSTP to solve the linearized matrix equation with point iteration solution strategies when specified by the user.

Subroutine PPCG: is called by subroutine ELRNSTP to solve the linearized matrix equation with the preconditioned conjugate gradient method using the polynomial as a preconditioner. PPCG calls to **Subroutine POLYP** to invert the preconditioner.

Subroutine ILUCG: is called by subroutine ELRNSTP to solve the linearized matrix equation with the preconditioned conjugate gradient method using the incomplete Cholesky decomposition as a preconditioner. It calls to **Subroutine LLTINV** to invert the preconditioner.

Subroutine MICPCG: is called by subroutine ELRNSTP to solve the linearized matrix equation with the preconditioned conjugate gradient method using the modified incomplete Cholesky decomposition as a preconditioner. It calls to **Subroutine MICP** to invert the preconditioner.

Subroutine SSORCG: is called by subroutine ELRNSTP to solve the linearized matrix equation with the preconditioned conjugate gradient method using the symmetric successive over-relaxation as a preconditioner. It calls to **Subroutine SSORP** to invert the preconditioner.

Subroutines for Biogeochemical Subsystem Calculations:

Subroutine OCSPIT: is called by subroutine GM2D. It calculates the individual chemical and microbial species concentrations and the total dissolved, total sorbed, and total precipitated concentrations of all components by calling the subroutine KEMOD. It then calculates the rate of change of all species subject to transport based on the new concentrations determined by KEMOD by calling subroutine RATECB. Finally, it calls to LPOUT to print chemical species distributions.

Subroutine KEMOD: The subroutine KEMOD is called by the subroutine OCSPIT and will perform either the steady-state computation alone (KSS = 0 and NTI = 0), or a transient state computation using the steady-state solution as the initial conditions (KSS = 0, NTI > 0), or a transient computation using user-supplied initial conditions (KSS = 1, NTI > 0). It initializes the concentrations of all product species given the estimate of component concentrations from subroutine OCSPIT. KEMOD then calls to subroutine KINEQL to solve a set of mixed ordinary differential and algebraic equations governing mole balance, and kinetic and equilibrium reactions. It calls to subroutine TOTDSP to compute total dissolved, total sorbed, and total precipitated concentrations of all components after concentrations of all species have been found.

Subroutine KINEQL: This subroutine solves the system of mixed ordinary differential and nonlinear algebraic equations specified by Eqs. (3.1.13) through (3.1.25). The solution is determined with the Newton-Raphson iteration method. For each iteration, subroutine KINEQL

calls:

- Subroutine DGELG to decompose the Jacobian matrix with full pivoting and for back substitution to obtain the differences between new iteration and previous iterations of all unknowns. New iterates are obtained by adding these differences to the old iterations.

Subroutine ACOEF: This subroutine is called by subroutine KINEQL to compute ionic strength and activity coefficients of all species.

Subroutine MODPRK: This subroutine is called by subroutine KINEQL to calculate the modified equilibrium formation constants for all precipitated species.

Subroutine MODBFK: This subroutine is called by subroutine KINEQL to calculate the modified equilibrium constants for all equilibrium reactions and the modified forward and backward rate constants for all chemical kinetic reactions.

Subroutine RESIDU: This subroutine is called to evaluate residuals of discretized ordinary differential and nonlinear algebraic equations governing biogeochemical kinetics and equilibrium. It calls:

- **Subroutine RAQC**: to evaluate residuals for equations governing the mole balance for aqueous components.

- **Subroutine RADC**: to evaluate residuals for equations governing the mole balance for adsorbent components, except for the contribution from reactions.

- **Subroutine RKXY**: to evaluate residuals for the kinetic complexed and adsorbed species, except for the contribution from reactions.

- **Subroutine RIES**: to evaluate residuals for ion-exchanged species, except for the contribution from reactions.

- **Subroutine RPRC:** to evaluate residuals for the kinetic precipitated species, except for the contribution from reactions.

- **Subroutine RMBA:** to evaluate residuals for the microbial species, except for the contribution from reactions.

- **Subroutine RRXNS:** to evaluate the contribution to all residual equations from reactions. This subroutine calls:

    - **Subroutine RATEL1:** to evaluate the contribution from equilibrium reactions (KRTYP = 0). This subroutine calls **Subroutine PRODBF** to

perform some of the repetitive calculations.

- **Subroutine RATEL2:** to evaluate the contribution from chemical kinetic reactions (KRTYP = 1). This subroutine calls **Subroutine PRODBF** to perform some of the repetitive calculations.

- **Subroutine RATEL3:** to evaluate the contribution from Monod kinetic microbial reactions (KRTYP = 2). This subroutine calls **Subroutine BIOLAGK** to determine the lag coefficient for a reaction, **Subroutine BIOINH** to determine the inhibition coefficients for a reaction, and **Subroutine BIORATE** to determine the rate of a reaction.

- **Subroutine RATEL4:** to evaluate the contribution from microbial phase transfer reactions (KRTYP = 3).

Subroutine JACOBI: This subroutine is called to evaluate the Jacobians of discretized ordinary differential and nonlinear algebraic equations governing biogeochemical kinetics and equilibrium. It calls:

- **Subroutine JAQC**: to evaluate Jacobians for equations governing the mole balance for aqueous components.

- **Subroutine JADC**: to evaluate Jacobians for equations governing the mole balance for adsorbent components, except for the contribution from reactions.

- **Subroutine JKXY**: to evaluate Jacobians for the kinetic complexed and adsorbed species, except for the contribution from reactions.

- **Subroutine JIES**: to evaluate Jacobians for ion-exchanged species, except for the contribution from reactions.

- **Subroutine JPRC:** to evaluate Jacobians for the kinetic precipitated species, except for the contribution from reactions.

- **Subroutine JMBA:** to evaluate Jacobians for the microbial species, except for the contribution from reactions.

- **Subroutine JRXNS:** to evaluate the contribution to all Jacobians from reactions. This subroutine calls:

  - **Subroutine JRATE1:** to evaluate the contribution from equilibrium reactions (KRTYP = 0). This subroutine calls **Subroutine PRODBF** to calculate constants for a given reaction that do not depend on the row and column in the Jacobian. It also calls **Subroutines JERZ and JERP** to

evaluate the Jacobian for equilibrium ion-exchanged and precipitated species, respectively. JERZ and JERP call **Subroutine JEQXY** to evaluate the case when an equilibrium aqueous complexed species is involved in the reaction.

- **Subroutine JRATE2:** to evaluate the contribution from chemical kinetic reactions (KRTYP = 1). This subroutine calls **Subroutine PRODBF** to calculate constants for a given reaction that do not depend on the row and column in the Jacobian. It also calls **Subroutine JKRC** to perform some of the repetitive calculations.

- **Subroutine JRATE3:** to evaluate the contribution from Monod kinetic microbial reactions (KRTYP = 2). This subroutine calls **Subroutine BIORATE** to determine the rate of a reaction, and **Subroutine JBIO** to calculate constants for a given reaction that do not depend on the row in the Jacobian. It then calls **Subroutine JCOL** to calculate the Jacobian for a given row from a given reaction for each column. JCOL calls **Subroutine JKRB** to perform some of the repetitive calculations.

- **Subroutine JRATE4:** to evaluate the contribution from microbial phase transfer reactions (KRTYP = 3).

Subroutine DGELG: This subroutine is called by subroutine KINEQL to solve the Jacobian matrix equation using Gaussian elimination with full pivoting.

Subroutine DECOMP: This subroutine is called by subroutine KINEQL to solve the Jacobian matrix equation with partial pivoting if specified by the user.

Subroutine SOLVP: This subroutine is called by subroutine KINEQL to solve the Jacobian matrix equation with partial pivoting if specified by the user.

Subroutine INDEXX: This subroutine is called by subroutine KINEQL to index the saturation value among all potential species that are subject to precipitation/dissolution reactions.

Subroutine NPPT: This subroutine is used to determine the number of species allowed to precipitate without violating the phase rule.

Subroutine DISOLV: This subroutine is called by KINEQL to dissolve an assumed precipitated species that has shown negative concentrations during two successive iterations.

Subroutine GUESS: This subroutine is called by KINEQL to make a better starting guess for species concentrations based on a directional search to

minimize the expected residuals. It calls **Subroutine RESGUES**, which is similar to RESIDU.

Subroutine TOTDSP: This subroutine is called by subroutines KINEQL and KEMOD to evaluate the log of component free species concentrations and total dissolved concentrations, total sorbed concentrations, and total precipitated concentrations of all components.

Subroutine LPOUT: This subroutine is called by the subroutines KINEQL and OCSPIT to line print chemical species distribution at desired nodes at desired time intervals. The output includes concentrations, modified equilibrium constants, and stoichiometric coefficients of all species.

Subroutine RATECB: This subroutine is called by the subroutine OCSPIT to evaluate the rate of change due to all reactions for the species subject to transport. It calls **Subroutine RCHEM** to evaluate the contribution from chemical reactions, **Subroutine RBIOL** to evaluate the contribution from microbial degradation reactions, **Subroutine RBIOXF** to evaluate the contribution from microbial phase transfer reactions, and **Subroutine RBIOER** to evaluate the contribution from microbial endogenous respiration.

5.0 ADAPTATION OF HYDROBIOGEOCHEM TO SITE SPECIFIC APPLICATIONS

The following describes the maximum control integers that must be defined by the user for each site-specific application and the data files that should be prepared.

5.1 Specification of Maximum Control Integers

For each site-specific application, 44 maximum control integers must be assigned by the user with the PARAMETER statements in the MAIN program to specify the size of the problem. The definitions of these parameters are given below.

5.1.1 Maximum Control Integers for the Spatial Domain

MAXNPK = maximum number of nodes.
MAXELK = maximum number of elements.
MXBNPK = maximum number of boundary nodal points.
MXBESK = maximum number of boundary-element surfaces.
MAXBWK = maximum bandwidth size.
MXJBDK = maximum number of nodes connecting to any node.
MXKBDK = maximum number of elements connecting to any node.

5.1.2 Maximum Control Integers for the Time Domain

MXNTIK = maximum number of time steps.
MXDTCK = maximum number of times to reset time step to a small number.

5.1.3 Maximum Control Integers for Source/Sinks

MXSELK = maximum number of distributed source/sink elements.
MXSPRK = maximum number of distributed source/sink profiles (source value vs. time).
MXSDPK = maximum number of data points on each element source/sink profile.
MXWNPK = maximum number of well nodal points.
MXWPRK = maximum number of well source/sink profiles.
MXWDPK = maximum number of data points on each well source/sink profile.

5.1.4 Maximum Control Integers for Variable Boundary Conditions

MXVNPK = maximum number of variable nodal points.
MXVESK = maximum number of variable element surfaces.
MXVPRK = maximum number of variable boundary condition (rainfall) profiles.
MXVDPK = maximum number of data points on each variable boundary condition profile.

5.1.5 Maximum Control Integers for Dirichlet Boundary Conditions

MXDNPK = maximum number of Dirichlet nodal points.
MXDPRK = maximum number of Dirichlet total head profiles.
MXDDPK = maximum number of data points on each Dirichlet profile.

5.1.6 Maximum Control Integers for Material Properties

MXMATK = maximum number of material types.
MXMPMK = maximum number of material properties per material (= 4 for this code version).

5.1.7 Maximum Control Integers for Transport Components

MAXHAK = maximum number of mobile hydro-components
MAXHSK = maximum number of immobile hydro-components
MAXKXK = maximum number of kinetic aqueous complexed chemical species
MAXKYK = maximum number of kinetic adsorbed chemical species
MAXKZK = maximum number of kinetic ion-exchanged chemical species
MAXKPK = maximum number of kinetic precipitated chemical species
MAXBK = maximum number of microbial species (aqueous + adsorbed)

HYDROBIOGEOCHEM calculates:

MAXHK = maximum number of hydrologic transport system components. (=MAXHAK+MAXHSK)
MAXTK = maximum number of mobile hydro-components, kinetic aqueous complexed species, and microbial species. (= MAXHAK+MAXKXK+MAXBK)
MAXT2K = MAXTK + 2
MXHKK = maximum number of hydro compoments and all kinetic species. (=MAXHK+MAXKXK+MAXKYK+MAXKZK+MAXKPK+MAXBK)

5.1.8 Maximum Control Integers for Biogeochemical Reactions

MAXNK = maximum number of chemical components.
MAXMZK = maximum number of ion-exchanging species.
MAXMPK = maximum number of species allowed for precipitation-dissolution.
MAXPDK = maximum number of chemical product species.
MXNIXK = maximum number of ion-exchange sites.
MXNSBK = maximum number of adsorbing sites.
MXRXNK = maximum number of all reactions.
MXRTSK = maximum number of reactant species in any reaction.
MXPDSK = maximum number of product species in any reaction.

HYDROBIOGEOCHEM calculates:

MAXEQK = maximum number of equations to be solved in biochemical reaction subsystem. (= MAXNK+MAXKXK+MAXKYK+MAXMZK + MAXMPK+MAXBK).
MXPDBK = maximum number of chemical product species and microbial species. (=MAXPDK + MAXBK)
MAXMK = maximum number of all species (= MAXNK + MAXPDK+MAXBK)

5.1.9 Maximum Control Integers for Subelement Particle Tracking

MXNPWK = maximum number of nodal points in any element for tracking.
MXELWK = maximum number of subelements in any element for tracking.
NXWK = maximum number of subelements in the x-direction in any element for tracking.
NYWK = maximum number of subelements in the y-direction in any element for tracking.

5.2 Specification of Maximum Control Integers with PARAMETER Statements

Let us assume that a region of interest is discretized into 15 by 89 nodes and 14 by 88 rectangular elements, i.e., 89 nodes along the longitudinal, or x-direction, and 15 nodes along the vertical, or z-direction. Because we have a total of 15 x 89 = 1,335 nodes, the maximum number of nodes is MAXNPK = 1335. The total number of elements is 14 x 88 = 1,232, i.e., MAXELK = 1232. For this simple discretization problem, the maximum number of nodes connected to any of the 1,335 nodes in the region of interest is nine (The nine nodes connecting to any node include the left node, the right node, the lower node, the upper node, the lower-left node, the lower-right node, the upper left node, the upper right node, and the node itself.) for a quadrilateral element (i.e., MXJBDK = 9), and the maximum number of elements connecting to any node is four (i.e, MXKBDK = 4). There will be 14 boundary element sides each on the right and left sides and 88 boundary element sides each on the top and bottom sides of the region, for a total of 204 boundary element sides (i.e., MXBESK = 204). By similar computations, the number of boundary nodes is 204 (i.e., MXBNPK = 204). The bandwidth is equal to 33 (i.e., MAXBWK = 33). (The bandwidth is equal to 2*MAXDIF + 1 where MAXDIF is the maximum difference of the four element indices for any element. For this simple discretization, we number the nodes from 1 to 15 in the first column, 16 to 30 in the second column, etc. For element 1, the four indices are 1, 16, 17, and 2; hence MAXDIF is 17 - 1 = 16. All other elements will have the same MAXDIF. Thus the bandwidth is 2*32 + 1 = 33.)

In this version of the code, there are four material properties per material, so MXMPMK = 4. If we assume that the whole region of interest is composed of three different kinds of materials, then we have MXMATK = 3. Assuming we will complete a 500-time-step simulation and change the time-step size 20 times during our simulation, then we have MXNTIK = 500 and MXDTCK = 20.

Assume that we are dealing with 12 transport components, all mobile (i.e., MAXHAK = 12, MAXHSK = 0). Let us further assume that we will deal with 12 chemical components, (i.e., MAXNK = 12). Among these 12 components, there will be 78 chemical reactions that result in

a maximum of 78 chemical product species, 10 of which may be involved ion-exchange, 8 of which are potentially precipitated species. In addition, there are two microbial species which degrade some of the chemical species per two biodegradation reactions. Thus, we have MAXPDK = 78, MAXMZK = 10, MAXMPK = 8, and MAXBK = 2. There are a total of 80 reactions (78 chemical and 2 microbiological), with a maximum of four reactants and three product species in a single reaction, so MXRXNK = 80, MXRTSK = 4, MXPDSK = 3. Two of the aqueous complexed product species will be formed by kinetic reactions, but all other product species are defined by equilibrium reactions, so we have MAXKXK = 2, MAXKYK = 0, MAXKZK = 0, MAXKPK = 0. If we have three types of ion-exchange sites and four types of adsorption sites (e.g., four colloids), then we have MXNIXK=3 and MXNSBK=4.

Assume there are a maximum of 11 elements with distributed sources/sinks (i.e., MXSELK = 11) and a maximum of 10 nodes that can be considered well (point) sources/sinks (i.e., MXWNPK = 10). Also assume there are three different distributed source/sink profiles (i.e., MXSPRK = 3) and five point source/sink profiles (i.e., MXWPRK = 5). Let us further assume four data points are needed to describe the distributed source/sink profiles as functions of time, and eight data points are required to described the point source/sink profiles (i.e., MXSDPK = 4 and MXWDPK = 8).

To specify maximum control integers for boundary conditions, let us assume that the total analytical concentrations on the left side of the domain and the left half of the top side are known. The right half of the top side, bottom side, and right side are specified as variable boundaries. On the left side there are 15 nodes, and on the left half of the top side there are 44 nodes. Thus, we have a total of 59 Dirichlet nodes, so MXDNPK = 59. Let us assume that the middle nine nodes on the left side have one specified total analytical concentration, the bottom three nodes on the left side have a different specified concentration, and the other 47 nodes have yet another specified total analytical concentration, so MXDPRK = 3. We further assume that these analytical concentrations can be described as a function of time using eight data points, so MXDDPK = 8. There are 147 nodes and 146 element sides on the right half of the top side, bottom side, and right side: MXVNPK = 147 and MXVESK = 146. We also assume there are three different incoming fluid concentrations (i.e., MXDPRK = 3) going into the region through the top, bottom, and right sides, respectively, and each of these three concentrations is a function of time that can be described by two data points. With these descriptions, we have MXVPRK = 3 and MXVDPK = 2.

Finally, to accurately perform particle tracking, let us assume that for each element we need to divide it into 3 by 3 subelements. Thus, we have MXNPWK=16, MXELWK=9, NXWK=3, NYWK=3.

From the above discussion, the following PARAMETER statements can be used to specify the maximum control integers in the MAIN program for the problem at hand. Most compilers will not accept zero as an array dimension, so any maximum control integers that are zero have been specified as =1 instead:

```
PARAMETER (MAXELK=1232,MAXNPK=1335,MXBESK=204,MXBNPK=204)
PARAMETER (MAXBWK=33,MXJBDK=9,MXKBDK=4,MXNTIK=500,MXDTCK=20)

PARAMETER (MXSELK=11,MXSPRK=3,MXSDPK=4)
PARAMETER (MXWNPK=10,MXWPRK=5,MXWDPK=8)

PARAMETER (MXDNPK=59,MXDPRK=3,MXDDPK=8)
PARAMETER (MXVESK=146,MXVNPK=147,MXVPRK=3,MXVDPK=2)

PARAMETER (MXMATK=3,MXMPMK=4)

PARAMETER (MAXHAK=12,MAXHSK=1)
PARAMETER (MAXKXK=2, MAXKYK=1, MAXKZK=1, MAXKPK=1, MAXBK=2)
PARAMETER (MAXHK=MAXHAK+MAXHSK)
PARAMETER (MAXTK= MAXHAK+MAXKXK+MAXBK, MAXT2K=MAXTK+2)
PARAMETER (MXHKK=MAXHK+MAXKXK+MAXKYK+MAXKZK+MAXKPK+MAXBK)

PARAMETER (MAXNK=12,MAXMZK=10,MAXMPK=8,MAXPDK=78)
PARAMETER (MXNIXK=3,MXNSBK=4)
P A R A M E T E R
(MAXEQK=MAXNK+MAXKXK+MAXKYK+MAXMZK+MAXMPK+MAXBK).
PARAMETER (MXPDBK=MAXPDK+MAXBK, MAXMK=MAXNK+MAXPDK+MAXBK)

PARAMETER (MXRXNK=80, MXRTSK=4, MXPDSK=3)

PARAMETER (MXNPWK=16, MXELWK=9, NXWK=3, NYWK=3)
```

If we use the point iteration method to solve the matrix equation instead of the direct band matrix solver, we should set the maximum bandwidth, MAXBWK, equal to the maximum number of nodes connected to any node, MXJBDK. (The solution method is specified in Data Set 2 by input parameter IPNTS. See Appendix A).

5.3 Input and Output Devices

Five logical units are needed to execute HYDROGEOCHEM. Logical Units 15 and 16 are used for data input and line printer output, respectively. Logical Unit 11 is used to read flow variables produced by HYDROFLOW (a simplified version of FEMWATER) if KVI $\geq$ 0. Logical Unit 12 must be specified to store the simulation results in binary form, which can be used for plotting purposes, or for initial conditions for a subsequent simulation. Logical Unit 13 is used to read initial conditions if the restart option is used. If the restart option is active, the file associated with Logical Unit 13 for the current job should be the same file associated with Logical Unit 12 of the previous job.