QUAL2K:

A Modeling Framework for Simulating River and Stream Water Quality (Version 3.02)

Documentation

Boulder Creek, Colorado

Steven C. Chapra, Hua Tao, Kyle F. Flynn, & Greg Pelletier

31 June 2022

DISCLAIMER

The information in this document has been funded partly by the United States Environmental Protection Agency. It is currently being subjected to the Agency's peer and administrative review and has yet to be approved for publication as an EPA document. Mention of trade names or commercial products does not constitute endorsement or recommendation for use by the U.S. Environmental Protection Agency.

The QUAL2K model (Q2K) described in this manual must be used at the user's own risk. Neither the U.S. Environmental Protection Agency, Tufts University, the Washington Dept. of Ecology, nor the program authors can assume responsibility for model operation, output, interpretation, or usage.

The creators of this program have used their best efforts in preparing this code. It is not absolutely guaranteed to be error free. The author/programmer makes no warrantees, expressed or implied, including without limitation warrantees of merchantability or fitness for any particular purpose. No liability is accepted in any event for any damages, including accidental or consequential damages, loss of profits, costs of lost data or programming materials, or otherwise in connection with or arising out of the use of this program.

PLEASE REFERENCE THIS DOCUMENT AS:

Chapra, S.C., Tao, H., Flynn, K.F., and Pelletier, G.J. 2022. QUAL2K: A Modeling Framework for Simulating River and Stream Water Quality, Version 3.01: Documentation. Civil and Environmental Engineering Dept., Tufts University, Medford, MA

IF YOU DETECT ERRORS IN THE **QUAL2K SOFTWARE** OR HAVE SUGGESTIONS TO IMPROVE IT, PLEASE SEND THEM TO: steven.chapra@tufts.edu

INTRODUCTION

QUAL2K (or Q2K) is a river and stream water quality model that is intended to represent a modernized version of the QUAL2E (or Q2E) model (Brown and Barnwell 1987). Q2K has the following characteristics and features:

- One dimensional. The channel is well-mixed vertically and laterally.
- Branching. The system can consist of a mainstem river with branched tributaries.
- Steady state hydraulics. Non-uniform, steady flow is simulated.
- Diel heat budget. The heat budget and temperature are simulated as a function of meteorology on a diel time scale.
- Diel water-quality. All water quality variables are simulated on a diel time scale.
- Heat and mass inputs. Point and non-point loads and withdrawals are simulated.
- Software Environment and Interface. Q2K is implemented within the Microsoft Windows environment. Numerical computations are programmed in Fortran 90. Excel is used as the graphical user interface. All interface operations are programmed in the Microsoft Office macro language: Visual Basic for Applications (VBA).
- Q2K divides the system into reaches and elements. The element size for Q2K can vary from reach to reach. In addition, multiple loadings and withdrawals can be input to any element.
- Carbonaceous BOD speciation. Q2K uses two forms of carbonaceous BOD to represent organic carbon. These forms are a slowly oxidizing form (slow CBOD) and a rapidly oxidizing form (fast CBOD).
- Anoxia. Q2K accommodates anoxia by reducing oxidation reactions to zero at low oxygen levels. In addition, denitrification is modeled as a first-order reaction that becomes pronounced at low oxygen concentrations.
- Sediment-water interactions. Sediment-water fluxes of dissolved oxygen and nutrients can be simulated internally rather than being prescribed. That is, oxygen (SOD) and nutrient fluxes are simulated as a function of settling particulate organic matter, reactions within the sediments, and the concentrations of soluble forms in the overlying waters.
- Algae. The model simulates 2 functional groups of both phytoplankton and attached bottom algae. Algal biomass is modeled as organic carbon and internal stores of nitrogen and phosphorus are modeled independently.
- Light extinction. Light extinction is calculated as a function of algae, POC and inorganic solids.
- pH. Both alkalinity and total inorganic carbon are simulated. The river's pH is then computed based on these two quantities.
- Pathogens. A generic pathogen is simulated. Pathogen removal is determined as a function of temperature, light, and settling.
- Reach specific kinetic parameters. Q2K allows you to specify many of the kinetic parameters on a reach-specific basis.
- Weirs and waterfalls. The hydraulics of weirs, as well as the effect of weirs and waterfalls on gas transfer, are explicitly included.

SEGMENTATION AND HYDRAULICS

The model represents a river as a series of reaches. These represent stretches of river that have constant hydraulic characteristics (e.g., slope, bottom width, etc.). As depicted in Figure 1, the reaches are numbered in ascending order starting from the headwater of the river's main stem. Notice that both point and non-point sources and point and non-point withdrawals (abstractions) can be positioned anywhere along the channel's length.

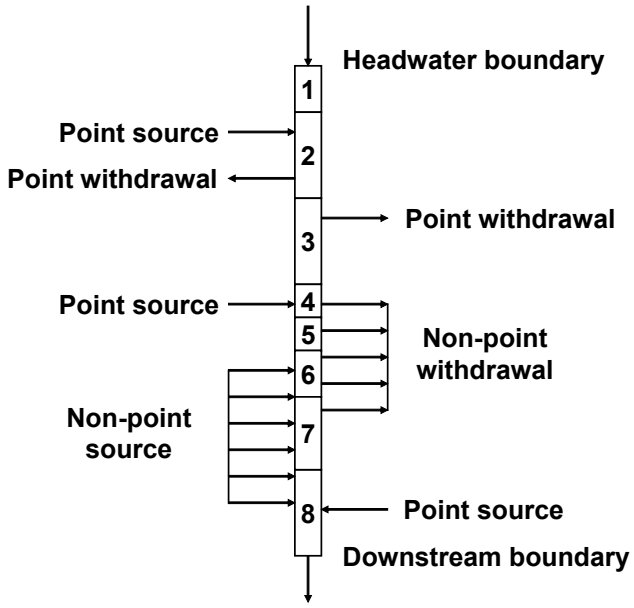

Figure 1 QUAL2K segmentation scheme for a river with no tributaries.

For systems with tributaries (Figure 2), the reaches are numbered in ascending order starting at reach 1 at the headwater of the main stem. When a junction with a tributary is reached, the numbering continues at that tributary's headwater. Observe that both the headwaters and the tributaries are also numbered consecutively following a sequencing scheme like the reaches. Note also that the major branches of the system (that is, the main stem and each of the tributaries) are referred to as segments. This distinction has practical importance because the software provides plots of model output on a segment basis. That is, the software generates individual plots for the main stem as well as each of the tributaries.

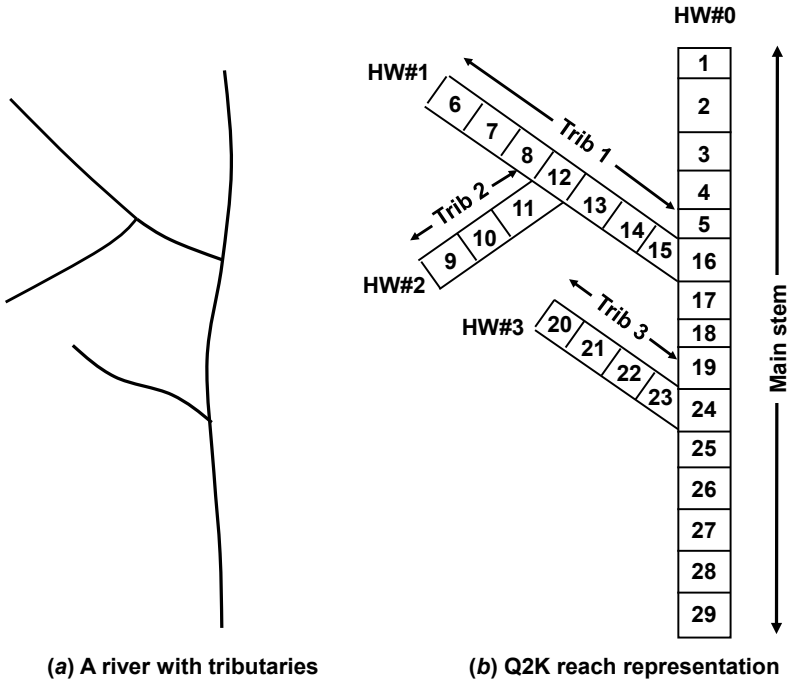

Figure 2 QUAL2K segmentation scheme for (a) a river with tributaries. The Q2K reach representation in (b) illustrates the reach, headwater, and tributary numbering schemes.

Finally, any model reach can be further divided into a series of equally-spaced elements. As in Figure 3, this is done by merely specifying the number of elements that are desired.

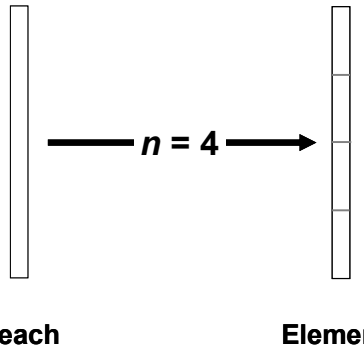

Figure 3 If desired, any model reach can be further subdivided into a series of *n* equal-length elements.

In summary, the nomenclature used to describe the way in which Q2K organizes river topology is as follows:

- *Reach.* A length of river with constant hydraulic characteristics.
- *Element.* The model's fundamental computational unit which consists of an equal length subdivision of a reach.
- *Segment.* A collection of reaches representing a branch of the system. These consist of the main stem as well as each tributary.

- *Headwater.* The upper boundary of a model segment.

Flow Balance

As described in the last section, Q2K's most fundamental unit is the element. A steady-state flow balance is implemented for each model element as (Figure 4)

$$Q_i = Q_{i-1} + Q_{in,i} - Q_{out,i} - Q_{evap,i} \quad (1)$$

where Q_i = outflow from element i into the downstream element $i + 1$ [m^3/d], Q_{i-1} = inflow from the upstream element $i - 1$ [m^3/d], $Q_{in,i}$ is the total inflow into the element from point and nonpoint sources [m^3/d], $Q_{out,i}$ is the total outflow from the element due to point and nonpoint withdrawals [m^3/d], and $Q_{evap,i}$ is the outflow due to evaporation [m^3/d]. Thus, the downstream outflow is simply the difference between inflow and source gains minus withdrawal and evaporative losses.

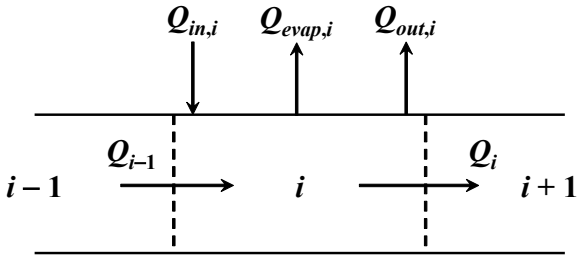

Figure 4 Element flow balance.

The total inflow from sources is computed as

$$Q_{in,i} = \sum_{j=1}^{psi} Q_{ps,i,j} + \sum_{j=1}^{npsi} Q_{nps,i,j} \quad (2)$$

where $Q_{ps,i,j}$ is the j^{th} point source inflow to element i [m^3/d], psi = the total number of point sources to element i, $Q_{nps,i,j}$ is the j^{th} non-point source inflow to element i [m^3/d], and $npsi$ = the total number of non-point source inflows to element i.

The total outflow from withdrawals is computed as

$$Q_{out,i} = \sum_{j=1}^{pai} Q_{pa,i,j} + \sum_{j=1}^{npai} Q_{npa,i,j} \quad (3)$$

where $Q_{pa,i,j}$ is the j^{th} point withdrawal outflow from element i [m^3/d], pai = the total number of point withdrawals from element i, $Q_{npa,i,j}$ is the j^{th} non-point withdrawal outflow from element i [m^3/d], and $npai$ = the total number of non-point withdrawal flows from element i.

The non-point sources and withdrawals are modeled as line sources. As in Figure 5, the non-point source or withdrawal is demarcated by its starting and ending kilometer points. Its flow is then distributed to or from each element in a length-weighted fashion.

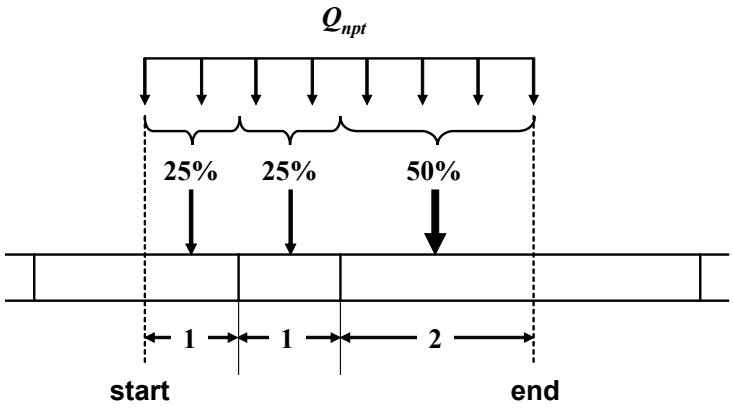

Figure 5 The way that non-point source flow is distributed to an element.

Evaporative Losses

The model internally computes the loss of water due to evaporation. This flow for each element is computed as

$$Q_{e,i} = \frac{J_{e,i} A_{s,i}}{\rho_w L_{e,i}} \times \frac{\text{m}}{100\ \text{cm}} \quad \textbf{(4)}$$

where $J_{e,i}$ = the heat flux due to evaporation (cal/cm^2/d), ρ_w = the density of water ($\cong$ 1 g/cm^3), and L_e = the latent heat of vaporization (cal/g). The latent heat is related to the element's temperature by

$$L_{e,i} = 597.3 - 0.57 T_i \quad \textbf{(5)}$$

where T_i = temperature (°C).

Hydraulic Characteristics

Once the outflow for each element is computed, the depth and velocity are calculated in one of three ways: weirs, rating curves, and Manning equations. The program decides among these options in the following manner:

- If weir height and width are entered, the weir option is implemented.
- If the weir height and width are zero and rating curve coefficients are entered (a and α), the rating curve option is implemented.
- If neither of the previous conditions is met, Q2K uses the Manning equation.

The following sections explain how depth and velocity are computed by each of the three options. This is followed by a short section describing how the other necessary hypergeometric parameters are computed.

The Trapezoidal Channel

In all cases, it is assumed that the river channel is idealized as a trapezoidal cross-section (Figure 6). Note that the default is for the channel to be rectangular (i.e., the side slopes, $s_{s1} = s_{s2} = 0$). By assigning the side slopes to values greater than zero, the channel becomes trapezoidal.

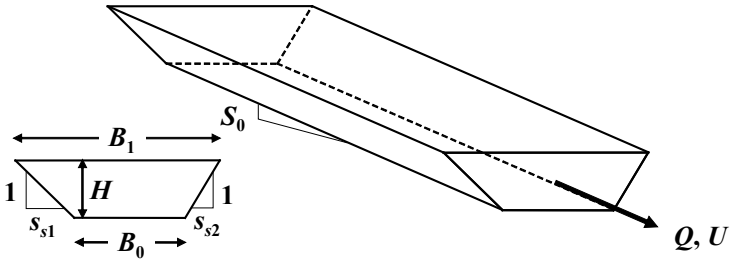

Figure 6 Trapezoidal channel.

Given B_0, s_{s1}, and s_{s2}, the top width (B_1, m) can be computed with

$$B_1 = B_0 + (s_{s1} + s_{s2})H \tag{6}$$

and the cross-sectional area (A_c, m^2) by

$$A_c = \left[B_0 + 0.5(s_{s1} + s_{s2})H\right]H \tag{7}$$

Weirs

Sharp-crested weirs

Figure 7 shows how weirs are generally represented in Q2K. **Note that a weir can only occur at the end of a reach consisting of a single element.** The width of the weir can also differ from the width of the element, B_i.

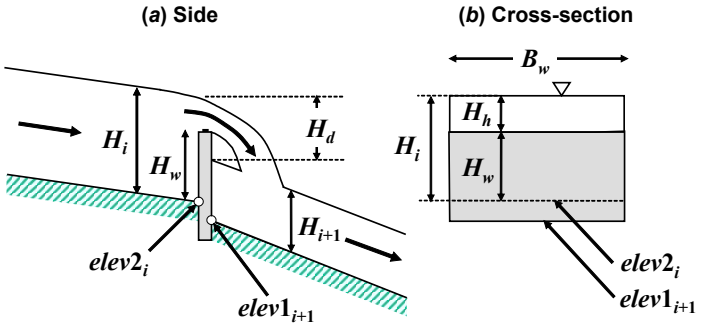

Figure 7 A sharp-crested weir occurring at the boundary between two reaches.

For the case where $H_h/H_w < 0.4$, flow is related to head by (Finnemore and Franzini 2002)

$$Q_i = 1.83 B_w H_h^{3/2} \tag{8}$$

where Q_i is the outflow from the element upstream of the weir (m^3/s), and B_w and H_h are in m. Equation (8) can be solved for

$$H_h = \left(\frac{Q_i}{1.83 B_w}\right)^{2/3} \tag{9}$$

This result can then be used to compute the depth of the element, i, upstream of the weir

$$H = H_w + H_h \tag{10}$$

and the drop over the weir by

$$H_{d,i} = elev2_i + H_i - elev1_{i+1} - H_{i+1} \quad (11)$$

As described in later sections, this drop is used to compute oxygen and carbon dioxide gas transfer due to the weir.

Broad-crested weirs

Using the same nomenclature as for the sharp-quested weir (Figure 7), the equations for the broad-crested weir are (Munson et al. 2009)

$$Q_i = C_w B_w \sqrt{g} \left(\frac{2}{3}\right)^{1.5} H_h^{1.5} \quad (12)$$

where C_w = weir coefficient (dimensionless), and g = the gravitational constant (m/s^2). C_w is determined using the weir height (H_w) as in

$$C_w = 1.125 \sqrt{\frac{1 + H_h / H_w}{2 + H_h / H_w}} \quad (13)$$

Substituting Eq. (11) into Eq. (10) and collecting terms gives

$$Q_i = 1.125 \sqrt{\frac{H_w + H_h}{2H_w + H_h}} B_w \sqrt{g} \left(\frac{2}{3}\right)^{1.5} H_h^{1.5} \quad (14)$$

Equation (14) cannot be solved explicitly for H_h. However, it can be solved with fixed-point iteration (also known as *successive substitution*) as described in Chapra (2022). This is done by rearranging it so that the last H_h is brought to the left-hand side,

$$H_h^j = \frac{1.5}{1.125^{2/3}} \left(\frac{Q_i}{B_w \sqrt{g}} \sqrt{\frac{2H_w + H_h^{j-1}}{H_w + H_h^{j-1}}} \right)^{2/3} \quad (15)$$

where H_h^j and H_h^{j-1} = the head above the top of the weir for the present and the previous iterations, respectively. By guessing an initial head (H_h^0) equal to zero, this equation can be solved iteratively to determine H_h.

For both types of weir, the cross-sectional area of the element above the weir can be computed with Eq. (7). Velocity can then be calculated with

$$U = \frac{Q}{A_c} \quad (16)$$

Rating Curves

Power equations (sometimes called *Leopold-Maddox relationships*) can be used to relate mean velocity and depth to flow for the elements in a reach,

$$U = aQ^b \quad (17)$$

$$H = \alpha Q^\beta \quad (18)$$

where a, b, α and β are empirical coefficients that are determined from velocity-discharge and stage-discharge rating curves, respectively.

The exponents b and β typically take on values listed in Table 1. Note that the sum of b and β must be less than or equal to 1. If this is not the case, the width will decrease with increasing flow. If their sum equals 1, the channel is rectangular and therefore, side slopes should not be specified.

Table 1 Typical values for the exponents of rating curves used to determine velocity and depth from flow (Barnwell et al. 1989).

Equation	Exponent	Typical value	Range
$U = aQ^b$	b	0.43	0.4–0.6
$H = \alpha Q^\beta$	β	0.45	0.3–0.5

In some applications, you might want to specify constant values of depth and velocity that do not vary with flow. This can be done by setting the exponents b and β to zero and setting a equal to the desired velocity and α equal to the desired depth.

Manning Equation

Under conditions of steady flow, the Manning equation expresses the relationship between flow and depth as

$$Q = \frac{S_0^{1/2}}{n}\frac{A_c^{5/3}}{P^{2/3}} \tag{19}$$

where Q = flow [m^3/s][1], S_0 = bottom slope [m/m], n = the Manning roughness coefficient, A_c = the cross-sectional area [m^2], and P = the wetted perimeter [m].

The cross-sectional area is computed with Eq. (7) and the wetted perimeter is computed as

$$P = B_0 + H\sqrt{s_{s1}^2 + 1} + H\sqrt{s_{s2}^2 + 1} \tag{20}$$

After substituting these equations into Eq. (20), the result is

$$Q = \frac{S_0^{1/2}}{n}\frac{\left(B_0 + H\sqrt{s_{s1}^2 + 1} + H\sqrt{s_{s2}^2 + 1}\right)^{5/3}}{\left(B_0 + H\sqrt{s_{s1}^2 + 1} + H\sqrt{s_{s2}^2 + 1}\right)^{2/3}} \tag{21}$$

which can be rearranged to give

$$H_k = \frac{(Qn)^{3/5}\left(B_0 + H_{k-1}\sqrt{s_{s1}^2 + 1} + H_{k-1}\sqrt{s_{s2}^2 + 1}\right)^{2/5}}{S^{3/10}\left[B_0 + 0.5(s_{s1} + s_{s2})H_{k-1}\right]} \tag{22}$$

This formula can be solved iteratively for depth with fixed-point iteration (Chapra 2022) where k = 1, 2, …, n, and n = the number of iterations. An initial guess of $H_0 = 0$ is employed. The

[1] Notice that time is measured in seconds in this and other formulas used to characterize hydraulics. This is how the computations are implemented within Q2K. However, once the hydraulic characteristics are determined they are converted to day units to be compatible with other model computations.

method is terminated when the estimated error falls below a specified value of 0.001%. The estimated error is calculated as

$$\varepsilon_a = \left| \frac{H_{k+1} - H_k}{H_{k+1}} \right| \times 100\% \quad (23)$$

The cross-sectional area is determined with Eq. (15) and the velocity can then be determined from the continuity equation,

$$U = \frac{Q}{A_c} \quad (24)$$

Suggested values for the Manning coefficient are listed in Table 2. Manning's *n* typically varies with flow and depth (Gordon et al. 1992). As the depth decreases at low flow, the relative roughness usually increases. Typical published values of Manning's n, which range from about 0.012 for smooth concrete channels to about 0.15 for rough natural channels, are representative of conditions when the flow is at the bankfull capacity (Rosgen, 1996). Critical conditions of depth for evaluating water quality under low flow conditions are generally much less than bankfull depth, and the relative roughness may be much higher. For example, in upland streams, rather than the type of bed material, the roughness is heavily influenced by the pool-riffle structure and can be very large (Beven et al. 1979).

Table 2 The Manning roughness coefficient for various open channel surfaces (from Chow et al. 1988).

Material	***n***
Man-made channels	
Concrete	0.012
Gravel bottom with sides:	
Concrete	0.020
mortared stone	0.023
Riprap	0.033
Natural stream channels	
Clean, straight	0.025-0.04
Clean, winding and some weeds	0.03-0.05
Weeds and pools, winding	0.05
Mountain streams with boulders	0.04-0.10
Heavy brush, timber	0.05-0.20
Steep upland channels	0.075-?

Other Hydrogeometric Parameters

Now that we have described how the three options are used to compute depth and velocity, the other hydrogeometric parameters required by Q2K can be computed. Top width and cross-sectional area can be computed with Eqs. (6) and (7), respectively. The surface area and volume of the element can then be computed as

$$A_s = B_1 \Delta x \quad (25)$$

$$V = A_c \Delta x \quad (26)$$

Waterfalls

The drop of water over a weir was computed on p. 9. This value is needed to compute the enhanced reaeration that occurs in such cases. In addition to weirs, such drops can also occur at waterfalls (Figure 8). **Note that waterfalls can only occur at the end of a reach.**

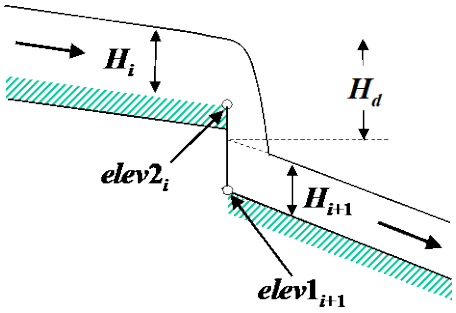

Figure 8 A waterfall occurring at the boundary between two reaches.

QUAL2K computes such drops for cases where the elevation above sea level drops abruptly at the boundary between two reaches. Equation (7) is used to compute the drop. It should be noted that the drop is only calculated when the elevation above sea level at the downstream end of a reach is greater than at the beginning of the next downstream reach; that is, $elev2_i > elev1_{i+1}$.

Travel Time

The residence time of each element is computed as

$$\tau_k = \frac{V_k}{Q_k} \tag{27}$$

where τ_k = the residence time of the k^{th} element [d]; V_k = the volume of the k^{th} element [m^3] = $A_{c,k} \cdot \Delta x_k$; $A_{c,k}$ = the cross-sectional area of the k^{th} element [m^2]; and Δx_k = the length of the k^{th} element [m]. These times are then accumulated to determine the travel time along each of the river's segments (that is, either the main stem or one of the tributaries). For example, the travel time from the headwater to the downstream end of the j^{th} element in a segment is computed as,

$$t_{t,j} = \sum_{k=1}^{j} \tau_k \tag{28}$$

where $t_{t,j}$ = the travel time [d].

Longitudinal Dispersion

Two options are used to determine the longitudinal dispersion for a boundary between two elements. First, the user can simply enter estimated values on the **Reach Worksheet**. If the user does not enter values, a formula is employed to internally compute dispersion based on the channel's hydraulics (Fischer et al. 1979),

$$E_{p,i} = 0.011 \frac{U_i^2 B_i^2}{H_i U_i^*} \tag{29}$$

where $E_{p,i}$ = the measured longitudinal dispersion between elements i and $i + 1$ [m^2/s], U_i = velocity [m/s], B_i = width [m], H_i = mean depth [m], and U_i^* = shear velocity [m/s], which is related to more fundamental characteristics by

$$U_i^* = \sqrt{gH_iS_i} \tag{30}$$

where g = acceleration due to gravity [= 9.81 m/s^2] and S = channel slope [dimensionless].

The numerical dispersion, $E_{n,i}$, is computed as

$$E_{n,i} \cong U\Delta x(\alpha - 0.5) \tag{31}$$

The model dispersion E_i (i.e., the value used in the model calculations) is then computed as follows:

if $E_n \le E_p$
  $E_m = E_p - E_n$ **(you're OK)**
else
  $E_m = 0$ **(and live with it, or reduce Δx until $E_n \le E_p$)**
end

For the latter case, the resulting dispersion will be greater than the physical dispersion. Thus, dispersive mixing will be higher than reality. It should be noted that for most steady-state rivers, the impact of this overestimation on concentration gradients will usually be negligible. If the discrepancy is significant, the only alternative is to make element lengths smaller so that the numerical dispersion becomes smaller than the physical dispersion.

TEMPERATURE MODEL

As in Figure 9, the heat balance for each water element includes heat transfers from adjacent elements, loads, withdrawals, the atmosphere, and the sediments. A heat balance can be written for element i as

$$\frac{dT_i}{dt} = \frac{Q_{i-1}}{V_i}T_{i-1,i} - \frac{Q_i}{V_i}T_{i,i+1} - \frac{Q_{out,i}}{V_i}T_i + \frac{E'_{i-1,i}}{V_i}(T_{i-1} - T_i) + \frac{E'_{i,i+1}}{V_i}(T_{i+1} - T_i) + \frac{W_{h,i}}{\rho_w C_{pw} V_i}\left(\frac{\text{m}^3}{10^6\ \text{cm}^3}\right) + \frac{J_{a,i}}{\rho_w C_{pw} H_i}\left(\frac{\text{m}}{100\ \text{cm}}\right) + \frac{J_{s,i}}{\rho_w C_{pw} H_i}\left(\frac{\text{m}}{100\ \text{cm}}\right) \tag{32}$$

where T_i = temperature in element i [ºC] , t = time [d], Q_i = outflow from element i into the downstream element, V_i = volume of element i [m^3], $T_{j,k}$ = temperature at the boundary between element j and k [ºC], $Q_{out,i}$ is the total outflow from the element due to point and nonpoint withdrawals [m^3/d], $E'_{j,k}$ = the bulk dispersion coefficient between elements j and k [m^3/d], $W_{h,i}$ = the net heat load from point and non-point sources into element i [cal/d], ρ_w = the density of water [g/cm^3], C_{pw} = the specific heat of water [cal/(g ºC)], $J_{a,i}$ = the air-water heat flux [cal/(cm^2 d)], and $J_{s,i}$ = the sediment-water heat flux [cal/(cm^2 d)].

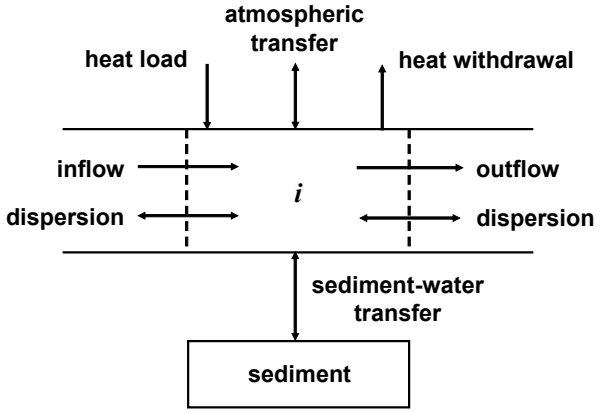

Figure 9 Heat balance for an element.

The net heat load from sources is computed as (recall Eq. 2)

$$W_{h,i} = \rho C_p \left[\sum_{j=1}^{psi} Q_{ps,i,j} T_{ps,i,j} + \sum_{j=1}^{npsi} Q_{nps,i,j} T_{nps,i,j} \right] \quad (33)$$

where $T_{ps,i,j}$ is the temperature of the j^{th} point source for element i [°C], and $T_{nps,i,j}$ is the temperature of the j^{th} non-point source temperature for element i [°C].

As depicted in Figure 10, the variable $T_{j,k}$ is the temperature at the interface between upstream element j and downstream element k, which is computed as a weighted average,

$$T_{j,k} = \alpha_{j,k} T_j + \beta_{j,k} T_k \quad (34)$$

where α and β are weighting coefficients that estimate the interface concentration with linear interpolation (Figure 14).

$$\alpha_{j,k} = \frac{\Delta x_k}{\Delta x_i + \Delta x_k} \quad \text{and} \quad \beta_{j,k} = 1 - \alpha_{j,k} = \frac{\Delta x_j}{\Delta x_j + \Delta x_k} \quad (35)$$

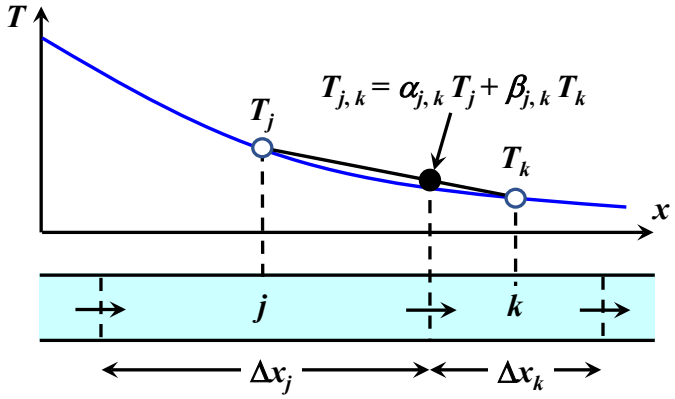

Figure 10 Illustration of how weighted averages based on linear interpolation are used to estimate the temperature at the interface between elements.

Setting $\alpha = 1$, which is called upstream or backward differencing, results in the interface concentration being the concentration of the upstream element. If this is done, the steady-state solution will be stable but will result in numerical dispersion. For equal length segments, $\alpha = 1/2$, which is called centered differencing, results in the interface concentration being the average of the two adjacent elements' concentrations. If this is done, the steady-state solution may be unstable but will not suffer from numerical dispersion.[2]

Bulk Dispersion

The bulk dispersion coefficient is computed as

$$E_i' = \frac{E_i A_{c,i}}{\left(\Delta x_i + \Delta x_{i+1}\right)/2} \tag{36}$$

Note that two types of boundary condition are used at the river's downstream terminus: (1) a zero-dispersion condition (natural boundary condition) and (2) a prescribed downstream boundary condition (Dirichlet boundary condition). The choice between these options is made on the **Downstream Worksheet**.

Surface Heat Flux

As depicted in Figure 11, surface heat exchange is modeled as a combination of five processes:

$$J_h = I(0) + J_{an} - J_{br} - J_c - J_e \tag{37}$$

where $I(0)$ = net solar shortwave radiation at the water surface, J_{an} = net atmospheric longwave radiation, J_{br} = longwave back radiation from the water, J_c = conduction, and J_e = evaporation. All fluxes are expressed as cal/cm^2/d.

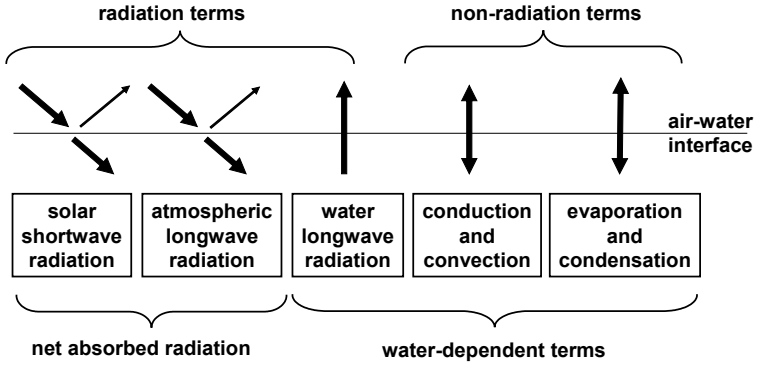

Figure 11 The components of surface heat exchange.

Solar Radiation

The model computes the amount of solar radiation entering the water at a particular latitude (L_{at}) and longitude (L_{on}) on the earth's surface. This quantity is a function of the radiation at the top of the earth's atmosphere which is attenuated by atmospheric transmission, cloud cover, reflection, and shade,

[2] For additional details on numerical dispersion and solution positivity, see Chapra (1997) as well as a later section starting on page 27)

$$I(0) = I_0 \; a_t \; a_c \; (1-R_s) \; (1-S_f) \tag{38}$$

extraterrestrial radiation; atmospheric attenuation; cloud attenuation; reflection; shading

where $I(0)$ = solar radiation at the water surface [cal/cm^2/d], I_0 = extraterrestrial radiation (i.e., at the top of the earth's atmosphere) [cal/cm^2/d], a_t = atmospheric attenuation, a_c = cloud attenuation, R_s = albedo (fraction reflected), and S_f = effective shade (fraction blocked by vegetation and topography).

Extraterrestrial radiation. The extraterrestrial radiation is computed as (TVA 1972)

$$I_0 = \frac{W_0}{r^2}\sin\alpha \tag{39}$$

where W_0 = the solar constant [1367 W/m^2 or 2823 cal/cm^2/d], r = normalized radius of the earth's orbit (i.e., the ratio of actual earth-sun distance to mean earth-sun distance), and α = the sun's altitude [radians], which can be computed as

$$\sin\alpha = \sin\delta\sin L_{at} + \cos\delta\cos L_{at}\cos(\tau) \tag{40}$$

where δ = solar declination [radians], Lat = local latitude [radians], and τ = the local hour angle of the sun [radians].

The local hour angle in radians is given by

$$\tau = \left(\frac{trueSolarTime}{4} - 180\right) \times \frac{\pi}{180} \tag{41}$$

where:

$$trueSolarTime = localTime + eqtime - 4 \times L_{on} - 60 \times timezone \tag{42}$$

where *trueSolarTime* is the solar time determined from the actual position of the sun in the sky [minutes], *localTime* is the local time in minutes (local standard time), L_{on} is the local longitude (positive decimal degrees for the western hemisphere), and *timezone* is the local time zone in hours relative to Greenwich Mean Time (e.g., –8 hours for Pacific Standard Time; the local time zone is selected on the **QUAL2K Worksheet**). The value of *eqtime* represents the difference between true solar time and mean solar time in minutes of time.

QUAL2K calculates the solar declination, hour angle, solar altitude, and normalized radius (distance between the earth and sun), as well as the times of sunrise and sunset using the Meeus (1999) algorithms as implemented by NOAA's Surface Radiation Research Branch (www.srrb.noaa.gov/highlights/sunrise/azel.html). The NOAA method for solar position that is used in QUAL2K also includes a correction for the effect of atmospheric refraction. The complete calculation method that is used to determine the solar position, sunrise, and sunset is presented in Appendix B.

The photoperiod f [hours] is computed as

$$f = t_{ss} - t_{sr} \tag{43}$$

where t_{ss} = time of sunset [hours] and t_{sr} = time of sunrise [hours].

Atmospheric attenuation. Various methods have been published to estimate the fraction of the atmospheric attenuation from a clear sky (*at*). Two alternative methods are available in QUAL2K

to estimate a_t (Note that the solar radiation model is selected on the **Light and Heat Worksheet** of QUAL2K):

1) Bras (default)

The Bras (1990) method computes a_t as:

$$a_t = e^{-n_{fac} a_1 m} \tag{44}$$

where n_{fac} is an atmospheric turbidity factor that varies from approximately 2 for clear skies to 4 or 5 for smoggy urban areas. The molecular scattering coefficient (a_1) is calculated as

$$a_1 = 0.128 - 0.054 \log_{10} m \tag{45}$$

where m is the optical air mass, calculated as

$$m = \frac{1}{\sin\alpha + 0.15(\alpha_d + 3.885)^{-1.253}} \tag{46}$$

where α_d is the sun's altitude in degrees from the horizon = $\alpha \times (180^o/\pi)$.

2) Ryan and Stolzenbach

The Ryan and Stolzenbach (1972) model computes a_t from ground surface elevation and solar altitude as:

$$a_t = a_{tc}^{m\left(\frac{288-0.0065 elev}{288}\right)^{5.256}} \tag{47}$$

where a_{tc} is the atmospheric transmission coefficient (0.70-0.91, typically approximately 0.8), and *elev* is the surface elevation above sea level (m).

Direct measurements of solar radiation are available at some locations. For example, NOAA's Integrated Surface Irradiance Study (ISIS) has data from various stations across the United States (http://www.atdd.noaa.gov/isis.htm). The selection of either the Bras or Ryan-Stolzenbach solar radiation model and the appropriate atmospheric turbidity factor or atmospheric transmission coefficient for a particular application should ideally be guided by a comparison of predicted solar radiation with measured values at a reference location.

Cloud Attenuation. Attenuation of solar radiation due to cloud cover is computed with

$$a_c = 1 - 0.65 C_L^2 \tag{48}$$

where C_L = fraction of the sky covered with clouds.

Reflectivity. Reflectivity is calculated as

$$R_s = A \alpha_d^{\ B} \tag{49}$$

where A and B are coefficients related to cloud cover (Table 3).

Table 3 Coefficients used to calculate reflectivity based on cloud cover.

Cloudiness	Clear		Scattered		Broken		Overcast	
C_L	0		0.1-0.5		0.5-0.9		1.0	
Coefficients	*A*	*B*	*A*	*B*	*A*	*B*	*A*	*B*
	1.18	–0.77	2.20	–0.97	0.95	–0.75	0.35	–0.45

Shade. Shade is an input variable for the QUAL2K model. Shade is defined as the fraction of potential solar radiation that is blocked by topography and vegetation. An Excel/VBA program named 'Shade.xls' is available from the Washington Department of Ecology to estimate shade from topography and riparian vegetation (Ecology 2003). Input values of integrated hourly estimates of shade for each reach are entered on the **Shade Worksheet** of QUAL2K.

Atmospheric Long-wave Radiation

The downward flux of longwave radiation from the atmosphere is one of the largest terms in the surface heat balance. This flux can be calculated using the Stefan-Boltzmann law

$$J_{an} = \sigma\left(T_{air} + 273\right)^4 \quad \varepsilon_{sky} \quad \left(1 - R_L\right) \tag{50}$$

where σ = the Stefan-Boltzmann constant = 11.7×10^{-8} cal/(cm^2 d K^4), T_{air} = air temperature [°C], ε_{sky} = effective emissivity of the atmosphere [dimensionless], and R_L = longwave reflection coefficient [dimensionless]. Emissivity is the ratio of the longwave radiation from an object compared with the radiation from a perfect emitter at the same temperature. The reflection coefficient is generally small and is assumed to equal 0.03.

The atmospheric longwave radiation model is selected on the **Light and Heat Worksheet** of QUAL2K. Three alternative methods are available for use in QUAL2K to represent the effective emissivity (ε_{sky}):

1) Brunt (default)

Brunt's (1932) equation is an empirical model that has been commonly used in water-quality models (Thomann and Mueller 1987),

$$\varepsilon_{clear} = A_a + A_b\sqrt{e_{air}}$$

where A_a and A_b are empirical coefficients. Values of A_a have been reported to range from about 0.5 to 0.7 and values of A_b have been reported to range from about 0.031 to 0.076 mmHg–0.5 for a wide range of atmospheric conditions. QUAL2K uses a default mid-range value of A_a = 0.6 together with a value of A_b = 0.031 mmHg-0.5 if the Brunt method is selected on the **Light and Heat Worksheet**.

2) Brutsaert

The Brutsaert equation is physically based instead of empirically derived and has been shown to yield satisfactory results over a wide range of atmospheric conditions of air temperature and humidity at intermediate latitudes for conditions above freezing (Brutsaert, 1982).

$$\varepsilon_{clear} = 1.24\left(\frac{1.333224e_{air}}{T_a}\right)^{1/7}$$

where e_{air} is the air vapor pressure [mm Hg], and T_a is the air temperature in °K. The factor of 1.333224 converts the vapor pressure from mm Hg to millibars. The air vapor pressure [in mm Hg] is computed as (Raudkivi 1979):

$$e_{air} = 4.596e^{\frac{17.27T_d}{237.3+T_d}} \tag{51}$$

where T_d = the dew-point temperature [°C].

3) Koberg

Koberg (1964) reported that the A_a in Brunt's formula depends on both air temperature and the ratio of the incident solar radiation to the clear-sky radiation (R_{sc}). As in Figure 12, he presented a series of curves indicating that A_a increases with T_{air} and decreases with R_{sc} with A_b held constant at 0.0263 millibars$^{-0.5}$ (about 0.031 mmHg$^{-0.5}$).

The following polynomial is used in Q2K to provide a continuous approximation of Koberg's curves.

$$A_a = a_k T_{air}^2 + b_k T_{air} + c_k$$

where

$$a_k = -0.00076437 R_{sc}^3 + 0.00121134 R_{sc}^2 - 0.00073087 R_{sc} + 0.0001106$$

$$b_k = 0.12796842\, R_{sc}^3 - 0.2204455 R_{sc}^2 + 0.13397992\, R_{sc} - 0.02586655$$

$$c_k = -3.25272249 R_{sc}^3 + 5.65909609 R_{sc}^2 - 3.43402413 R_{sc} + 1.43052757$$

The fit of this polynomial to points sampled from Koberg's curves are depicted in Figure 12. Note that an upper limit of 0.735 is prescribed for A_a.

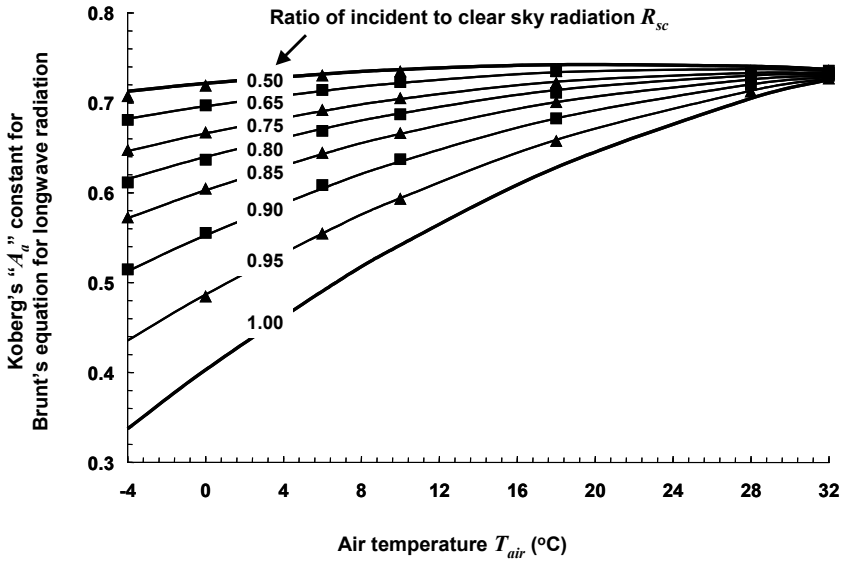

Figure 12 The points are sampled from Koberg's family of curves for determining the value of the A_a constant in Brunt's equation for atmospheric longwave radiation (Koberg, 1964). The lines are the functional representation used in Q2K.

For cloudy conditions the atmospheric emissivity may increase because of increased water vapor content. High cirrus clouds may have a negligible effect on atmospheric emissivity, but lower stratus and cumulus clouds may have a significant effect. The Koberg method accounts for the effect of clouds on the emissivity of longwave radiation in the determination of the *Aa* coefficient. The Brunt and Brutsaert methods determine emissivity of a clear sky and do not account for the effect of clouds. Therefore, if the Brunt or Brutsaert methods are selected, then the effective atmospheric emissivity for cloudy skies (ε_{sky}) is estimated from the clear sky emissivity by using a nonlinear function of the fractional cloud cover (C_L) of the form (TVA, 1972):

$$\varepsilon_{sky} = \varepsilon_{clear}(1 + 0.17C_L^2) \quad (52)$$

The selection of the longwave model for a particular application should ideally be guided by a comparison of predicted results with measured values at a reference location. However, direct measurements are rarely collected. The Brutsaert method is recommended to represent a wide range of atmospheric conditions.

Water Long-wave Radiation

The back radiation from the water surface is represented by the Stefan-Boltzmann law,

$$J_{br} = \varepsilon\sigma\left(T + 273\right)^4 \quad (53)$$

where ε = emissivity of water (= 0.97) and T = the water temperature [°C].

Conduction and Convection

Conduction is the transfer of heat from molecule to molecule when masses of different temperatures are brought into contact. *Convection* is heat transfer that occurs due to mass movement of fluids. Both can occur at the air-water interface and can be described by,

$$J_c = c_1 f\left(U_w\right)\ \left(T_s - T_{air}\right) \quad (54)$$

where c_1 = Bowen's coefficient (= 0.47 mmHg/°C). The term, $f(U_w)$, defines the dependence of the transfer on wind velocity over the water surface where U_w is the wind speed measured a fixed distance above the water surface.

Many relationships exist to define the wind dependence. Bras (1990), Edinger et al. (1974), and Shanahan (1984) provide reviews of various methods. Some researchers have proposed that conduction/convection and evaporation are negligible in the absence of wind (e.g., Marciano and Harbeck, 1952), which is consistent with the assumption that only molecular processes contribute to the transfer of mass and heat without wind (Edinger et al. 1974). Others have shown that significant conduction/convection and evaporation can occur in the absence of wind (e.g., Brady Graves and Geyer 1969, Harbeck 1962, Ryan and Harleman 1971, Helfrich et al. 1982, and Adams et al. 1987). This latter opinion has gained favor (Edinger et al. 1974), especially for waterbodies that exhibit water temperatures that are greater than the air temperature.

Brady, Graves, and Geyer (1969) pointed out that if the water surface temperature is warmer than the air temperature, then "the air adjacent to the water surface would tend to become both warmer and moister than that above it, thereby (due to both of these factors) becoming less dense. The resulting vertical convective air currents … might be expected to achieve much higher rates of heat and mass transfer from the water surface [even in the absence of wind] than would be possible by molecular diffusion alone" (Edinger et al. 1974). Water temperatures in natural waterbodies are frequently greater than the air temperature, especially at night.

Edinger et al. (1974) recommend that the relationship that was proposed by Brady, Graves and Geyer (1969) based on data from cooling ponds, could be representative of most environmental conditions. Shanahan (1984) recommends that the Lake Hefner equation (Marciano and Harbeck, 1952) is appropriate for natural waters in which the water temperature is less than the air temperature. Shanahan also recommends that the Ryan and Harleman (1971) equation as recalibrated by Helfrich et al. (1982) is best suited for waterbodies that experience water temperatures that are greater than the air temperature. Adams et al. (1987) revisited the Ryan and Harleman and Helfrich et al. models and proposed another recalibration using

additional data for waterbodies that exhibit water temperatures that are greater than the air temperature.

Three options are available on the **Light and Heat Worksheet** in QUAL2K to calculate $f(U_w)$:

1) Brady, Graves, and Geyer (default)

$$f(U_w) = 19.0 + 0.95U_w^2$$

where U_w = wind speed at a height of 7 m [m/s].

2) Adams 1

Adams et al. (1987) updated the work of Ryan and Harleman (1971) and Helfrich et al. (1982) to derive an empirical model of the wind speed function for heated waters that accounts for the enhancement of convection currents when the virtual temperature difference between the water and air ($\Delta\theta_v$ in degrees F) is greater than zero. Two wind functions reported by Adams et al., also known as the East Mesa method, are implemented in QUAL2K (wind speed in these equations is at a height of 2m).

This formulation uses an empirical function to estimate the effect of convection currents caused by virtual temperature differences between water and air, and the Harbeck (1962) equation is used to represent the contribution to conduction/convection and evaporation that is not due to convection currents caused by high virtual water temperature.

$$f(U_w) = 0.271\sqrt{(22.4\Delta\theta_v^{1/3})^2 + (24.2A_{acres,i}^{-0.05}U_{w,mph})^2}$$

where $U_{w,mph}$ is wind speed in mph and $A_{acres,i}$ is surface area of element i in acres. The constant 0.271 converts the original units of BTU ft^{-2} day^{-1} mmHg^{-1} to cal cm^{-2} day^{-1} mmHg^{-1}.

3) Adams 2

This formulation uses an empirical function of virtual temperature differences with the Marciano and Harbeck (1952) equation for the contribution to conduction/convection and evaporation that is not due to the high virtual water temperature

$$f(U_w) = 0.271\sqrt{(22.4\Delta\theta_v^{1/3})^2 + (17U_{w,mph})^2}$$

Virtual temperature is defined as the temperature of dry air that has the same density as air under the *in situ* conditions of humidity. The virtual temperature difference between the water and air ($\Delta\theta_v$ in °F) accounts for the buoyancy of the moist air above a heated water surface. The virtual temperature difference is estimated from water temperature ($T_{w,f}$ in °F), air temperature ($T_{air,f}$ in °F), vapor pressure of water and air (e_s and e_{air} in mmHg), and the atmospheric pressure (p_{atm} is estimated as standard atmospheric pressure of 760 mmHg in QUAL2K):

$$\Delta\theta_v = \left(\frac{T_{w,f} + 460}{1 + 0.378e_s / p_{atm}} - 460\right) - \left(\frac{T_{air,f} + 460}{1 + 0.378e_{air} / p_{atm}} - 460\right) \quad \textbf{(55)}$$

The height of wind speed measurements is also an important consideration for estimating conduction/convection and evaporation. QUAL2K internally adjusts the wind speed to the correct height for the wind function that is selected on the **Light and Heat Worksheet**. The input values for wind speed on the **Wind Speed Worksheet** in QUAL2K are assumed to be representative of conditions at a height of 7 meters above the water surface. To convert wind speed measurements

($U_{w,z}$ in m/s) taken at any height (z_w in meters) to the equivalent conditions at a height of z = 7 m for input to the **Wind Speed Worksheet** of QUAL2K, the exponential wind law equation may be used (TVA, 1972):

$$U_w = U_{wz}\left(\frac{z}{z_w}\right)^{0.15} \tag{56}$$

For example, if wind speed data were collected from a height of 2 m, then the wind speed at 7 m for input to the **Wind Speed Worksheet** of QUAL2K would be estimated by multiplying the measured wind speed by a factor of 1.2.

Evaporation and Condensation

The heat loss due to evaporation can be represented by Dalton's law,

$$J_e = f(U_w)(e_s - e_{air}) \tag{57}$$

where e_s = the saturation vapor pressure at the water surface [mmHg], and e_{air} = the air vapor pressure [mmHg]. The saturation vapor pressure is computed as

$$e_{air} = 4.596e^{\frac{17.27T}{237.3+T}} \tag{58}$$

Sediment-Water Heat Transfer

Recall (Figure 9) that the heat balance includes heat transfers from adjacent elements, loads, withdrawals, the atmosphere, and the sediments, which are included in the element heat balance,

$$\frac{dT_i}{dt} = \frac{Q_{i-1}}{V_i}T_{i-1,i} - \frac{Q_i}{V_i}T_{i,i+1} - \frac{Q_{out,i}}{V_i}T_i + \frac{E'_{i-1}}{V_i}\left(T_{i-1} - T_i\right) + \frac{E'_i}{V_i}\left(T_{i+1} - T_i\right) + \frac{W_{h,i}}{\rho_w C_{pw} V_i}\left(\frac{\text{m}^3}{10^6\ \text{cm}^3}\right) + \frac{J_{a,i}}{\rho_w C_{pw} H_i}\left(\frac{\text{m}}{100\ \text{cm}}\right) + \frac{J_{s,i}}{\rho_w C_{pw} H_i}\left(\frac{\text{m}}{100\ \text{cm}}\right) \tag{59}$$

A heat balance for bottom sediment underlying a water element i can be written as

$$\frac{dT_{s,i}}{dt} = -\frac{J_{s,i}}{\rho_s C_{ps} H_{sed,i}} \tag{60}$$

where $T_{s,i}$ = the temperature of the bottom sediment below element i [°C], $J_{s,i}$ = the sediment-water heat flux [cal/(cm^2 d)], ρ_s = the density of the sediments [g/cm^3], C_{ps} = the specific heat of the sediments [cal/(g °C)], and $H_{sed,i}$ = the effective thickness of the sediment layer [cm].

The flux from the sediments to the water can be computed as

$$J_{s,i} = \rho_s C_{ps}\frac{\alpha_s}{H_{sed,i}/2}\left(T_{si} - T_i\right) \times \frac{86{,}400\ \text{s}}{\text{d}} \tag{61}$$

where α_s = the sediment thermal diffusivity [cm^2/s]. Therefore,

$$\frac{dT_{s,i}}{dt} = -\frac{\alpha_s}{H_{sed,i}^2/2}\left(T_{si} - T_i\right) \times \frac{86{,}400\ \text{s}}{\text{d}} \tag{62}$$

substituting Eq. (62) into (59) gives

$$\frac{dT_i}{dt} = \frac{Q_{i-1}}{V_i}T_{i-1} - \frac{Q_i}{V_i}T_i - \frac{Q_{out,i}}{V_i}T_i + \frac{E'_{i-1}}{V_i}\left(T_{i-1} - T_i\right) + \frac{E'_i}{V_i}\left(T_{i+1} - T_i\right) + \frac{W_{h,i}}{\rho_w C_{pw} V_i}\left(\frac{\text{m}^3}{10^6\ \text{cm}^3}\right) + \frac{J_{a,i}}{\rho_w C_{pw} H_i}\left(\frac{\text{m}}{100\ \text{cm}}\right) + \alpha_s \frac{\rho_s C_{ps}}{\rho_w C_{pw}} \frac{2}{H_{sed,i} H_i}\left(T_{si} - T_i\right) \times \frac{86{,}400\ \text{s}}{\text{d}}\left(\frac{\text{m}}{100\ \text{cm}}\right) \quad \textbf{(63)}$$

The thermal properties of some natural sediments along with its components are summarized in Table 4. Note that soft, gelatinous sediments found in the deposition zones of lakes are very porous and approach the values for water. Some very slow, impounded rivers may approach such a state. However, many smaller rivers tend to have coarser sediments with significant fractions of sands, gravels and stones. Upland streams can have bottoms that are dominated by boulders and rock substrates.

Table 4 Thermal properties for natural sediments and the materials that comprise natural sediments.

Table 4. Thermal properties of various materials

Type of material	thermal conductivity		thermal diffusivity		ρ	C_p	ρCp	reference
	w/m/°C	cal/s/cm/°C	m^2/s	cm^2/s	g/cm3	cal/(g °C)	cal/(cm^3 °C)	
Sediment samples								
Mud Flat	1.82	0.0044	4.80E-07	0.0048			0.906	(1)
Sand	2.50	0.0060	7.90E-07	0.0079			0.757	"
Mud Sand	1.80	0.0043	5.10E-07	0.0051			0.844	"
Mud	1.70	0.0041	4.50E-07	0.0045			0.903	"
Wet Sand	1.67	0.0040	7.00E-07	0.0070			0.570	(2)
Sand 23% saturation with water	1.82	0.0044	1.26E-06	0.0126			0.345	(3)
Wet Peat	0.36	0.0009	1.20E-07	0.0012			0.717	(2)
Rock	1.76	0.0042	1.18E-06	0.0118			0.357	(4)
Loam 75% saturation with water	1.78	0.0043	6.00E-07	0.0060			0.709	(3)
Lake, gelatinous sediments	0.46	0.0011	2.00E-07	0.0020			0.550	(5)
Concrete canal	1.55	0.0037	8.00E-07	0.0080	2.200	0.210	0.460	"
Average of sediment samples:	*1.57*	*0.0037*	*6.45E-07*	*0.0064*			*0.647*	
Miscellaneous measurements:								
Lake, shoreline	0.59	0.0014						(5)
Lake soft sediments			3.25E-07	0.0033				"
Lake, with sand			4.00E-07	0.0040				"
River, sand bed			7.70E-07	0.0077				"
Component materials:								
Water	0.59	0.0014	1.40E-07	0.0014	1.000	0.999	1.000	(6)
Clay	1.30	0.0031	9.80E-07	0.0098	1.490	0.210	0.310	"
Soil, dry	1.09	0.0026	3.70E-07	0.0037	1.500	0.465	0.700	"
Sand	0.59	0.0014	4.70E-07	0.0047	1.520	0.190	0.290	"
Soil, wet	1.80	0.0043	4.50E-07	0.0045	1.810	0.525	0.950	"
Granite	2.89	0.0069	1.27E-06	0.0127	2.700	0.202	0.540	"
Average of composite materials:	*1.37*	*0.0033*	*6.13E-07*	*0.0061*	*1.670*	*0.432*	*0.632*	

(1) Andrews and Rodvey (1980)
(2) Geiger (1965)
(3) Nakshabandi and Kohnke (1965)
(4) Chow (1964) and Carslaw and Jaeger (1959)
(5) Hutchinson 1957, Jobson 1977, and Likens and Johnson 1969
(6) Cengel, Grigull, Mills, Bejan, Kreith and Bohn

Inspection of the component properties of Table 4 suggests that the presence of solid material in stream sediments leads to a higher coefficient of thermal diffusivity than that for water or porous lake sediments. In Q2K, we suggest a default value of 0.005 cm^2/s for this quantity. Of course, if you have a direct measurement of thermal diffusivity, you should use it.

In addition, specific heat tends to decrease with density. Thus, the product of these two quantities tends to be more constant than the multiplicands. Nevertheless, it appears that the presence of solid material in stream sediments leads to a lower product than that for water or gelatinous lake sediments. In Q2K, we suggest default values of ρ_s = 1.6 g/cm^3 and C_{ps} = 0.4 cal/(g ºC). This corresponds to a product of 0.64 cal/(cm^3 ºC) for this quantity. Finally, as derived

in Appendix C, the sediment thickness is set by default to 10 cm to capture the effect of the sediments on the diel heat budget for the water.

CONSTITUENT MODEL

Constituents and General Mass Balance

The model constituents are listed in Table 5.

Table 5 Model state variables. New state variables added for Q2K 3 are highlighted in yellow

i	*Variable*	*Symbol*	*Abbreviation*	*Units*
1	Conductivity	s		µmhos
2	Inorganic suspended solids	m_i	ISS	mgD/L
3	Dissolved oxygen	o	DO	mgO_2/L
4	Particulate organic carbon	c_{poc}	POC	mgC/L
5	Slowly reacting DOC	c_s	DOCs	mgC/L
6	Fast reacting DOC	c_f	DOCf	mgC/L
7	Particulate organic nitrogen	n_{pon}	PON	µgN/L
8	Dissolved organic nitrogen	n_{don}	DON	µgN/L
9	Ammonia nitrogen	n_a	NH4	µgN/L
10	Nitrite/Nitrate nitrogen	n_n	NO3	µgN/L
11	Particulate organic phosphorus	p_{pop}	PON	µgP/L
12	Dissolved organic phosphorus	p_{dop}	DOP	µgP/L
13	Inorganic phosphorus	p_i	Inorg P	µgP/L
14	Phytoplankton 1 biomass	c_{p1}		mgC/L
15	Phytoplankton 1 nitrogen	IN_{p1}		µgN/L
16	Phytoplankton 1 phosphorus	IP_{p1}		µgP/L
17	Phytoplankton 2 biomass	c_{p2}		mgC/L
18	Phytoplankton 2 nitrogen	IN_{p2}		µgN/L
19	Phytoplankton 2 phosphorus	IP_{p2}		µgP/L
20	Bottom algae 1 biomass	c_{b1}		mgC/m^2
21	Bottom algae 1 nitrogen	IN_{b1}		mgN/m^2
22	Bottom algae 1 phosphorus	IP_{b1}		mgP/m^2
23	Bottom algae 2 biomass	c_{b2}		mgC/m^2
24	Bottom algae 2 nitrogen	IN_{b2}		mgN/m^2
25	Bottom algae 2 phosphorus	IP_{b2}		mgP/m^2
26	Alkalinity	Alk		mg$CaCO_3$/L
27	Total inorganic carbon	c_T		mole/L
28	Salinity	S		ppt
29	Constituent i			
30	Constituent ii			
31	Constituent iii			
32	Pathogen	X		cfu/100 mL

* mg/L ≡ g/m^3; In addition, the terms D, C, N, and P, refer to dry weight, carbon, nitrogen, and phosphorus, respectively. The term cfu stands for colony forming unit which is a measure of viable bacterial numbers.

For all but the bottom algae variables, a general mass balance for a constituent in an element is written as

$$\frac{dc_i}{dt} = \frac{Q_{i-1}}{V_i}c_{i-1,i} - \frac{Q_i}{V_i}c_{i,i+1} - \frac{Q_{out,i}}{V_i}c_i + \frac{E'_{i-1}}{V_i}\left(c_{i-1} - c_i\right) + \frac{E'_i}{V_i}\left(c_{i+1} - c_i\right) + \frac{W_i}{V_i} + S_i \quad \textbf{(64)}$$

where c_i = temperature of element i [°C] , t = time [d], Q_i = outflow from element i into the downstream element, V_i = volume of element i [m^3], $c_{j,k}$ = temperature at the boundary between element j and k [°C], $Q_{out,i}$ is the total outflow from the element due to point and nonpoint withdrawals [m^3/d], $E'_{j,k}$ = the bulk dispersion coefficient between elements j and k [m^3/d], W_i = the external loading of the constituent to element i [g/d or mg/d], and S_i = sources and sinks of the constituent due to reactions and mass transfer mechanisms [g/m^3/d or mg/m^3/d].

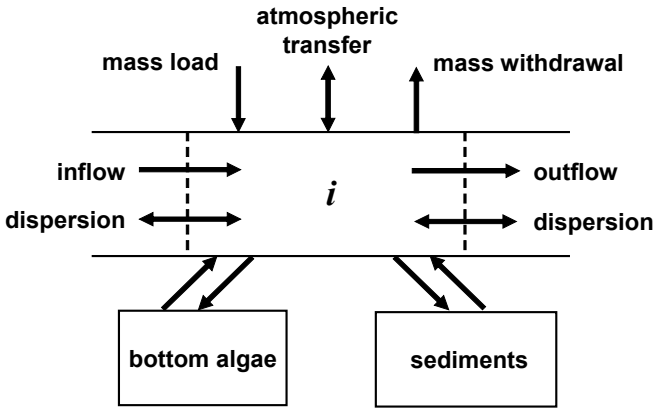

Figure 13 Mass balance for a water element.

The external load is computed as (recall Eq. 2),

$$W_i = \sum_{j=1}^{psi} Q_{ps,i,j}c_{psi,j} + \sum_{j=1}^{npsi} Q_{nps,i,j}c_{npsi,j} \quad \textbf{(65)}$$

where $c_{ps,i,j}$ is the j^{th} point source concentration for element i [mg/L or µg/L], and $c_{nps,i,j}$ is the j^{th} non-point source concentration for element i [mg/L or µg/L].

As depicted in Figure 14, the variable $c_{j,k}$ is the concentration at the interface between upstream element j and downstream element k, which is computed as a weighted average,

$$c_{j,k} = \alpha_{j,k}c_j + \beta_{j,k}c_k \quad \textbf{(66)}$$

where α and β are weighting coefficients that estimate the interface concentration with linear interpolation

$$\alpha_{j,k} = \frac{\Delta x_k}{\Delta x_i + \Delta x_k} \quad \text{and} \quad \beta_{j,k} = 1 - \alpha_{j,k} = \frac{\Delta x_j}{\Delta x_j + \Delta x_k} \quad \textbf{(67)}$$

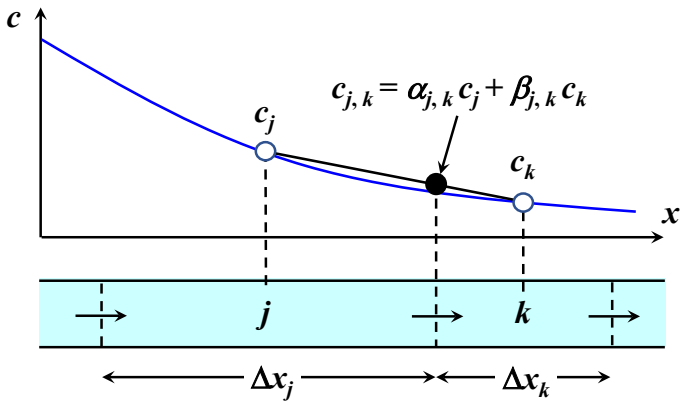

Figure 14 Illustration of how weighted averages based on linear interpolation are used to estimate the concentration at the interface between elements.

$$c_{i-1,i} = \alpha_{i-1,i} c_{i-1} + \beta_{i-1,1} c_i \tag{68}$$

Substituting Eq. (66) with the appropriate subscripts into Eq. (64) gives

$$\frac{dc_i}{dt} = \frac{Q_{i-1}}{V_i}\left(\alpha_{i-1,i} c_{i-1} + \beta_{i-1,1} c_i\right) - \frac{Q_i}{V_i} c_{i,i+1}\left(\alpha_{i,i+1} c_i + \beta_{i,i+1} c_{i+1}\right) - \frac{Q_{out,i}}{V_i} c_i + \frac{E'_{i-1}}{V_i}\left(c_{i-1} - c_i\right) + \frac{E'_i}{V_i}\left(c_{i+1} - c_i\right) + \frac{W_i}{V_i} + S_i \tag{69}$$

Omitting withdrawals and setting $\alpha = 1$, which is called *upstream* or *backward differencing*, results in the interface concentration being the concentration of the upstream element. For this case, the equation becomes

$$\frac{dc_i}{dt} = \frac{Q_{i-1}}{V_i} c_{i-1} - \frac{Q_i}{V_i} c_i + \frac{E'_{i-1}}{V_i}\left(c_{i-1} - c_i\right) + \frac{E'_i}{V_i}\left(c_{i+1} - c_i\right) + \frac{W_i}{V_i} + S_i \tag{70}$$

If this is done, the steady-state solution will be stable but will exhibit numerical dispersion (Chapra 1997).

For equal length segments, $\alpha = 1/2$, which is called *central* or *centered differencing*, results in the interface concentration being the average of the two adjacent elements' concentrations. For this case, the equation becomes

$$\frac{dc_i}{dt} = \frac{Q_{i-1}}{V_i}\frac{c_{i-1} + c_i}{2} - \frac{Q_i}{V_i}\frac{c_i + c_{i+1}}{2} + \frac{E'_{i-1}}{V_i}\left(c_{i-1} - c_i\right) + \frac{E'_i}{V_i}\left(c_{i+1} - c_i\right) + \frac{W_i}{V_i} + S_i \tag{71}$$

If this is done, the steady-state solution may be unstable but if stable will not suffer from numerical dispersion.

Numerical Dispersion

The numerical dispersion can be estimated by (Chapra 1997)

$$E_n \cong U\Delta x\left(\alpha - 0.5\right) \tag{72}$$

where U = mean velocity (m/d).

It can also be shown that to obtain positive solutions the following must hold,

$$\alpha_{i,j} > 1 - \frac{E_{i,j}'}{Q_i} = 1 - \frac{EA_c}{\Delta x_i U A_c} = 1 - \frac{E}{\Delta x_i U} \tag{73}$$

where the dimensionless parameter grouping, $\Delta xU/E$, is called the *element Péclet number*,

$$\mathrm{Pe} = \frac{\Delta x_i U}{E} \tag{74}$$

So, the positivity constraint can also be expressed as

$$\alpha_{i,j} > 1 - \frac{1}{\mathrm{Pe}} \tag{75}$$

If Pe > 10 to 100, you will be more prone to have stability problems with centered differencing. Therefore, for highly advective systems (high Pe or $U\Delta x >> E$), the system is forced to use upwind differencing ($\alpha \to 1$) to maintain stability and we must make a correction for numerical dispersion or accept the numerical dispersion. In contrast, for highly dispersive systems (low Pe or $E >> U\Delta x$), centered differences will be stable and numerical dispersion is eliminated. Note that Q2K minimizes but does not totally eliminate numerical dispersion as described on p. xx.

For advection dominated elements your only option is to reduce the element size. As recognized by Chapra (1997), a critical element length, Δx_c, can be computed as,

$$\Delta x_c < \frac{2E_p}{U} \tag{76}$$

By setting the element length less than this value, numerical instabilities and numerical dispersion can be avoided.

Bottom Algae Mass Balance

For bottom algae, the transport and loading terms are omitted,

$$\frac{da_{b,i}}{dt} = S_{b,i} \tag{77}$$

$$\frac{dIN_{b,i}}{dt} = S_{bN,i} \tag{78}$$

$$\frac{dIP_{b,i}}{dt} = S_{bP,i} \tag{79}$$

where $S_{b,i}$ = sources and sinks of bottom algae biomass due to reactions [gC/m^2/d], $S_{bN,i}$ = sources and sinks of bottom algae nitrogen due to reactions [mgN/m^2/d], and $S_{bP,i}$ = sources and sinks of bottom algae phosphorus due to reactions [mgP/m^2/d].

The sources and sinks for the state variables are depicted in Figure 15 (note that the internal levels of nitrogen and phosphorus in the bottom algae are not depicted). The mathematical representations of these processes are presented in the following sections.

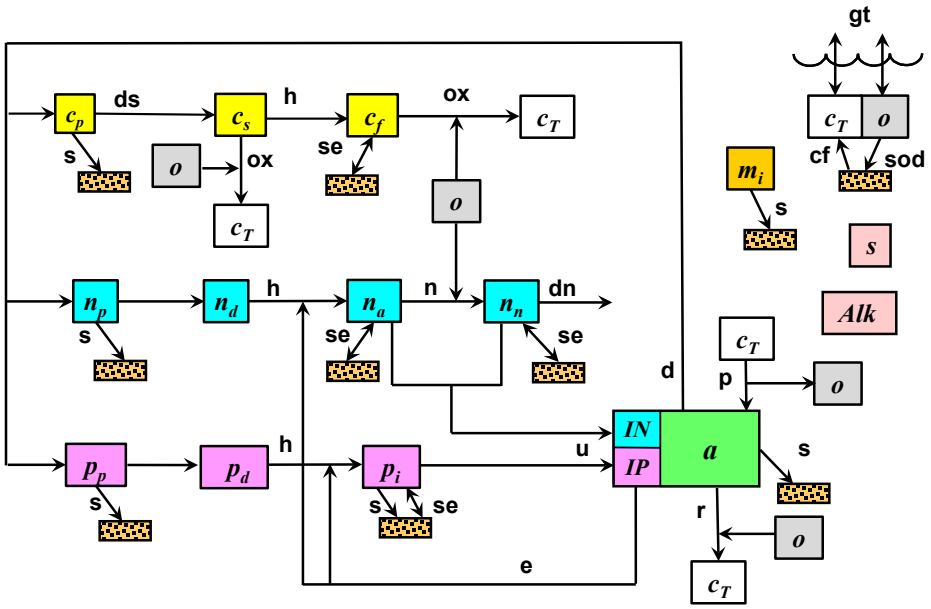

Figure 15 Model kinetics and mass transfer processes. The state variables are defined in Table 5. Kinetic processes are dissolution (ds), hydrolysis (h), oxidation (ox), nitrification (n), denitrification (dn), photosynthesis (p), respiration (r), excretion (e), death (d), respiration/excretion (rx). Mass transfer processes are reaeration (re), settling (s), sediment oxygen demand (SOD), sediment exchange (se), and sediment inorganic carbon flux (cf).

Reaction Fundamentals

Biochemical Reactions

The following chemical equations are used to represent the major biochemical reactions that take place in the model (Stumm and Morgan 1996):

Plant Photosynthesis and Respiration:

Ammonium as substrate:

$$106CO_2 + 16NH_4^+ + HPO_4^{2-} + 106H_2O \underset{R}{\overset{P}{\rightleftharpoons}} C_{106}H_{263}O_{110}N_{16}P_1 + 106O_2 + 14H^+ \quad \textbf{(80)}$$

Nitrate as substrate:

$$106CO_2 + 16NO_3^- + HPO_4^{2-} + 122H_2O + 18H^+ \underset{R}{\overset{P}{\rightleftharpoons}} C_{106}H_{263}O_{110}N_{16}P_1 + 138O_2 \quad \textbf{(81)}$$

The higher oxygen generated when nitrate is assimilated (138 versus 106 moles) occurs because, prior to algal uptake, nitrate must be reduced to ammonium ion via the enzyme, nitrate reductase (Eppley et al. 1969). The stoichiometry of this reaction is

$$NO_3^- + H_2O + 2H^+ \rightarrow NH_4^+ + 2O_2 \quad (82)$$

Thus, nitrate reduction yields 2 moles of oxygen generated per mole of nitrate reduced (Williams et al. 1979; Raine 1985). This will have an impact on modeling dissolved oxygen kinetics as well as pH and alkalinity.

Although the total amount of oxygen generated would be the same as using daily averaging with a Monod model, the temporal evolution will obviously differ when employing the Droop model used in Q2K which treats uptake and growth separately.

Nitrification:

$$NH_4^+ + 2O_2 \rightarrow NO_3^- + H_2O + 2H^+ \quad (83)$$

Denitrification:

$$5CH_2O + 4NO_3^- + 4H^+ \rightarrow 5CO_2 + 2N_2 + 7H_2O \quad (84)$$

Note that additional reactions are used in the model such as those involved with simulating pH and unionized ammonia. These will be outlined when these topics are discussed later in this document.

Organic Matter Stoichiometry

Previous versions of Qual2K (*a*) expressed the concentration of algae (i.e., phytoplankton and bottom algae) as chlorophyll *a* and (*b*) assumed the algae had a constant stoichiometry specified by the user. The following representation was recommended as a first approximation for the stoichiometry (Redfield et al. 1963, Chapra 1997),

$$100\ gD : 40\ gC : 7200\ mgN : 1000\ mgP : 1000\ mgA \quad (85)$$

where gX = mass of element X [g] and mgY = mass of element Y [mg]. The terms D, C, N, P, and A refer to dry weight, carbon, nitrogen, phosphorus, and chlorophyll *a*, respectively. It should be noted that chlorophyll *a* is the most variable of these quantities with a range of approximately 500-2000 mgA (Laws and Chalup 1990, Chapra 1997). Conversions between the units could then be determined with stoichiometric ratios as in

$$r_{xy} = \frac{gX}{gY} \quad (86)$$

where r_{xy} = the mass of constituent *X* contained in a unit mass of constituent *Y*.

For the current version, we use an improved representation of organic matter that is more straightforward and scientifically superior. Algal biomass is now expressed as organic carbon with units of mgC/L for the phytoplankton and gC/m^2 for the bottom algae. Because internal algal nitrogen and phosphorus content are modeled explicitly as state variables, they no longer need to be specified.

Although the new representation simplifies the stoichiometry of Q2K, there are several stoichiometric ratios that are still required.

Dry Weight-to-Carbon Ratio

The model computes total suspended solids (*TSS*) as the sum of inorganic solids, phytoplankton biomass, and particulate organic carbon. Because TSS has units of mgD/L, we need a

stoichiomentric ratio to convert phytoplankton biomass, and particulate organic carbon from carbon to dry weight. Using Eq. (66), this ratio is $\underline{r_{dc}}$ = 100/40 = 2.5 gD/gC.

Carbon-to-Chlorophyll Ratio

Although we now use organic carbon to represent algae, chlorophyll *a* is still a widely used measurement for algal biomass. Hence, it is convenient to be able to convert biomass from organic carbon to chlorophyll units to compare model output in carbon units with measured chlorophyll concentrations. In addition, chlorophyll concentrations are needed to estimate the impact of phytoplankton biomass on light extinction coefficients.

As noted in our discussion of Eq. (66), the chlorophyll-to-carbon ratio has a typical value of $\underline{r_{ac}}$ = 25 with a range from about 12.5 to 50 mgA/gC. For this reason, the current version of Q2K allows the user to specify this parameter with a default value of 25 mgA/gC

Oxygen Generation and Consumption

The model requires that the rates of oxygen generation and consumption be prescribed. Because previous versions of Q2K did not employ Droop kinetics, algal nutrient uptake and algal growth were modeled as a single process. Consequently, different stoichiometric ratios were employed depending on whether ammonia or nitrate were the source of inorganic nitrogen for phytoplankton growth.

Because Q2K separates uptake and growth, a single stoichiometric ratio can be used to determine the grams of oxygen generated for each gram of plant matter that is produced through photosynthesis,

$$r_{oca} = \frac{106 \text{ moleO}_2(32 \text{ gO}_2/\text{moleO}_2)}{106 \text{ moleC}(12 \text{ gC/moleC})} = 2.67 \frac{\text{gO}_2}{\text{gC}} \quad (87)$$

Note that Eq. (84) is also used for the stoichiometry of the amount of oxygen consumed for plant respiration.

For nitrification, the following ratio is based on Eq. (69)

$$r_{on} = \frac{2 \text{ moleO}_2(32 \text{ gO}_2/\text{moleO}_2)}{1 \text{ moleN}(14 \text{ gN/moleN})} = 4.57 \frac{\text{gO}_2}{\text{gN}} \quad (88)$$

This ratio is also used to compute the oxygen generated

CBOD Utilization Due to Denitrification

As represented by Eq. (70), CBOD is utilized during denitrification,

$$r_{ondn} = 2.67 \frac{\text{gO}_2}{\text{gC}} \frac{5 \text{ moleC} \times 12 \text{ gC/moleC}}{4 \text{ moleN} \times 14 \text{ gN/moleN}} \times \frac{1 \text{ gN}}{1000 \text{ mgN}} = 0.00286 \frac{\text{gO}_2}{\text{mgN}} \quad (89)$$

Temperature Effects on Reactions

The temperature effect for all first-order reactions used in the model is represented by

$$k(T) = k(20)\theta^{T-20} \quad (90)$$

where $k(T)$ = the reaction rate [/d] at temperature T [ºC] and θ = the temperature coefficient for the reaction.

Composite Variables

In addition to the model's state variables, Q2K also displays several composite variables that are computed as follows:

Total Organic Carbon (mgC/L):

$$TOC = c_p + c_s + c_f + a_p \quad (91)$$

Total Nitrogen (μgN/L):

$$TN = n_{op} + n_{od} + n_a + n_n + IN_{p1} + IN_{p2} \quad (92)$$

Total Phosphorus (μgP/L):

$$TP = p_{op} + p_{od} + p_i + IP_{p1} + IP_{p2} \quad (93)$$

Total Kjeldahl Nitrogen (μgN/L):

$$TKN = n_{op} + n_{od} + n_a + IN_{p1} + IN_{p2} \quad (94)$$

Total Suspended Solids (mgD/L):

$$TSS = m_i + r_{dc}\left(c_{POC} + c_p\right) \quad (95)$$

Ultimate Carbonaceous BOD (mgO_2/L):

$$CBOD_u = r_{oc} TOC \quad (96)$$

Relationship of Model Variables and Data

For all but slow and fast CBOD (c_f and c_s), there exists a relatively straightforward relationship between the model state variables and standard water-quality measurements. These are outlined next. Then we discuss issues related to the more difficult problem of measuring CBOD.

Non-CBOD Variables and Data

The following are measurements that are needed for comparison of non-BOD variables with model output:

TEMP = temperature (°C)
TKN = total kjeldahl nitrogen (μgN/L) or TN = total nitrogen (μgN/L)
NH4 = ammonium nitrogen (μgN/L)
NO2 = nitrite nitrogen (μgN/L)
NO3 = nitrate nitrogen (μgN/L)
CHLA = chlorophyll *a* (μgA/L)
TP = total phosphorus (μgP/L)
SRP = soluble reactive phosphorus (μgP/L)
TSS = total suspended solids (mgD/L)
VSS = volatile suspended solids (mgD/L)
TOC = total organic carbon (mgC/L)
DOC = dissolved organic carbon (mgC/L)
DO = dissolved oxygen (mgO_2/L)
PH = pH
ALK = alkalinity ($mgCaCO_3$/L)

COND = specific conductance (μmhos)

The model state variables can then be related to these measurements as follows:

s = COND
m_i = TSS – VSS or TSS – r_{dc} (TOC – DOC)
o = DO
n_o = TKN – NH4 – r_{na} CHLA or n_o = TN – NO2 – NO3 – NH4 – rna CHLA
n_a = NH4
n_n = NO2 + NO3
p_o = TP – SRP – r_{pa} CHLA
p_i = SRP
a_p = CHLA
m_o = VSS – r_{da} CHLA or r_{dc} (TOC – DOC) – r_{da} CHLA
pH = PH
Alk = ALK

Carbonaceous BOD

The interpretation of BOD measurements in natural waters is complicated by three primary factors:

- Filtered versus unfiltered. If the sample is unfiltered, the BOD will reflect oxidation of both dissolved and particulate organic carbon. Since Q2K distinguishes between dissolved (c_s and c_f) and particulate (m_o and a_p) organics, an unfiltered measurement alone does not provide an adequate basis for distinguishing these individual forms. In addition, one component of the particulate BOD, phytoplankton (a_p) can further complicate the test through photosynthetic oxygen generation.
- Nitrogenous BOD. Along with the oxidation of organic carbon (CBOD), nitrification also contributes to oxygen depletion (NBOD). Thus, if the sample (a) contains reduced nitrogen and (b) nitrification is not inhibited, the measurement includes both types of BOD.
- Incubation time. Short-term, usually 5-day, BODs are typically performed. Because Q2K uses ultimate CBOD, 5-day BODs must be converted to ultimate BODs in a sensible fashion.

We suggest the following as practical ways to measure CBOD in a manner that accounts for the above factors and results in measurements that are compatible with Q2K.

Filtration. The sample should be filtered prior to incubation in order to separate dissolved from particulate organic carbon.

Nitrification inhibition. Nitrification can be suppressed by adding a chemical inhibiting agent such as TCMP (2-chloro-6-(trichloro methyl) pyridine. The measurement then truly reflects CBOD. In the event that inhibition is not possible, the measured value can be corrected for nitrogen by subtracting the oxygen equivalents of the reduced nitrogen (= r_{on} × TKN) in the sample. However, as with all such difference-based adjustments, this correction may exhibit substantial error.

Incubation time. The model is based on ultimate CBOD, so two approaches are possible: (1) use a sufficiently long period so that the ultimate value is measured, or (2) use a 5-day measurement and extrapolate the result to the ultimate. The latter method is often computed with the formula

$$\text{CBODFNU} = \frac{\text{CBODFN5}}{1 - e^{-k_1 5}} \quad (97)$$

where CBODFNU[3] = the ultimate dissolved carbonaceous BOD [mgO_2/L], CBODFN5 = the 5-day dissolved carbonaceous BOD [mgO_2/L], and k_1 = the CBOD decomposition rate in the bottle [/d].

It should be noted that, besides practical considerations of time and expense, there may be other benefits from using the 5-day measurement with extrapolation, rather that performing the longer-term CBOD. Although extrapolation does introduce some error, the 5-day value has the advantage that it would tend to minimize possible nitrification effects which, even when inhibited, can begin to be exerted on longer time frames.

If all the above provisions are implemented, the result should correspond to the model variables by

$$c_f + c_s = \text{CBODFNU}$$

Slow versus Fast CBOD. The final question relates to discrimination between fast and slow CBODU. Although we believe that there is currently no single, simple, economically feasible answer to this problem, we think that the following 2 strategies represent the best current alternatives.

Option 1: Represent all the dissolved, oxidizable organic carbon with a single pool (fast CBOD). The model includes parameters to bypass slow CBOD. If no slow CBOD inputs are entered, this effectively drops it from the model. For this case,

$$c_f = \text{CBODFNU}$$

$$c_s = 0$$

Option 2: Use an ultimate CBOD measurement for the fast fraction and compute slow CBOD by difference with a DOC measurement. For this case,

$$c_f = \text{CBODFNU}$$

$$c_s = r_{oc}\text{DOC} - \text{CBODFNU}$$

Option 2 works very nicely for systems where two distinct types of CBOD are present. For example, sewage effluent and autochthonous carbon from the aquatic food chain might be considered as fast CBOD. In contrast, industrial wastewaters such as pulp and paper mill effluent or allochthonous DOC from the watershed might be considered more recalcitrant and hence could be lumped into the slow CBOD fraction. In such case, the hydrolysis rate converting the slow into the fast fraction could be set to zero to make the two forms independent.

For both options, the CBODFNU can either be (a) measured directly using a long incubation time or (b) computed by extrapolation with Eq. 75. In both situations, a time frame of several weeks to a month (i.e., a 20- to 30-day CBOD) is probably a valid period in order to oxidize most of the readily degradable organic carbon. We base this assumption on the fact that bottle rates for sewage-derived organic carbon are on the order of 0.05 to 0.3/d (Chapra 1997). As in Figure 16, such rates suggest that much of the readily oxidizable CBOD will be exerted in about 20 to 30 days.

[3] The nomenclature FNU stands for Filtration, Nitrification inhibition and Ultimate

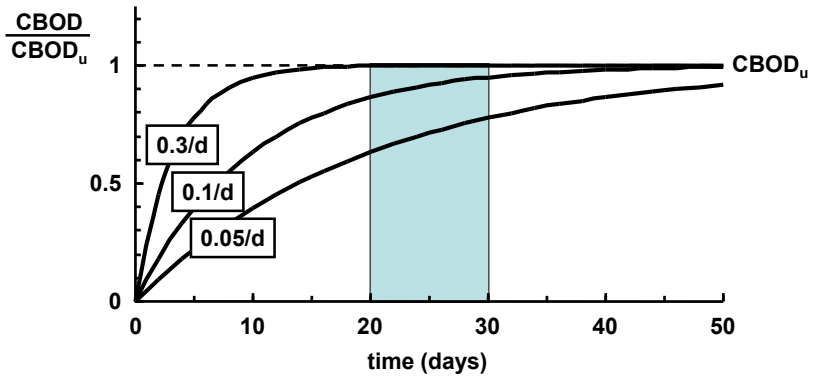

Figure 16 Progression of CBOD test for various levels of the bottle decomposition rate.

In addition, we believe that practitioners should consider conducting long-term CBOD tests at 30°C rather than at the commonly employed temperature of 20°C. The choice of 20°C originated from the fact that the average daily temperature of most receiving waters and wastewater treatment plants in the temperate zone in summer is approximately 20°C.

If the CBOD measurement is intended to be used for regulation or to assess treatment plant performance, it makes sense to standardize the test at a particular temperature. And for such purposes, 20°C is as reasonable a choice as any. However, if the intent is to measure an ultimate CBOD, anything that speeds up the process while not jeopardizing the measurement's integrity would seem beneficial.

The saprophytic bacteria that break down nonliving organic carbon in natural waters and sewage thrive best at temperatures from 20°C to 40°C. Thus, a temperature of 30°C is not high enough that the bacterial assemblage would shift to thermophilic organisms that are atypical of natural waters and sewage. The benefit should be higher oxidation rates which would result in shorter analysis times for CBOD measurements. Assuming that a $Q_{10} \cong 2$ is a valid approximation for bacterial decomposition, a 20-day BOD at 30°C should be equivalent to a 30-day BOD at 20°C.

Constituent Reactions

The mathematical relationships that describe the individual reactions and concentrations of the model state variables (Table 5) are presented in the following paragraphs.

Specific Conductance (*s*)

Conservative substances are not subject to reactions. Therefore,

$$S_s = 0 \tag{98}$$

Phytoplankton (c_p)

The following descriptions of phytoplankton biomass and internal nutrients model equations are used for both of the functional groups of phytoplankton currently in Q2K. In most cases, we do not present separate equations for the two cases. However, where necessary, we use the subscripts, “1” and “2” to distinguish between the functional groups. This is the case for light extinction as well as some later sections where the two groups are distinguished by subscripts “1” and “2”.

Phytoplankton biomass increases due to photosynthesis. They are lost via respiration, death, and settling

$$S_{cp} = \text{PhytoPhoto} - \text{PhytoResp} - \text{PhytoDeath} - \text{PhytoSettl} - \text{PhytoSalTox} - \text{PhytoFreshTox} \quad (99)$$

Photosynthesis

Phytoplankton photosynthesis is computed as

$$\text{PhytoPhoto} = \mu_p c_p \quad (100)$$

where μ_p = phytoplankton photosynthesis rate [/d], which is a function of temperature, nutrients, and light,

$$\mu_p = k_{gp}(T)\phi_{Np}\phi_{Lp} \quad (101)$$

where $k_{gp}(T)$ = the maximum photosynthesis rate at temperature T [/d], ϕ_{Np} = phytoplankton nutrient attenuation factor [dimensionless number between 0 and 1], and ϕ_{Lp} = the phytoplankton light attenuation coefficient [dimensionless number between 0 and 1].

Nutrient Limitation

The nutrient limitation due to inorganic carbon is represented by a Michaelis-Menten equation. In contrast, for nitrogen and phosphorus, the photosynthesis rate depends on intracellular nutrient levels using a formulation originally developed by Droop (1974). The minimum value is then employed to compute the nutrient attenuation factor,

$$\phi_{Np} = \min\left[1 - \frac{q_{0Np}}{q_{Np}}, 1 - \frac{q_{0Pp}}{q_{Pp}}, \frac{[H_2CO_3^*] + [HCO_3^-]}{k_{sCp} + [H_2CO_3^*] + [HCO_3^-]}\right] \quad (102)$$

where q_{Np} and q_{Pp} = the phytoplankton cell quotas of nitrogen [mgN gC^{-1}] and phosphorus [mgP gC^{-1}], respectively; q_{0Np} and q_{0Pp} = the minimum phytoplankton cell quotas of nitrogen [mgN gC^{-1}] and phosphorus [mgP gC^{-1}], respectively; k_{sCp} = inorganic carbon half-saturation constant for phytoplankton [mole/L]; $[H_2CO_3^*]$ = dissolved carbon dioxide concentration [mole/L]; and $[HCO_3^-]$ = bicarbonate concentration [mole/L]. The minimum cell quotas are the levels of intracellular nutrient at which growth ceases. Note that the nutrient limitation terms cannot be negative. That is, if $q < q_0$, the limitation term is set to 0.

The cell quotas represent the ratios of the intracellular nutrient to the phytoplankton biomass,

$$q_{Np} = \frac{IN_p}{c_p} \quad (103)$$

$$q_{Pp} = \frac{IP_p}{c_p} \quad (104)$$

where IN_p = phytoplankton intracellular nitrogen concentration [μgN/L] and IP_p = phytoplankton intracellular phosphorus concentration [μgP/L].

Light Limitation

It is assumed that light attenuation through the water follows the Beer-Lambert law,

$$PAR(z) = PAR(0)e^{-k_e z} \quad (105)$$

where $PAR(z)$ = photosynthetically active radiation (PAR) at depth z below the water surface [ly/d][4], and k_e = the light extinction coefficient [m^{-1}]. The PAR at the water surface is assumed to be a fixed fraction of the solar radiation (Szeicz 1984, Baker and Frouin 1987):

$$PAR(0) = 0.47\ I(0)$$

The extinction coefficient is related to model variables by

$$k_e = k_{eb} + \alpha_i m_i + \alpha_o r_{dc} c_p + \alpha_p (r_{ac} c_p) + \alpha_{pn} (r_{ac} c_p)^{2/3} \quad (106)$$

where k_{eb} = the background coefficient accounting for extinction due to water and color [/m], α_i, α_o, α_p, and α_{pn}, are constants accounting for the impacts of inorganic suspended solids [L/mgD/m], POC expressed as dry weight [L/mgD/m], and chlorophyll *a* [L/µgA/m and $(L/\mu gA)^{2/3}/m$], respectively. Suggested values for these coefficients are listed in Table 6.

Table 6 Suggested values for light extinction coefficients

Symbol	Value	Reference
α_i	0.052	Di Toro (1978)
α_o	0.174	Di Toro (1978)
α_p	0.0088	Riley (1956)
α_{pn}	0.054	Riley (1956)

Three models are used to characterize the impact of light on phytoplankton photosynthesis (Figure 17):

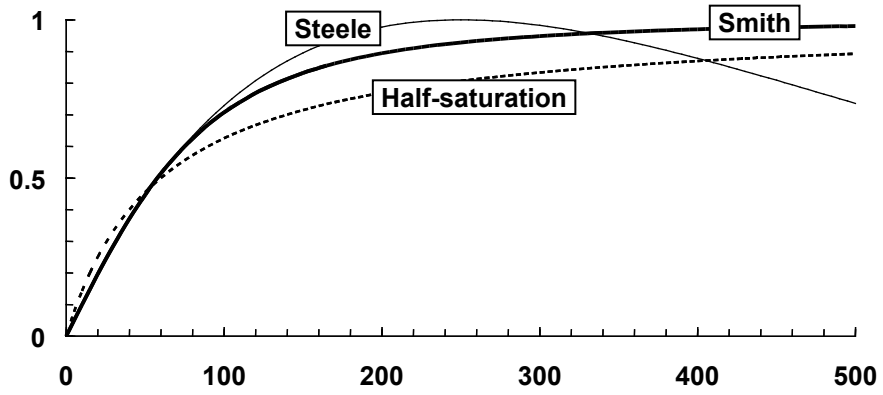

Figure 17 The three models used for phytoplankton and bottom algae photosynthetic light dependence. The plot shows growth attenuation versus PAR intensity [ly/d].

Half-Saturation (Michaelis-Menten) Light Model (Baly 1935):

$$F_{Lp} = \frac{PAR(z)}{K_{Lp} + PAR(z)} \quad (107)$$

[4] ly/d = langley per day. A langley is equal to a calorie per square centimeter. Note that a ly/d is related to the $\mu E/m^2/s$ by the following approximation: 1 $\mu E/m^2/s \cong 0.45$ Langley/day (LIC-OR, Lincoln, NE).

where F_{Lp} = phytoplankton growth attenuation due to light and K_{Lp} = the phytoplankton light parameter. In the case of the half-saturation model, the light parameter is a half-saturation coefficient [ly/d]. This function can be combined with the Beer-Lambert law and integrated over water depth, H [m], to yield the phytoplankton light attenuation coefficient

$$\phi_{Lp} = \frac{1}{k_e H} \ln\left(\frac{K_{Lp} + PAR(0)}{K_{Lp} + PAR(0)e^{-k_e H}} \right) \quad \textbf{(108)}$$

Smith's Function (Smith 1936):

$$F_{Lp} = \frac{PAR(z)}{\sqrt{K_{Lp}^2 + PAR(z)^2}} \quad \textbf{(109)}$$

where K_{Lp} = the Smith parameter for phytoplankton [ly/d]; that is, the PAR at which growth is 70.7% of the maximum. This function can be combined with the Beer-Lambert law and integrated over water depth to yield

$$\phi_{Lp} = \frac{1}{k_e H} \ln\left(\frac{PAR(0)/K_{Lp} + \sqrt{1 + \left(PAR(0)/K_{Lp}\right)^2}}{\left(PAR(0)/K_{Lp}\right) e^{-k_e H} + \sqrt{1 + \left(\left(PAR(0)/K_{Lp}\right) e^{-k_e H}\right)^2}} \right) \quad \textbf{(110)}$$

Steele's Equation (Steele 1962):

$$F_{Lp} = \frac{PAR(z)}{K_{Lp}} e^{1 - \frac{PAR(z)}{K_{Lp}}} \quad \textbf{(111)}$$

where K_{Lp} = the PAR at which phytoplankton growth is optimal [ly/d]. This function can be combined with the Beer-Lambert law and integrated over water depth to yield

$$\phi_{Lp} = \frac{2.718282}{k_e H} \left(e^{-\frac{PAR(0)}{K_{Lp}} e^{-k_e H}} - e^{-\frac{PAR(0)}{K_{Lp}}} \right) \quad \textbf{(112)}$$

Losses

Respiration

Phytoplankton respiration is represented as a first-order rate of biomass that is attenuated at low oxygen concentration,

$$\text{PhytoResp} = F_{oxp} k_{rp}(T)\, a_p \quad \textbf{(113)}$$

where $k_{rp}(T)$ = temperature-dependent phytoplankton respiration/excretion rate [/d] and F_{oxp} = attenuation due to low oxygen [dimensionless]. Oxygen attenuation is modeled by Eqs. (127) to (129) with the oxygen dependency represented by the parameter K_{sop}.

Death

Phytoplankton death is represented as a first-order rate,

$$\text{PhytoDeath} = k_{dp}(T) c_p \quad \textbf{(114)}$$

where $k_{dp}(T)$ = temperature-dependent phytoplankton death rate [/d].

Salinity Toxicity

Some freshwater phytoplankton die when they enter high salinity waters like estuaries. This effect can be represented by a first-order salinity dependent death rate,

$$\text{PhytoSalinity} = \phi_{dS} k_{dS}(T) c_p \tag{115}$$

where $k_{dS}(T)$ = temperature-dependent phytoplankton salinity death rate [/d], and ϕ_{dS} = a salinity dependent mortality factor. The latter is a dimensionless number between 0 and 1, where $\phi_{dS} = 0$ corresponds to no salinity mortality and $\phi_{dS} = 1$ to maximum salinity mortality.

Note that there are also saltwater phytoplankton that can die when exposed to low salinity water. This effect can also be represented by a first-order salinity dependent death rate, but with the mortality factor modified as in

$$\text{PhytoSalinity} = (1 - \phi_{dS}) k_{dS}(T) c_p \tag{116}$$

For this case, when $\phi_{dS} = 0$ (freshwater), $1 - \phi_{dS} = 1$ mortality is at a maximum. When $\phi_{dS} = 1$ (saltwater), $1 - \phi_{dS} = 0$ and the salinity dependent mortality is 0.

For Q2K, the mortality factor is represented by two types of threshold functions: (1) sigmoid functions and (2) saturating hyperbolas.

Sigmoid (or Logistic) functions. These functions are appropriate for cases where death gradually increases over a range centered at a particular salinity concentration as in

$$\phi_{sS} = \frac{1}{1 + e^{-n_n (S - S_0)}} \tag{117}$$

where S_0 = the value of the sigmoid's midpoint (ppt), and n_n = the logistic growth rate (1/ppt) which governs the function's steepness. Note that if you specify S_0 and n_n, Eq. (118) is fully determined. Hence, it is a two-parameter function.

Figure 18 depicts several examples for different values of n_n. Because $n_n = 0$ yields a constant value of $\phi_{ds} = 0.5$, n_n should be set to a value greater than zero. As n_n increases, the function steepens. For very high values, Eq. (118) yields a step function that abruptly rises from 0 to 1 at $S = S_0$.

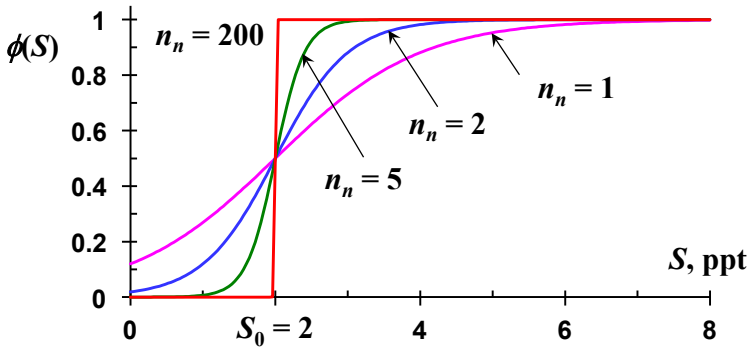

Figure 18 Several examples of threshold saturating hyperbolas with progressively higher logistic growth rates. For all cases, S_0 = 2 ppt and $S_{0.5}$ = 2.

Saturating hyperbolas. These are appropriate for cases where death is triggered abruptly at a particular salinity concentration as in

$$\phi_{dS} = \begin{cases} 0 & \text{for } S \le 0 \\ \dfrac{(S - S_0)}{\left((S_s - S_0)^{n_n} + (S - S_0)^{n_n}\right)^{1/n_n}} & \text{for } S > S_0 \end{cases} \tag{118}$$

where S_0 = the threshold or "trigger" salinity at which toxicity begins (ppt), S_s = the salinity half-saturation constant (ppt), and n_n = the order of the saturating hyperbola, which governs the steepness of the function as it rise from 0 to 1.

It can be shown that the following equation can be used to determine the S_s so that $\phi_{dS} = 0.5$ at the desired salinity of $S_{0.5}$,

$$S_s = S_0 + (2^{n_n} - 1)^{1/n_n}(S_{0.5} - S_0) \tag{119}$$

Consequently, if you specify S_0, $S_{0.5}$, and n_n, Eq. (119) is fully determined. Hence, it is a three-parameter function.

Several cases are illustrated in Figure 19. When n_n is set to 1, the result is a threshold version of the standard first-order Michaelis-Menten function (recall Eq. 108),

$$\phi_{dS} = \begin{cases} 0 & \text{for } S \le 0 \\ \dfrac{(S - S_0)}{S_s + (S - S_0)} & \text{for } S > S_0 \end{cases} \tag{120}$$

that rises from 0 at $S = S_0$ and reaches $\phi_{dS} = 0.5$ at S_s.

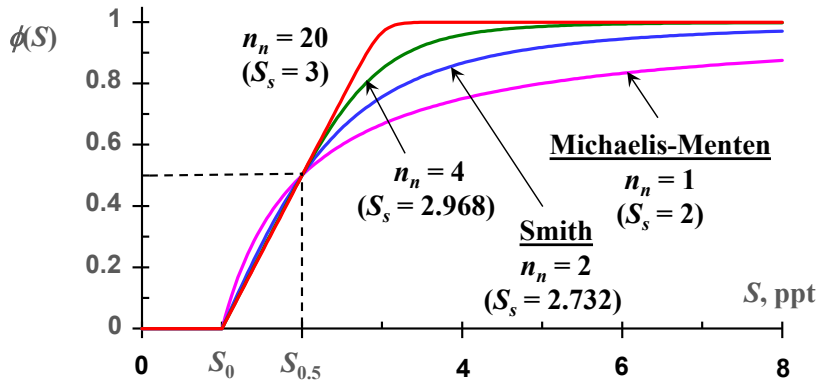

Figure 19 Several examples of threshold saturating hyperbolas with the threshold set to S_0 = 1 ppt and the half-saturation salinity set to $S_{0.5}$ = 2.

When $n_n = 2$, the result is a threshold second-order Michaelis-Menten or Smith function (recall Eq. 110),

$$\phi_{dS} = \begin{cases} 0 & \text{for } S \le 0 \\ \dfrac{(S - S_0)}{\sqrt{S_s^2 + (S - S_0)^2}} & \text{for } S > S_0 \end{cases} \quad (121)$$

that also reaches $\phi_{dS} = 0.5$ at $S = S_{0.5}$, but thereafter rises more rapidly to 1. Higher values of n_n make the result steeper. Note that as n_n is set to a very high value (e.g., $n_n = 20$), the function approaches a ramp function that rises linearly from 0 to 1 with a slope of $0.5/(S_{0.5} - S_0)$. If you specify S_0, $S_{0.5}$, and n_n, Eq. (119) is fully determined. Hence, this is a three-parameter function.

Figure 20 provides an example for a river estuary flowing into the ocean. The top plot (*a*) depicts the rise in salinity as the river approaches the ocean boundary. The bottom plot (a) depicts the logistic model ($n_n = 4$ and $S_0 = 2$ ppt) and the saturating hyperbolic model ($n_n = 2$, $S_0 = 1$, $S_{0.5} = 2$ ppt, and $S_s = 2.73205$ ppt). For both models, salinity toxicity to freshwater phytoplankton and freshwater toxicity to saltwater phytoplankton are depicted.

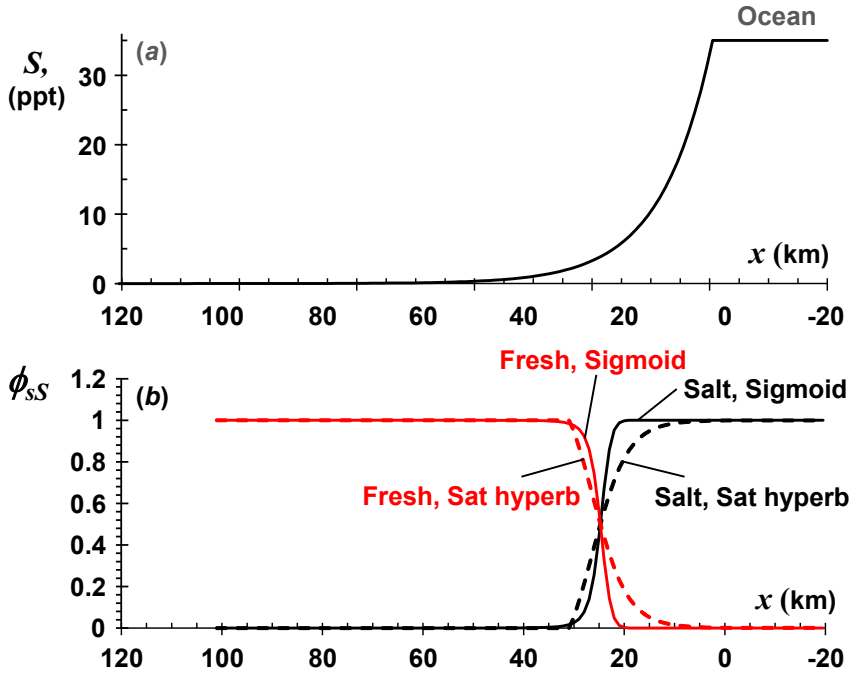

Figure 20 Plots of (a) salinity (ppt) and (b) salinity dependent mortality factors versus distance (km) where x = 0 at the ocean boundary. For (b), several models of mortality factors are depicted.

Settling

Phytoplankton settling is represented as

$$\text{PhytoSettl} = \frac{v_p}{H} c_p \quad (122)$$

where v_v = phytoplankton settling velocity [m/d].

Phytoplankton Internal Nitrogen (IN_p)

The change in intracellular nitrogen in phytoplankton cells is calculated from

$$S_{pN} = \text{PhytoUpN} - q_{Np}\ \text{PhytoDeath} - \text{PhytoExN} \quad (123)$$

where PhytoUpNH4 = the uptake rate of nitrogen by phytoplankton (μgN/L/d), PhytoDeath = phytoplankton death (μgN/L/d), and PhytoExN = the phytoplankton excretion of nitrogen (μgN/L/d), which is computed as

$$\begin{aligned} \text{PhytoExN} &= 0 && \text{if } IN_p \le q_{0Np} \\ \text{PhytoExN} &= k_{ep}(T)\left(IN_p - q_{0Np}\right) && \text{if } IN_p > q_{0Np} \end{aligned} \quad (124)$$

where $k_{ep}(T)$ = the temperature-dependent phytoplankton excretion rate [/d].

The N uptake rate depends on both external and intracellular nutrients as in (Rhee 1973),

$$\text{PhytoUpN} = \rho_{mNp} \frac{n_a + n_n}{k_{sNp} + n_a + n_n} \frac{K_{qNp}}{K_{qNp} + (q_{Np} - q_{0Np})} a_p \quad (125)$$

where ρ_{mNp} = the maximum uptake rate for nitrogen [mgN/gC/d], k_{sNp} = half-saturation constant for external nitrogen [μgN/L] and K_{qNp} = half-saturation constant for intracellular nitrogen [mgN gC^{-1}].

Phytoplankton Internal Phosphorus (IP_p)

The change in intracellular phosphorus in phytoplankton cells is calculated from

$$S_{pP} = \text{PhytoUpP} - q_{Pp}\ \text{PhytoDeath} - \text{PhytoExP} \quad (126)$$

where PhytoUpP = the uptake rate of phosphorus by phytoplankton (μgP/L/d), PhytoDeath = phytoplankton death (μgP/L/d), and PhytoExP = the phytoplankton excretion of phosphorus (μgP/L/d), which is computed as

$$\begin{aligned} \text{PhytoExP} &= 0 && \text{if } IP_p \le q_{0Pp} \\ \text{PhytoExP} &= k_{ep}(T)\left(IP_p - q_{0Pp}\right) && \text{if } IP_p > q_{0Pp} \end{aligned} \quad (127)$$

where $k_{ep}(T)$ = the temperature-dependent phytoplankton excretion rate [/d].

The P uptake rate depends on both external and intracellular nutrients as in (Rhee 1973),

$$\text{PhytoUpP} = \rho_{mPp} \frac{p_i}{k_{sPp} + p_i} \frac{K_{qPp}}{K_{qPp} + (q_{Pp} - q_{0Pp})} a_p \quad (128)$$

where ρ_{mPp} = the maximum uptake rate for phosphorus [mgP/gC/d], k_{sPp} = half-saturation constant for external phosphorus [μgP/L] and K_{qPp} = half-saturation constant for intracellular phosphorus [mgP gC^{-1}].

Bottom algae biomass (c_b)

The following model equations for bottom algae biomass and internal nutrients are used for the two functional groups of bottom algae currently in Q2K. Note that the specific parameters for the two groups are distinguished in later sections by subscripts “1” and “2”.

Bottom algae biomass increases due to photosynthesis. They are lost via respiration and death.

$$S_{ab} = \text{BotAlgPhoto} - \text{BotAlgResp} - \text{BotAlgDeath} \quad (129)$$

Photosynthesis

Two representations can be used to model bottom algae photosynthesis. The first is based on a temperature-corrected zero-order rate attenuated by nutrient and light limitation (McIntyre 1973, Rutherford et al. 1999),

$$\text{BotAlgPhoto} = C_{gb}(T)\phi_{Nb}\phi_{Lb} \quad (130)$$

where $C_{gb}(T)$ = the zero-order temperature-dependent maximum photosynthesis rate [gC/(m^2 d)], ϕ_{Nb} = bottom algae nutrient attenuation factor [dimensionless number between 0 and 1], and ϕ_{Lb} = the bottom algae light attenuation coefficient [dimensionless number between 0 and 1].

The second uses a first-order model,

$$\text{BotAlgPhoto} = C_{gb}(T)\phi_{Nb}\phi_{Lb}\phi_{Sb}c_b \quad (131)$$

where, for this case, $C_{gb}(T)$ = the first-order temperature-dependent maximum photosynthesis rate [d^{-1}], and ϕ_{Sb} = bottom algae space limitation attenuation factor.

Temperature

As for the first-order rates, an Arrhenius model is employed to quantify the effect of temperature on bottom algae photosynthesis,

$$C_{gb}(T) = C_{gb}(20)\theta^{T-20} \quad (132)$$

Nutrient Limitation

The effect of nutrient limitation on bottom plant photosynthesis is modeled in the same way as for the phytoplankton. That is, a Droop (1974) formulation is used for nitrogen and phosphorus limitation whereas a Michaelis-Menten equation is employed for inorganic carbon,

$$\phi_{Nb} = \min\left[1 - \frac{q_{0Nb}}{q_{Nb}}, 1 - \frac{q_{0Pb}}{q_{Pb}}, \frac{[H_2CO_3^*] + [HCO_3^-]}{k_{sCb} + [H_2CO_3^*] + [HCO_3^-]}\right] \quad (133)$$

where q_{Nb} and q_{Pb} = the bottom algae cell quotas of nitrogen [mgN gC^{-1}] and phosphorus [mgP gC^{-1}], respectively, q_{0Nb} and q_{0Pb} = the bottom algae minimum cell quotas of nitrogen [mgN gC^{-1}] and phosphorus [mgP gC^{-1}], respectively, and k_{sCb} = the bottom algae inorganic carbon half-saturation constant [mole/L]. As was the case for phytoplankton, the nutrient limitation terms cannot be negative.

The cell quotas represent the ratios of the intracellular nutrient to the bottom plant's biomass,

$$q_{Nb} = \frac{IN_b}{c_b} \quad (134)$$

$$q_{Pb} = \frac{IP_b}{c_b} \quad (135)$$

where IN_b = intracellular nitrogen concentration [mgN/m^2] and IP_b = intracellular phosphorus concentration [mgP/m^2].

Space Limitation

If a first-order growth model is used, a term must be included to impose a space limitation on the bottom algae. A logistic model is used for this purpose as in

$$\phi_{Sb} = 1 - \frac{c_b}{c_{b,\max}}$$

where $c_{b,\max}$ = the carrying capacity [gC/m^2].

Light Limitation

When two functional groups of bottom algae are simulated,Q2K handles light limitation in two different ways: (1) As a homogeneous biofilm, or (2) layered bottom plants.

Homogeneous photosynthetic biofilm. This way assumes that the two bottom algae functional groups are uniformly distributed through a benthic biofilm. An example would be when the biofilm consists of both diatoms and cyanobacteria. For this situation, light limitation for both groups is determined by the amount of PAR reaching the top of the biofilm as computed with the Beer-Lambert law (recall Eq. 83),

$$PAR(H) = PAR(0)e^{-k_e H} \tag{136}$$

As with the phytoplankton, three models (recall Eqs. 85, 87 and 89) are used to characterize the impact of light on bottom algae photosynthesis. It is assumed that all the PAR is utilized in the biofilm. This implied that there is minimal reflection and that there is no significant radiation reaching the underlying rock substrate. Substituting Eq. (107) into these models yields the following formulas for the bottom algae light attenuation coefficient,

Half-Saturation Light Model (Baly 1935):

$$\phi_{Lb} = \frac{PAR(H)}{K_{Lb} + PAR(H)} \tag{137}$$

Smith's Function: (Smith 1936)

$$\phi_{Lb} = \frac{PAR(H)}{\sqrt{K_{Lb}^2 + PAR(H)^2}} \tag{138}$$

Steele's Equation (Steele 1962):

$$\varphi_{Lb} = \frac{PAR(H)}{K_{Lb}} e^{1-\frac{PAR(H)}{K_{Lb}}} \tag{139}$$

where K_{Lb} = the appropriate bottom algae light parameter for each light model.

Layered Bottom Plants. This is the case where the groups are layered vertically (Figure 21). An example would be periphyton growing on a rocky substrate are shaded by attached filamentous algae such as Cladophora.

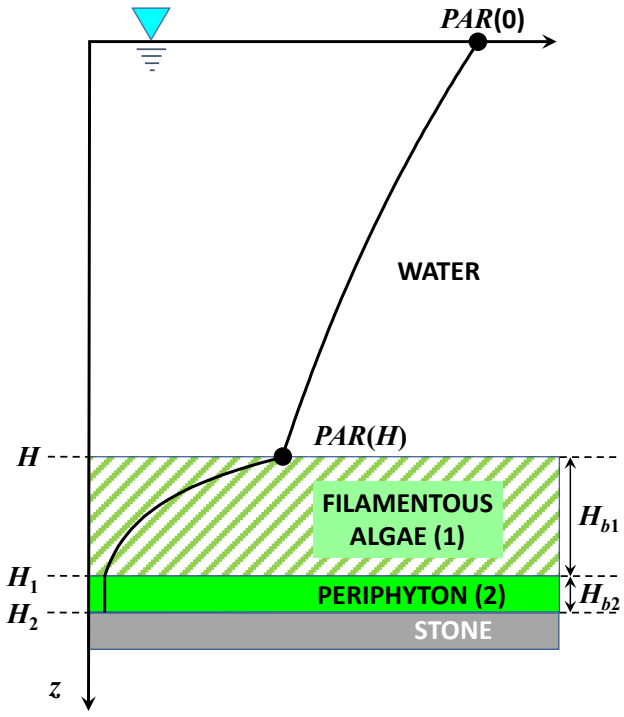

Figure 21 Light extinction for a system where a periphyton biofilm is overlain by filamentous algae overly.

The light delivered to the top of the periphyton is computed with the Beer-Lambert law,

$$PAR(H_{b1}) = PAR(H)e^{-k_{eb1}H_{b1}} \tag{140}$$

where H_{b1} = the thickness of the filamentous layer (m), and k_{eb1} = the light extinction coefficient of the filamentous algae (m). Note that the filamentous layer thickness can be related to its areal biomass by

$$H_{b1} = \frac{c_{b1}}{\rho_{b1}} \tag{141}$$

where ρ_{b1} = the density of the filamentous algae layer (gC/m^3); that is, the biomass of Cladophora per unit layer volume. Rearranging this equation yields

$$\rho_{b1} = \frac{c_{b1}}{H_{b1}} \tag{142}$$

Thus, this parameter can be quantified by plotting the thickness of the filamentous layer versus its corresponding biomass. Of course, the density also depends on the river velocity (Flynn et al. 2018).

The light limitation for the filamentous algae would then be calculated with Eqs. (107), (109), and (111), but with appropriate coefficients, $PAR(0) = PAR(H)$, $K_{Lp} = K_{Lb1}$, $H = H_{b1}$, and $k_e = k_{eb1}$. For example, if the Michaelis-Menten model is chosen, Eq. (109) would be rewritten as

$$\phi_{Lb} = \frac{1}{k_{eb1}H_{b1}} \ln\left(\frac{K_{Lb1} + PAR(H)}{K_{Lb1} + PAR(H)e^{-k_{eb1}H_{b1}}}\right) \tag{143}$$

The Smith and Steele functions would be handled in a similar fashion.

For the periphyton, the PAR delivered to the top of the periphyton layer, $PAR(H_{b1})$, is assumed to be totally available to the periphyton. The light limitation for the periphyton would then be calculated with Eqs. (108), (110), and (112), but with appropriate coefficients, $PAR(0) = PAR(H_1)$, and $K_{Lp} = K_{Lb2}$. For example, if the Michaelis-Menten model is chosen, Eq. (109) would be rewritten as

$$\phi_{Lb2} = \frac{PAR(H_1)}{K_{Lb2} + PAR(H_1)} \quad (144)$$

Losses

Respiration. Bottom algae respiration is represented as a first-order rate that is attenuated at low oxygen concentration,

$$\text{BotAlgResp} = F_{oxb} k_{rb}(T)\, c_b \quad (145)$$

where $k_{rb}(T)$ = temperature-dependent bottom algae respiration rate [/d] and F_{oxb} = attenuation due to low oxygen [dimensionless]. Oxygen attenuation is modeled by Eqs. (127) to (129) with the oxygen dependency represented by the parameter K_{sob}.

Death. Bottom algae death is represented as a first-order rate,

$$\text{BotAlgDeath} = k_{db}(T) c_b \quad (146)$$

where $k_{db}(T)$ = the temperature-dependent bottom algae death rate [/d].

Bottom Algae Internal Nitrogen (IN_b)

The change in intracellular nitrogen in bottom algal cells is calculated from

$$S_{bN} = \text{BotAlgUpN} - q_{Nb}\ \text{BotAlgDeath} - \text{BotAlgExN} \quad (147)$$

where BotAlgUpN = the uptake rate of nitrogen by bottom algae (mgN/m^2/d), BotAlgDeath = bottom algae death (gC/m^2/d), and BotAlgExN = the bottom algae excretion of nitrogen (mgN/m^2/d), which is computed as

$$\begin{aligned} \text{BotAlgExN} &= 0 && \text{if } IN_b \le q_{0Nb} \\ \text{BotAlgExN} &= k_{eb}(T)\left(IN_b - q_{0Nb}\right) && \text{if } IN_b > q_{0Nb} \end{aligned} \quad (148)$$

where $k_{eb}(T)$ = the temperature-dependent bottom algae excretion rate [/d].

The N uptake rate depends on both external and intracellular nutrients as in (Rhee 1973),

$$\text{BotAlgUpN} = \rho_{mNb} \frac{n_a + n_n}{k_{sNb} + n_a + n_n} \frac{K_{qNb}}{K_{qNb} + (q_{Nb} - q_{0Nb})} c_b \quad (149)$$

where ρ_{mNb} = the maximum uptake rate for nitrogen [mgN/gC/d], k_{sNb} = half-saturation constant for external nitrogen [µgN/L] and K_{qNb} = half-saturation constant for intracellular nitrogen [mgN/gC].

Bottom Algae Internal Phosphorus (IP_b)

The change in intracellular phosphorus in bottom algal cells is calculated from

$$S_{bP} = \text{BotAlgUpP} - q_{Pb}\ \text{BotAlgDeath} - \text{BotAlgExP} \quad (150)$$

where BotAlgUpP = the uptake rate of phosphorus by bottom algae (mgP/m^2/d), BotAlgDeath = bottom algae death (gC/m^2/d), and BotAlgExP = the bottom algae excretion of phosphorus (mgP/m^2/d), which is computed as

$$\begin{aligned} \text{BotAlgExP} &= 0 && \text{if } IP_b \le q_{0Pb} \\ \text{BotAlgExP} &= k_{eb}(T)\left(IP_b - q_{0Pb}\right) && \text{if } IP_b > q_{0Pb} \end{aligned} \quad (151)$$

where $k_{eb}(T)$ = the temperature-dependent bottom algae excretion rate [/d].

The P uptake rate depends on both external and intracellular nutrients as in (Rhee 1973),

$$\text{BotAlgUpP} = \rho_{mPb} \frac{p_i}{k_{sPb} + p_i} \frac{K_{qPb}}{K_{qPb} + (q_{Pb} - q_{0Pb})} c_b \quad (152)$$

where ρ_{mPb} = the maximum uptake rate for phosphorus [mgP/gC/d], k_{sPb} = half-saturation constant for external phosphorus [μgP/L] and K_{qPb} = half-saturation constant for intracellular phosphorus [mgP gC^{-1}].

Particulate Organic Carbon (c_p)

Detrital carbon or particulate organic carbon (POC) increases due to plant death. It is lost via dissolution and settling

$$S_{co} = \text{PhytoDeath} + \frac{\text{BotAlgDeath}}{H} - \text{POCDiss} - \text{POCSettl} \quad (153)$$

where

$$\text{POCDiss} = k_{dt}(T) c_p \quad (154)$$

where $k_{poc}(T)$ = the temperature-dependent POC dissolution rate [/d] and

$$\text{POCSettl} = \frac{v_{poc}}{H} c_p \quad (155)$$

where v_{poc} = POC settling velocity [m/d].

Slowly Reacting CBOD (c_s)

Slowly reacting CBOD increases due to POC dissolution. It is lost via hydrolysis and oxidation,

$$S_{cs} = (1 - F_f) r_{oc} \text{ POCDiss} - \text{SlowCHydr} - \text{SlowCOxid} \quad (156)$$

where F_f = the fraction of detrital dissolution that goes to fast reacting CBOD [dimensionless], and

$$\text{SlowCHydr} = k_{hc}(T) c_s \quad (157)$$

where $k_{hc}(T)$ = the temperature-dependent slow CBOD hydrolysis rate [/d], and

$$\text{SlowCOxid} = F_{oxc}\ k_{dcs}(T) c_s \quad (158)$$

where $k_{dcs}(T)$ = the temperature-dependent slow CBOD oxidation rate [/d] and F_{oxc} = attenuation due to low oxygen [dimensionless].

Fast Reacting CBOD (c_f)

Fast reacting CBOD is gained via the dissolution of POC and the hydrolysis of slowly-reacting CBOD. It is lost via oxidation and denitrification.

$$S_{cf} = F_f r_{oc}\text{POCDiss} + \text{SlowCHydr} - \text{FastCOxid} - r_{ondn}\text{Denitr} \quad (159)$$

where

$$\text{FastCOxid} = F_{oxc}\ k_{dc}(T)c_f \quad (160)$$

where $k_{dc}(T)$ = the temperature-dependent fast CBOD oxidation rate [/d] and F_{oxc} = attenuation due to low oxygen [dimensionless]. The parameter r_{ondn} is the ratio of oxygen equivalents lost per nitrate nitrogen that is denitrified (Eq. 67). The term Denitr is the rate of denitrification [μgN/L/d]. It will be defined in the section on nitrate below.

Three formulations are used to represent the oxygen attenuation:

Half-Saturation:

$$F_{oxrp} = \frac{o}{K_{socf} + o} \quad (161)$$

where K_{socf} = half-saturation constant for the effect of oxygen on fast CBOD oxidation [mgO_2/L].

Exponential:

$$F_{oxrp} = (1 - e^{-K_{socf}o}) \quad (162)$$

where K_{socf} = exponential coefficient for the effect of oxygen on fast CBOD oxidation [L/mgO_2].

Second-Order Half Saturation:

$$F_{oxrp} = \frac{o^2}{K_{socf} + o^2} \quad (163)$$

where K_{socf} = half-saturation constant for second-order effect of oxygen on fast CBOD oxidation [mgO_2^2/L^2].

Particulate Organic Nitrogen (n_{op})

Particulate organic nitrogen (PON) increases due to plant death. It is lost via PON dissolution and settling.

$$S_{nop} = q_{Np}\text{PhytoDeath} + q_{Nb}\frac{\text{BotAlgDeath}}{H} - \text{PONDiss} - \text{PONSettl} \quad (164)$$

The rate of organic nitrogen dissolution is computed as

$$\text{PONDiss} = k_{dn}(T)n_{op} \quad (165)$$

where $k_{dn}(T)$ = the temperature-dependent PON dissolution rate [/d]. PON settling is determined as

$$\text{PONSettl} = \frac{v_{pon}}{H}n_{op} \quad (166)$$

where v_{pon} = organic nitrogen settling velocity [m/d].

Dissolved Organic Nitrogen (n_{od})

Organic nitrogen increases due to PON dissolution. It is lost via hydrolysis and settling.

$$S_{nod} = \text{PONDiss} - \text{DONHydr} \quad (167)$$

The rate of organic nitrogen hydrolysis is computed as

$$\text{DONHydr} = k_{hn}(T) n_{od} \quad (168)$$

where $k_{hn}(T)$ = the temperature-dependent organic nitrogen hydrolysis rate [/d].

Ammonia Nitrogen (n_a)

Ammonia nitrogen increases due to dissolved organic nitrogen hydrolysis and plant death and excretion. It is lost via nitrification and plant photosynthesis:

$$S_{na} = \text{DONHydr} + \text{PhytoExN} + \frac{\text{BotAlgExN}}{H} - \text{Nitrif} - P_{ap}\ \text{PhytoUpN} - P_{ab}\ \frac{\text{BotAlgUpN}}{H} - \text{NH3GasLoss} \quad (169)$$

The ammonia nitrification rate is computed as

$$\text{Nitrif} = F_{oxna} k_n(T) n_a \quad (170)$$

where $k_n(T)$ = the temperature-dependent nitrification rate for ammonia nitrogen [/d] and F_{oxna} = attenuation due to low oxygen [dimensionless]. Oxygen attenuation is modeled by Eqs. (127) to (129) with the oxygen dependency represented by the parameter K_{sona}.

The coefficients P_{ap} and P_{ab} are the preferences for ammonium as a nitrogen source for phytoplankton and bottom algae, respectively,

$$P_{ap} = \frac{n_a n_n}{(k_{hnxp} + n_a)(k_{hnxp} + n_n)} + \frac{n_a k_{hnxp}}{(n_a + n_n)(k_{hnxp} + n_n)} \quad (171)$$

$$P_{ab} = \frac{n_a n_n}{(k_{hnxb} + n_a)(k_{hnxb} + n_n)} + \frac{n_a k_{hnxb}}{(n_a + n_n)(k_{hnxb} + n_n)} \quad (172)$$

where k_{hnxp} = preference coefficient of phytoplankton for ammonium [mgN/m^3] and k_{hnxb} = preference coefficient of bottom algae for ammonium [mgN/m^3].

As depicted in Figure 22, when the ammonium and nitrate concentrations are at the preference coefficient, both are equally preferred (k_{hnx} = 0.5). When the ammonium ion is high, it is preferred by the algae. When ammonium is low and nitrate is abundant, the algae prefer nitrate.

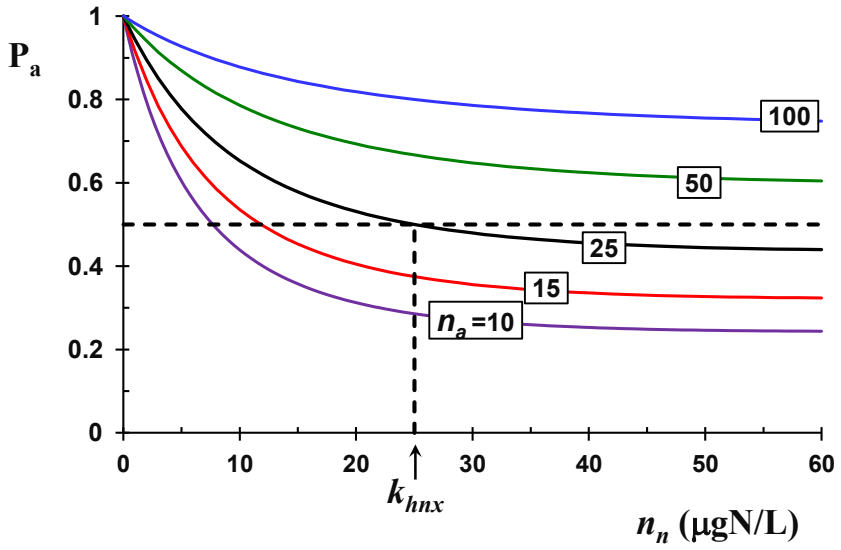

Figure 22 Ammonium preference (Pa) versus nitrate and ammonium concentrations for different levels of ammonium for a preference factor of k_{hnx} = 20 µgN/L.

Unionized Ammonia

The model simulates total ammonia. In water, the total ammonia consists of two forms: ammonium ion, NH_4^+, and unionized ammonia, NH_3. At normal pH (6 to 8), most of the total ammonia will be in the ionic form. However at high pH, unionized ammonia predominates. The amount of unionized ammonia can be computed as

$$n_{au} = F_u n_a \tag{173}$$

where n_{au} = the concentration of unionized ammonia [µgN/L], and F_u = the fraction of the total ammonia that is in unionized form,

$$F_u = \frac{K_a}{10^{-\mathrm{pH}} + K_a} \tag{174}$$

where K_a = the equilibrium coefficient for the ammonia dissociation reaction, which is related to temperature by

$$\mathrm{p}K_a = 0.09018 + \frac{2729.92}{T_a} \tag{175}$$

where T_a is absolute temperature [K] and $\mathrm{p}K_a = -\log_{10}(K_a)$. Note that the fraction of the total ammonia that is in ionized form, F_i, can be computed as $1 - F_u$ or

$$F_i = \frac{10^{-\mathrm{pH}}}{10^{-\mathrm{pH}} + K_a} \tag{176}$$

Ammonia Gas Transfer

The loss of ammonia via gas transfer is computed as

$$\text{NH3GasLoss} = \frac{v_{nh3}(T)}{H}\left(n_{aus}(T) - n_{au}\right)$$

where $v_{nh3}(T)$ = the temperature-dependent ammonia gas-transfer coefficient [m/d], and $n_{aus}(T)$ = the saturation concentration of ammonia [μgN/L] at temperature, T.

The transfer coefficient is calculated as

$$v_{nh3} = K_l \frac{H_e}{H_e + RT_a\left(\frac{K_l}{K_g}\right)}$$

where v_v = the mass-transfer coefficient (m/d), K_l and K_g = liquid and gas film exchange coefficients [m/d], respectively, R = the universal gas constant (= 8.206×10^{-5} atm m^3/(K mole)), T_a = absolute temperature [K], and H_e = Henry's constant (atm m^3/mole).

The saturation concentration is calculated as

$$n_{aus}(T) = \frac{p_{nh3}}{H_e} \times CF$$

where p_{nh3} = the partial pressure of ammonia in the atmosphere (atm), and CF is a conversion factor (μgN/L per moleNH$_3$/m^3). The partial pressure of ammonia ranges from 1-10 ppb in rural and moderately polluted areas to 10-100 ppb in heavily polluted areas (Holland 1978, Finlayson-Pitts and Pitts 1986). We will assume that a value of 2 ppb, which corresponds to 2×10^{-9} atm, represents a typical value. The conversion factor is

$$CF = \frac{\text{m}^3}{1000\text{L}} \times \frac{14\text{gN}}{\text{moleNH}_3} \times \frac{10^6\mu\text{gN}}{\text{gN}} = 14\times10^3 \frac{\mu\text{gN/L}}{\text{moleNH}_3/\text{m}^3}$$

The liquid-film coefficient can be related to the oxygen reaeration rate by (Mills et al. 1982),

$$K_l = k_a H\left[\frac{32}{17}\right]^{0.25} = 1.171\times k_a H$$

The gas-film coefficient is computed by

$$K_g = K_{g,\text{H2O}}\left[\frac{18}{17}\right]^{0.25}$$

where $K_{g,\text{H2O}}$ = a mass-transfer velocity for water vapor [m/d], which can be related to wind speed by (Schwarzenbach et al. 1993)

$$K_{g,\text{H2O}} = \left(0.2U_{w,10} + 0.3\right) \times \frac{\text{m}}{100\text{cm}} \times \frac{86{,}400\text{s}}{\text{d}}$$

where $U_{w,10}$ = wind speed at a height of 10 m (m/s). Combining the above equations gives

$$K_g = 175.287U_{w,10} + 262.9305$$

The Henry's constant for ammonia at 20°C is 1.3678×10^{-5} atm m^3/mole (Kavanaugh and Trussell 1980). The value at temperatures other than 20°C can be computed with

$$H_e(T) = H_e(20)\times1.052^{T-20}$$

Nitrate Nitrogen (n_n)

Nitrate nitrogen increases due to nitrification of ammonia. It is lost via denitrification and plant uptake:

$$S_{ni} = \text{Nitrif} - \text{Denitr} - (1-P_{ap})\,\text{PhytoUpN} - (1-P_{ab})\,\frac{\text{BotAlgUpN}}{H} \quad (177)$$

The denitrification rate is computed as

$$\text{Denitr} = (1-F_{oxdn})k_{dn}(T)n_n \quad (178)$$

where $k_{dn}(T)$ = the temperature-dependent denitrification rate of nitrate nitrogen [/d] and F_{oxdn} = effect of low oxygen on denitrification [dimensionless] as modeled by Eqs. (127) to (129) with the oxygen dependency represented by the parameter K_{sodn}.

Particulate Organic Phosphorus (p_{op})

Particulate organic phosphorus (POP) increases due to plant death. It is lost via POP dissolution and settling.

$$S_{po} = q_{Pp}\,\text{PhytoDeath} + q_{Pb}\,\frac{\text{BotAlgDeath}}{H} - \text{POPDiss} - \text{POPSettl} \quad (179)$$

The rate of organic phosphorus dissolution is computed as

$$\text{POPDiss} = k_{dp}(T)p_{op} \quad (180)$$

where $k_{dp}(T)$ = the temperature-dependent POP dissolution rate [/d]. Settling of PON is calculated as

$$\text{POPSettl} = \frac{v_{pop}}{H}p_{op} \quad (181)$$

where v_{pop} = particulate organic phosphorus settling velocity [m/d].

Dissolved Organic Phosphorus (p_{od})

Dissolved organic phosphorus (DOP) increases due to POP dissolution. It is lost via hydrolysis and settling.

$$S_{pod} = \text{POPDiss} - \text{DOPHydr} \quad (182)$$

The rate of organic phosphorus hydrolysis is computed as

$$\text{DOPHydr} = k_{hp}(T)p_{od} \quad (183)$$

where $k_{hp}(T)$ = the temperature-dependent organic phosphorus hydrolysis rate [/d].

Inorganic Phosphorus (p_i)

Inorganic phosphorus increases due to dissolved organic phosphorus hydrolysis and plant excretion. It is lost via plant uptake. In addition, a settling loss is included for cases in which inorganic phosphorus is lost due to sorption onto settleable particulate matter such as iron oxyhydroxides:

$$S_{pi} = \text{DOPHydr} + \text{PhytoExP} + \frac{\text{BotAlgExP}}{H} - \text{PhytoUpP} - \frac{\text{BotAlgUpP}}{H} - \text{IPSettl} \quad (184)$$

where

$$\text{IPSettl} = \frac{v_{ip}}{H} p_i \quad (185)$$

where v_{ip} = inorganic phosphorus mass transfer coefficient to the sediments [m/d].

Inorganic Suspended Solids (m_i)

Inorganic suspended solids are lost via settling,

$S_{mi} = -$ InorgSettl

where

$$\text{InorgSettl} = \frac{v_i}{H} m_i \quad (186)$$

where v_i = inorganic suspended solids settling velocity [m/d].

Dissolved Oxygen (o)

Dissolved oxygen increases due to plant photosynthesis and nitrate uptake. It is lost via fast CBOD oxidation, nitrification, and plant respiration. Depending on whether the water is undersaturated or oversaturated it is gained or lost via reaeration,

$$\begin{aligned} S_o = {} & r_{oc}\text{PhytoPhoto} + r_{on}(1-P_{ap})\text{PhytoUptakeN} - r_{oc}\text{PhytoResp} \\ & + r_{oc}\frac{\text{BotAlgPhoto}}{H} + r_{on}(1-P_{ab})\frac{\text{BotAlgUptakeN}}{H} - r_{oc}\frac{\text{BotAlgResp}}{H} \\ & - r_{oc}\left(\text{FastCOxid} + \text{SlowCOxid}\right) - r_{on}\text{NH4Nitr} + \text{OxReaer} \end{aligned} \quad (187)$$

where

$$\text{OxReaer} = k_a(T)\left(o_s(T, elev) - o\right) \quad (188)$$

where $k_a(T)$ = the temperature-dependent oxygen reaeration coefficient [/d], $o_s(T, elev)$ = the saturation concentration of oxygen [mgO_2/L] at temperature, T, and elevation above sea level, $elev$.

Oxygen Saturation

For freshwater systems, dissolved oxygen saturation depends on temperature, dissolved solids, and oxygen partial pressure. These factors influence saturation as described by (APHA 2017),

$$o_s(T, elev, S) = \phi_{elev} \cdot \phi_S \cdot e^{\ln o_{sf}(T)} \quad (189)$$

where o_s = dissolved oxygen saturation concentration (mgO_2/L), ϕ_{elev} = the fractional reduction of freshwater sea-level saturation due to elevation above sea level (dimensionless), ϕ_S = the fractional reduction of freshwater sea-level saturation due to salinity (dimensionless), and o_{sf} = the dissolved oxygen saturation concentration of sea-level freshwater (mgO_2/L). The individual effects of temperature, salinity, and elevation are quantified as follows.

Temperature, T (°C): The oxygen saturation of freshwater at sea level is determined by evaluating the exponent of the exponential function of Eq. (2) with (APHA 2017),

$$\ln o_{sf}(T) = -139.34411 + \frac{1.575701\times10^5}{(T+273.15)} - \frac{6.642308\times10^7}{(T+273.15)^2} + \frac{1.243800\times10^{10}}{(T+273.15)^3} - \frac{8.621949\times10^{11}}{(T+273.15)^4} \quad \textbf{(190)}$$

Salinity, S (ppt):

The oxygen saturation of sea water is obtained by multiplying the sea-level freshwater saturation by (APHA 2017)

$$\phi_S = e^{-S\left(1.7674\times10^{-2} + \frac{10.754}{T_a} - \frac{2140.7}{T_a^2}\right)} \quad \textbf{(191)}$$

Elevation, *elev* (km): The effect of atmospheric pressure on gas saturation at elevation is based on the standard atmosphere as described by the cubic polynomial (Camacho and Chapra 2020):

$$\phi_{elev} = 1 - 0.11988\ elev + 6.10834\times10^{-3}\ elev^2 - 1.60747\times10^{-4}\ elev^3 \quad \textbf{(192)}$$

Reaeration Formulas

The reaeration coefficient (at 20 °C) can be prescribed on the **Reach Worksheet**. If reaeration is not prescribed (that is, it is blank or zero for a particular reach), it is computed as a function of the river's hydraulics and (optionally) wind velocity,

$$k_a(20) = k_{ah}(20) + \frac{K_{Lw}(20)}{H} \quad \textbf{(193)}$$

where $k_{ah}(20)$ = the reaeration rate at 20°C computed based on the river's hydraulic characteristics [d^{-1}], $K_L(20)$ = the reaeration mass-transfer coefficient based on wind velocity [m/d], and H = mean depth [m].

Hydraulic-based Formulas:

O'Connor-Dobbins (O'Connor and Dobbins 1958):

$$k_{ah}(20) = 3.93\frac{U^{0.5}}{H^{1.5}} \quad \textbf{(194)}$$

where U = mean water velocity [m/s] and H = mean water depth [m].

Churchill (Churchill et al. 1962):

$$k_{ah}(20) = 5.026\frac{U}{H^{1.67}} \quad \textbf{(195)}$$

Owens-Gibbs (Owens et al. 1964):

$$k_{ah}(20) = 5.32\frac{U^{0.67}}{H^{1.85}} \quad \textbf{(196)}$$

Tsivoglou-Neal (Tsivoglou and Neal 1976):

Low flow, Q = 0.0283 to 0.4247 cms (1 to 15 cfs):

$$k_{ah}(20) = 31{,}183\ US \quad (197)$$

High flow, Q = 0.4247 to 84.938 cms (15 to 3000 cfs):

$$k_{ah}(20) = 15{,}308\ US \quad (198)$$

where S = channel slope [m/m].

Thackston-Dawson (Thackston and Dawson 2001):

$$k_{ah}(20) = 2.16(1 + 9F^{0.25})\frac{U_*}{H} \quad (199)$$

where U* = shear velocity [m/s], and F = the Froude number [dimensionless]. The shear velocity and the Froude number are defined as

$$U_* = \sqrt{gR_hS} \quad (200)$$

and

$$F = \frac{U}{\sqrt{gH_d}} \quad (201)$$

where g = gravitational acceleration (= 9.81 m/s^2), R_h = hydraulic radius [m], S = channel slope [m/m], and H_d = the hydraulic depth [m]. The hydraulic depth is defined as

$$H_d = \frac{A_c}{B_t} \quad (202)$$

where B_t = the top width of the channel [m].

USGS (Pool-riffle) (Melching and Flores 1999):

Low flow, Q < 0.556 cms (< 19.64 cfs):

$$k_{ah}(20) = 517(US)^{0.524}Q^{-0.242} \quad (203)$$

High flow, Q > 0.556 cms (> 19.64 cfs):

$$k_{ah}(20) = 596(US)^{0.528}Q^{-0.136} \quad (204)$$

where Q = flow (cms).

USGS (Channel-control) (Melching and Flores 1999):

Low flow, Q < 0.556 cms (< 19.64 cfs):

$$k_{ah}(20) = 88(US)^{0.313}H^{-0.353} \quad (205)$$

High flow, Q > 0.556 cms (> 19.64 cfs):

$$k_{ah}(20) = 142(US)^{0.333}H^{-0.66}B_t^{-0.243} \quad (206)$$

Internal (Covar 1976):

Reaeration can also be internally calculated based on the following scheme patterned after a plot developed by Covar (1976) (Figure 23):

- If H < 0.61 m, use the Owens-Gibbs formula

- If $H > 0.61$ m and $H > 3.45U^{2.5}$, use the O'Connor-Dobbins formula
- Otherwise, use the Churchill formula

This is referred to as option **Internal** on the **Rates Worksheet** of Q2K. Note that if no option is specified, the Internal option is the default.

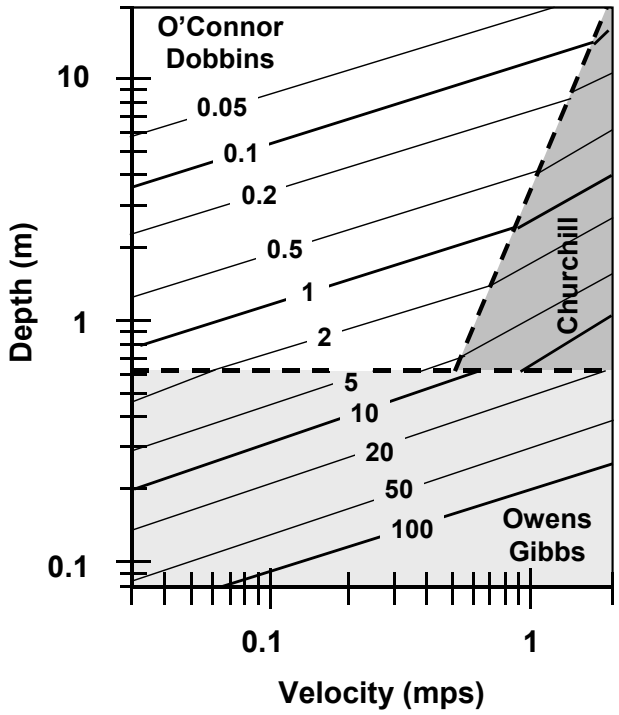

Figure 23 Reaeration rate (/d) versus depth and velocity (Covar 1976).

Wind-based Formulas:

Three options are available to incorporate wind effects into the reaeration rate: (1) it can be omitted, (2) the Banks-Herrera formula and (3) the Wanninkhof formula.

Banks-Herrera formula (Banks 1975, Banks and Herrera 1977):

$$K_{lw} = 0.728U_{w,10}^{0.5} - 0.317U_{w,10} + 0.0372U_{w,10}^{2} \quad (207)$$

where $U_{w,10}$ = wind speed measured 10 meters above the water surface [m s^{-1}]

Wanninkhof formula (Wanninkhof 1991):

$$K_{lw} = 0.0986U_{w,10}^{1.64} \quad (208)$$

Note that the model uses Eq. (48) to correct the wind velocity entered on the **Meteorology Worksheet** (7 meters above the surface) so that it is scaled to the 10-m height.

Effect of Control Structures: Oxygen

Oxygen transfer in streams is influenced by the presence of control structures such as weirs, dams, locks, and waterfalls (Figure 24).

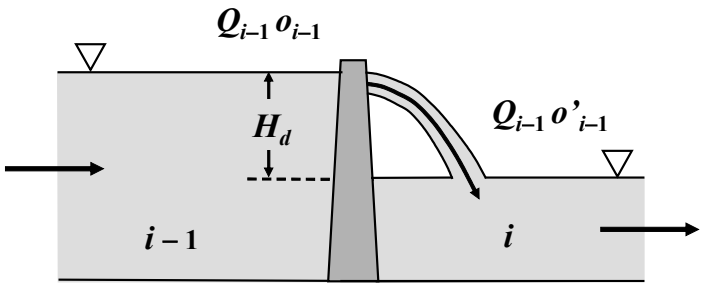

Figure 24 Water flowing over a river control structure.

Butts and Evans (1983) have reviewed efforts to characterize this transfer and have suggested the following formula,

$$r_d = 1 + 0.38 a_d b_d H_d (1 - 0.11 H_d)(1 + 0.046T) \quad \textbf{(209)}$$

where r_d = the ratio of the deficit above and below the dam, H_d = the difference in water elevation [m] as calculated with Eq. (7), T = water temperature (°C) and a_d and b_d are coefficients that correct for water-quality and dam-type. Values of a_d and b_d are summarized in Table 7. If no values are specified, Q2K uses the following default values for these coefficients: a_d = 1.25 and b_d = 0.9.

Table 7 Coefficient values used to predict the effect of dams on stream reaeration.

(*a*) **Water-quality coefficient**	
Polluted state	a_d
Gross	0.65
Moderate	1.0
Slight	1.6
Clean	1.8
(*b*) **Dam-type coefficient**	
Dam type	b_d
Flat broad-crested regular step	0.70
Flat broad-crested irregular step	0.80
Flat broad-crested vertical face	0.60
Flat broad-crested straight-slope face	0.75
Flat broad-crested curved face	0.45
Round broad-crested curved face	0.75
Sharp-crested straight-slope face	1.00
Sharp-crested vertical face	0.80
Sluice gates	0.05

The oxygen mass balance for the element below the structure is written as

$$\frac{do_i}{dt} = \frac{Q_{i-1}}{V_i} o'_{i-1} - \frac{Q_i}{V_i} o_i - \frac{Q_{ab,i}}{V_i} o_i + \frac{E'_i}{V_i}(o_{i+1} - o_i) + \frac{W_{o,i}}{V_i} + S_{o,i} \quad \textbf{(210)}$$

where o'_{i-1} = the oxygen concentration entering the element [mgO_2/L], where

$$o'_{i-1} = o_{s,i-1} - \frac{o_{s,i-1} - o_{i-1}}{r_d} \quad \textbf{(211)}$$

pH

The following equilibrium, mass balance and electroneutrality equations define a freshwater dominated by inorganic carbon (Stumm and Morgan 1996),

$$K_1 = \frac{\{HCO_3^-\}\{H^+\}}{\{H_2CO_3^*\}} \quad (212)$$

$$K_2 = \frac{\{CO_3^{2-}\}\{H^+\}}{\{HCO_3^-\}} \quad (213)$$

$$K_w = \{H^+\}\{OH^-\} \quad (214)$$

$$c_T = [H_2CO_3^*] + [HCO_3^-] + [CO_3^{2-}] \quad (215)$$

$$Alk = [HCO_3^-] + 2[CO_3^{2-}] + [OH^-] - [H^+] \quad (216)$$

where K_1, K_2 and K_w are acidity constants, Alk = alkalinity [eq L^{-1}], H_2CO_3* = the sum of dissolved carbon dioxide and carbonic acid, HCO_3^- = bicarbonate ion, CO_3^{2-} = carbonate ion, H^+ = hydronium ion, OH^- = hydroxyl ion, and c_T = total inorganic carbon concentration [mole L^{-1}]. The square brackets [] designate molar concentrations and the curly brackets {} designate activities.

The model state variables are listed in Table 5. Note that concentrations are expressed in a variety of units. In addition, for equilibrium calculations, the concentration of ions are also expressed as activities with

$$\{X\} = \gamma(z)[X] \quad (217)$$

where {X} = activity of ion X (M), $\gamma(z)$ = activity coefficient, z = charge, and [X] = concentration (M). The activity coefficient is computed with the Davies equation (Davies 1962, Allison et al. 1991),

$$\gamma(z) = 10^{-0.5z^2\left(\frac{\sqrt{IS}}{1+\sqrt{IS}} - 0.24IS\right)} \quad (218)$$

The activity of an ion, {X}, can be determined by multiplying its molar concentration [X] by an activity coefficient,

$$\{X\} = \gamma[X] \quad (219)$$

where the activity coefficient γ is less than or equal to one.

To compute the activity of a solution, the strength of the ionic interactions must be quantified. Russell (1976) suggested that the ionic strength could be estimated on the basis of the specific conductance, s, as

$$\mu = 1.6 \times 10^{-5} s \quad (220)$$

where specific conductance must be measured in μmho cm^{-1}.

Once the ionic strength is quantified, a number of equations are available for determining the activity coefficient. One, which holds for solutions with ionic strengths less than approximately 5×10^{-3}, is the *Debye-Hückel limiting law*,

$$\gamma = 10^{-0.5 Z_i^2 \sqrt{\mu}} \quad (221)$$

where Z_i = the charge of species i. Note that according to Eq. (185), an ionic strength of less than approximately 5×10^{-3} corresponds to a specific conductance, s = $5\times10^{-3}/1.6\times10^{-5}$ = 312.5 µmho cm^{-1}.

```
lam1 = 10.0_DP ** (-0.5_DP * 1.0_DP * SQRT(mu))
  lam2 = 10.0_DP ** (-0.5_DP * 2.0_DP ** 2 * SQRT(mu))
```

*lam1 = 10.0_DP ** (-0.5_DP * SQRT(mu))*
*lam2 = 10.0_DP ** (-2.0_DP * SQRT(mu))*

3, is called the Debye-Hiickel limiting law,

lo g y ; = - o.szf Ju (37.12)

An alternative equation is also available for more concentrated solutions (ionic strength < 0.1). Called the extended Debye-Hiickel approximation, it has the form,

Note that the alkalinity is expressed in units of eq/L for the internal calculations. For input and output, it is expressed as $mgCaCO_3/L$. The two units are related by

$$Alk(\text{mgCaCO}_3/\text{L}) = 50{,}000 \times Alk(\text{eq/L}) \quad (222)$$

The equilibrium constants are corrected for temperature by

```
ELEMENTAL SUBROUTINE ChemRates(Te, K1, K2, KW, Kh, Cond)

  !input
  REAL(DP), INTENT(IN) :: Te, Cond
  !output
  REAL(DP), INTENT(OUT) :: K1, K2, KW, Kh
  !local
  REAL(DP) Ta, mu, lam1, lam2

  Ta = Te + 273.15_DP
  mu = 0.000016_DP * Cond

  lam1 = 10.0_DP ** (-0.5_DP * 1.0_DP * SQRT(mu))
  lam2 = 10.0_DP ** (-0.5_DP * 2.0_DP ** 2 * SQRT(mu))

  K1 = -356.3094_DP - 0.06091964_DP * Ta + 21834.37_DP / Ta + 126.8339_DP *
LOG(Ta) / LOG(10.0_DP)
  K1 = K1 - 1684915.0_DP / Ta ** 2
  K1 = 10.0_DP ** K1 / lam1 / lam1
  K2 = -107.8871_DP - 0.03252849_DP * Ta + 5151.79_DP / Ta + 38.92561_DP *
LOG(Ta) / LOG(10.0_DP)
  K2 = K2 - 563713.9_dp / Ta ** 2
  K2 = 10.0_DP ** K2 / lam2

  KW = 10.0_DP ** (6.0875_DP - 0.01706_DP * Ta - 4470.99_DP / Ta) / lam1 /
lam1
  Kh = 10.0_dp ** -(-2385.73_dp / Ta - 0.0152642_dp * Ta + 14.0184_dp)

END SUBROUTINE
```

Harned and Hamer (1933):

$$\mathrm{p}K_w = \frac{4787.3}{T_a} + 7.1321\ \log_{10}(T_a) + 0.010365T_a - 22.80 \quad (223)$$

Plummer and Busenberg (1982):

$$\log K_1 = -356.3094 - 0.06091964T_a + 21834.37 / T_a + 126.8339\log T_a - 1{,}684{,}915 / T_a^2 \quad (224)$$

Plummer and Busenberg (1982):

$$\log K_2 = -107.8871 - 0.03252849T_a + 5151.79 / T_a + 38.92561\log T_a - 563{,}713.9 / T_a^2 \quad (225)$$

The nonlinear system of five simultaneous equations (177 through 181) can be solved numerically for the five unknowns: [H_2CO_3*], [HCO_3^-], [CO_3^{2-}], [OH^-], and {H^+}. An efficient solution method can be derived by combining Eqs. (177), (178) and (180) to define the quantities (Stumm and Morgan 1996)

$$\alpha_0 = \frac{[\mathrm{H}^+]^2}{[\mathrm{H}^+]^2 + K_1[\mathrm{H}^+] + K_1K_2} \quad (226)$$

$$\alpha_1 = \frac{K_1[\mathrm{H}^+]}{[\mathrm{H}^+]^2 + K_1[\mathrm{H}^+] + K_1K_2} \quad (227)$$

$$\alpha_2 = \frac{K_1K_2}{[\mathrm{H}^+]^2 + K_1[\mathrm{H}^+] + K_1K_2} \quad (228)$$

where α_0, α_1, and α_2 = the fraction of total inorganic carbon in carbon dioxide, bicarbonate, and carbonate, respectively. Equations (179), (187), and (188) can be substituted into Eq. (181) to yield,

$$\mathrm{Alk} = (\alpha_1 + 2\alpha_2)c_T + \frac{K_w}{[\mathrm{H}^+]} - [\mathrm{H}^+] \quad (229)$$

Thus, solving for pH reduces to determining the root, {H^+}, of

$$f([\mathrm{H}^+]) = (\alpha_1 + 2\alpha_2)c_T + \frac{K_w}{[\mathrm{H}^+]} - [\mathrm{H}^+] - \mathrm{Alk} \quad (230)$$

where pH is then calculated with

$$\mathrm{pH} = -\log_{10}[\mathrm{H}^+] \quad (231)$$

The root of Eq. (190) is determined with a numerical method. The user can choose either bisection, Newton-Raphson or Brent's method (Chapra and Canale 2006, Chapra 2007) as specified on the QUAL2K sheet. The Newton-Raphson is the fastest but can sometimes diverge. In contrast, the bisection method is slower, but more reliable. Because it balances speed with reliability, Brent's method is the default.

Total Inorganic Carbon (c_T)

Total inorganic carbon concentration increases due to fast carbon oxidation and plant respiration. It is lost via plant photosynthesis. Depending on whether the water is undersaturated or oversaturated with CO_2, it is gained or lost via reaeration,

$$\begin{aligned} S_{cT} = {} & r_{ccc}\text{FastCOxid} + \text{CO2Reaer} \\ & + r_{ccc}\left(\text{Phyto1Resp} + \text{Phyto2Resp}\right) + r_{ccc}\frac{\left(\text{BotAlg1Resp} + \text{BotAlg2Resp}\right)}{H} \\ & - r_{cc\alpha}\left(\text{Phyto1Photo} + \text{Phyto2Photo}\right) - r_{ccc}\frac{\left(\text{BotAlg1Photo} + \text{BotAlg2Photo}\right)}{H} \end{aligned} \quad \textbf{(232)}$$

where

$$\text{CO2Reaer} = k_{ac}(T)\left([\text{CO}_2]_s - \alpha_0 c_T\right) \quad \textbf{(233)}$$

where $k_{ac}(T)$ = the temperature-dependent carbon dioxide reaeration coefficient [/d], and $[CO_2]_s$ = the saturation concentration of carbon dioxide [mole/L]. The stoichiometric coefficient is computed as[5]

$$r_{ccc} = \frac{\text{moleC}}{12\ \text{gC}} \times \frac{\text{m}^3}{1000\ \text{L}} \quad \textbf{(234)}$$

Carbon Dioxide Saturation

The CO_2 saturation is computed with Henry's law,

$$[\text{CO}_2]_s = K_H \text{p}_{\text{CO}_2} \quad \textbf{(235)}$$

where K_H = Henry's constant [mole (L atm)$^{-1}$] and p_{CO_2} = the partial pressure of carbon dioxide in the atmosphere [atm]. Note that the partial pressure is input on the **Rates Worksheet** in units of ppm. The program internally converts ppm to atm using the conversion: 10^{-6} atm/ppm.

The value of K_H can be computed as a function of temperature by (Edmond and Gieskes 1970)

$$\text{p}K_{\text{H}} = -\frac{2385.73}{T_a} - 0.0152642 T_a + 14.0184 \quad \textbf{(236)}$$

The partial pressure of CO_2 in the atmosphere has been increasing, largely due to the combustion of fossil fuels (Figure 25). The mean monthly value in June 2021 is approximately $10^{-2.62}$ atm (= 417.4 ppm) which represents an increase of over 100 ppm since 1958 when the Mauna Loa measurements were started.

[5] The conversion, m^3 = 1000 L is included because all mass balances express volume in m^3, whereas total inorganic carbon is expressed as mole/L.

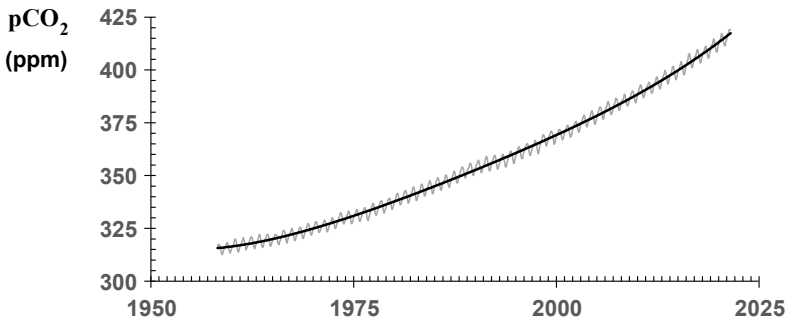

Figure 25 Concentration of carbon dioxide in the atmosphere as recorded at Mauna Loa Observatory, Hawaii.[6]

We have fit a 4th order polynomial with a base year of 2000 to the monthly data

$$\begin{aligned} p_{CO_2} &= 369.222 + 1.7584\left(t-2000\right) + 1.2782\times10^{-2}\left(t-2000\right)^2 \\ &\quad + 3.1481\times10^{-4}\left(t-2000\right)^3 + 6.7550\times10^{-6}\left(t-2000\right)^4 \end{aligned} \tag{237}$$

with an adjusted $R^2 = 0.9946$, and a standard error of 2.17 ppm.

CO_2 Gas Transfer

The CO_2 reaeration coefficient can be computed from the oxygen reaeration rate by

$$k_{ac}(20) = \left(\frac{32}{44}\right)^{0.25} = 0.923\ k_a(20) \tag{238}$$

Effect of Control Structures: CO_2

As was the case for dissolved oxygen, carbon dioxide gas transfer in streams can be influenced by the presence of control structures. Q2K assumes that carbon dioxide behaves similarly to dissolved oxygen (recall Sec. 0). Thus, the inorganic carbon mass balance for the element immediately downstream of the structure is written as

$$\frac{dc_{T,i}}{dt} = \frac{Q_{i-1}}{V_i}c'_{T,i-1} - \frac{Q_i}{V_i}c_{T,i} - \frac{Q_{ab,i}}{V_i}c_{T,i} + \frac{E_i'}{V_i}\left(c_{T,i+1} - c_{T,i}\right) + \frac{W_{cT,i}}{V_i} + S_{cT,i} \tag{239}$$

where $c'_{T,i-1}$ = the concentration of inorganic carbon entering the element [mgO_2/L], where

$$c'_{T,i-1} = (\alpha_1 + \alpha_2)c_{T,i-1} + CO_{2,s,i-1} - \frac{CO_{2,s,i-1} - \alpha_2 c_{T,i-1}}{r_d} \tag{240}$$

where r_d is calculated with Eq. (171).

[6] http://www.esrl.noaa.gov/gmd/ccgg/trends/

Alkalinity (Alk)

Alkalinity[7] is the capacity of water to resist acidification. Chemically, it is the strength of a buffer solution composed of weak acids and their conjugate bases. It can be derived from a charge balance of the major ions including those of the major nutrients, phosphorus, and nitrogen:

$$2\{Ca^{2+}\} + 2\{Mg^{2+}\} + 2\{K^{+}\} + 2\{Na^{+}\} + \{NH_4^{+}\} + \{H^{+}\} =$$
$$\{HCO_3^{-}\}^{+} + 2\{CO_3^{2-}\} + \{OH^{-}\} + \{Cl^{-}\} + 2\{SO_4^{2-}\} + 3\{PO_4^{3-}\} + 2\{HPO_4^{2-}\} + \{H_2PO_4^{-}\} + \{NO_3^{-}\} \quad (241)$$

This equation can be rearranged to place the ions due to strong acids and bases on the left of the equal sign and the ions due to the weak acids and bases on the right:

$$2\{Ca^{2+}\} + 2\{Mg^{2+}\} + 2\{K^{+}\} + 2\{Na^{+}\} - \{Cl^{-}\} - 2\{SO_4^{2-}\} - \{NO_3^{-}\} =$$
$$\{HCO_3^{-}\}^{+} + 2\{CO_3^{2-}\} + \{OH^{-}\} - \{H^{+}\} - \{NH_4^{+}\} + \{H_2PO_4^{-}\} + 2\{HPO_4^{2-}\} + 3\{PO_4^{3-}\} \quad (242)$$

Based on this rearrangement, alkalinity can be expressed in two ways. First, it can be expressed as the sum of the ions on the right-hand side; that is, the concentrations of the ions of the weak acids and bases

$$Alk = \{HCO_3^{-}\}^{+} + 2\{CO_3^{2-}\} + \{OH^{-}\} - \{H^{+}\} - \{NH_4^{+}\} + \{H_2PO_4^{-}\} + 2\{HPO_4^{2-}\} + 3\{PO_4^{3-}\} \quad (243)$$

In the context of this form, *Alk* is determined by titration.

Second, if we have direct measurements of the ions of the strong acids and bases [that is, those on the left-hand side of Eq. (220)], alkalinity can be computed directly from these values as

$$Alk = 2\{Ca^{2+}\} + 2\{Mg^{2+}\} + \{K^{+}\} + \{Na^{+}\} - \{Cl^{-}\} - 2\{SO_4^{2-}\} - \{NO_3^{-}\} \quad (244)$$

Given the above as background, Q2K accounts for changes in alkalinity as a function of all the processes that induce changes in the ions on the right-hand side of Eq. (220)[8]:

$$S_{Alk} = -r_{AN}\text{Nitrif} + r_{AN}\text{Denitrif} + r_{AP}\text{DOPHydr} + r_{AN}\text{DONHydr} + r_{AP}\text{PhytoUpP} + r_{AN}\text{PhytoUpN} + r_{AN}\text{PhytoExP} + r_{AN}\text{PhytoExN} + r_{AP}\text{BotAlgUpP} + r_{AN}\text{BotAlgUpN} + r_{AP}\text{BotAlgExP} + r_{AN}\text{BotAlgExN} \quad (245)$$

where the following sections describe how the stoichiometric ratios and signs are calculated.

[7] The Arabic word *al-qalī* represented an alkaline material derived from the ashes of plants, specifically plants that grew on salty soils such as glassworts (also known as saltworts). The Arabs used it as an ingredient in making glass and soap. The medieval term al-qalī referred to a mixture mainly consisting of sodium carbonate and potassium carbonate.

[8] Q2K does not include sulfate under the assumption that it is insignificant in most freshwater systems. Thus, changes in the sulfate ion and hydrogen sulfide are not included. This means that the alkalinity and pH will exhibit some error for brackish estuaries and for so called "black water" rivers. The latter refers to highly polluted rivers with significant sulfur loadings that can generate iron sulfides which impart a black color to the water.

Nitrification

According to Eq. (74), nitrification utilizes ammonium. Hence, because a positive ion (NH4+) is taken up and a negative ion is created, the alkalinity is decreased by two equivalent per mole nitrified. This translates to the following term in our alkalinity mass balance,

$$S_{A,\text{Nitrif}} = -r_{A,\text{Nitrif}}\,\text{Nitrif} \tag{246}$$

where

$$r_{A,\text{Nitrif}} = \frac{2\text{ eq}}{\text{mole NH}_4^+}\frac{\text{mole NH}_4^+}{14.007\text{ gN}}\frac{\text{gN}}{10^6\,\mu\text{gN}}\frac{50{,}000\text{ mgCaCO}_3}{1\text{ eq}} = 7.1393\times10^{-3}\frac{\text{mgCaCO}_3}{\mu\text{gN}} \tag{247}$$

Denitrification

According to Eq. (75), denitrification utilizes a mole of nitrate and a mole of hydrogen ion and creates neutral nitrogen gas. Hence, the alkalinity is increased by one equivalent, and the change in alkalinity can be related to the denitrification rate by

$$S_{A,\text{Denitrif}} = -r_{A,\text{Denitrif}}\,\text{Denitrif} \tag{248}$$

where

$$r_{A,\text{Nitrif}} = \frac{1\text{ eq}}{\text{mole NH}_4^+}\frac{\text{mole NH}_4^+}{14.007\text{ gN}}\frac{\text{gN}}{10^6\,\mu\text{gN}}\frac{50{,}000\text{ mgCaCO}_3}{1\text{ eq}} = 3.5696\times10^{-3}\frac{\text{mgCaCO}_3}{\mu\text{gN}} \tag{249}$$

Dissolved Organic P Hydrolysis

Hydrolysis of organic P results in the creation of dissolved inorganic phosphate which is a weak acid known as triprotic phosphoric acid (H_3PO_4). Thus, DIP exists as three ionic species in natural waters, corresponding to the conjugate bases of H_3PO_4. As depicted in Figure 26, depending on the pH, the phosphate will either have 1 (pH ≅ 2 to 7), 2 (pH ≅ 7 to 12), or 3 (pH > 12) negative charges.

Because negative ions of a weak base are being created by hydrolysis of DOP, the alkalinity is decreased by one, two, or three equivalents, respectively. The change in alkalinity can be related to the DOP hydrolysis rate by,

$$S_{A,\text{DOPHydr}} = r_{A,\text{DOPHydr}}\times\left(\alpha_{\text{HPO4}} + 2\alpha_{\text{PO4}}\right)\times\text{DOPHydr}\left(\frac{\mu\text{gP}}{\text{L d}}\right) \tag{250}$$

where

$$r_{AP} = -\frac{1\text{ eq}}{\text{moleP}}\frac{\text{moleP}}{30.974\text{ gP}}\frac{\text{gP}}{10^6\,\mu\text{gP}}\times\frac{50{,}000\text{ mgCaCO}_3}{1\text{ eq}} = 1.6143\times10^{-3}\frac{\text{mgCaCO}_3}{\mu\text{gP}} \tag{251}$$

$$\alpha_{\text{H3PO4}} = \frac{[\text{H}^+]^3}{[\text{H}^+]^3 + K_{p1}[\text{H}^+]^2 + K_{\text{p1}}K_{p2}[\text{H}^+] + K_{\text{p1}}K_{p2}K_{p3}} \tag{252}$$

$$\alpha_{\text{H2PO4}} = \frac{K_{p1}[\text{H}^+]^2}{[\text{H}^+]^3 + K_{p1}[\text{H}^+]^2 + K_{\text{p1}}K_{p2}[\text{H}^+] + K_{\text{p1}}K_{p2}K_{p3}} \tag{253}$$

$$\alpha_{\text{HPO4}} = \frac{K_{\text{p1}}K_{p2}[\text{H}^+]}{[\text{H}^+]^3 + K_{p1}[\text{H}^+]^2 + K_{\text{p1}}K_{p2}[\text{H}^+] + K_{\text{p1}}K_{p2}K_{p3}} \quad (254)$$

$$\alpha_{\text{PO4}} = \frac{K_{\text{p1}}K_{p2}K_{p3}}{[\text{H}^+]^3 + K_{p1}[\text{H}^+]^2 + K_{\text{p1}}K_{p2}[\text{H}^+] + K_{\text{p1}}K_{p2}K_{p3}} \quad (255)$$

where $K_{p1} = 10^{-2.15}$, $K_{p2} = 10^{-7.2}$, and $K_{p3} = 10^{-12.35}$.

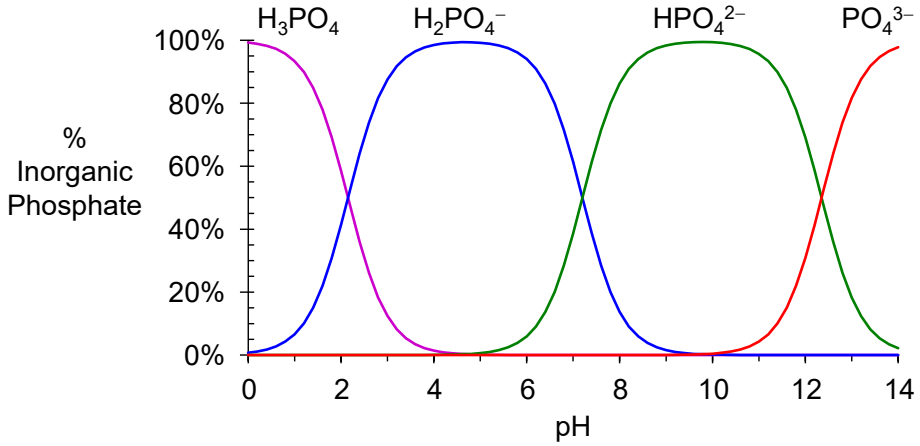

Figure 26 Phosphoric acid speciation

Organic N Hydrolysis

Hydrolysis of organic N results in the creation of ammonia. Depending on the pH, the ammonia will either be in the form of ammonium ion with a single positive charge (pH < 9) or neutral ammonia gas (pH > 9). Hence, when the positive ions are created, the alkalinity is increased by one equivalent. The change in alkalinity can be related to the N hydrolysis rate by

$$S_{A,\text{DONHydr}} = F_i \times r_{A,\text{DONHydr}} \times \text{DONHydr}\left(\frac{\mu\text{gN}}{\text{L d}}\right) \quad (256)$$

where recall that F_i = the fraction of total ammonia that is ionized (i.e., NH_4^+).

Phytoplankton Nitrogen Uptake

Phytoplankton take up nitrogen as either ammonium or nitrate. When ammonium is taken up this leads to a decrease in alkalinity because the ammonium ions are positively charged. When nitrate is taken up this leads to an increase in alkalinity because the nitrate ions are negatively charged. The following representation relates the change in alkalinity due to nitrogen uptake depending on the nitrogen source as well as ammonium speciation as governed by the pH,

$$S_{A,\text{PhytoUpN}} = \left[-P_{ap}F_i + \left(1 - P_{ap}\right)\right] \times r_{AN} \times \text{PhytoUpN}\left(\frac{\mu\text{gN}}{\text{L d}}\right) \quad (257)$$

Ammonium as substrate:

$$106CO_2 + 16NH_4^+ + HPO_4^{2-} + 106H_2O \underset{R}{\overset{P}{\rightleftharpoons}} C_{106}H_{263}O_{110}N_{16}P_1 + 106O_2 + 14H^+ \quad (258)$$

Nitrate as substrate:

$$106CO_2 + 16NO_3^- + HPO_4^{2-} + 122H_2O + 18H^+ \underset{R}{\overset{P}{\rightleftharpoons}} C_{106}H_{263}O_{110}N_{16}P_1 + 138O_2 \quad (259)$$

Phytoplankton Phosphorus Uptake

Phytoplankton take up phosphorus as dissolved inorganic phosphorus (DIP). When DIP is taken up this leads to an increase in alkalinity because the phosphate ions are negatively charged. The following representation relates the change in alkalinity due to DIP uptake depending on inorganic phosphorus speciation as governed by the pH,

$$S_{A,\text{PhytoUpP}} = (\alpha_{\text{H2PO4}} + 2\alpha_{\text{HPO4}} + 3\alpha_{\text{PO4}}) \times r_{AP} \times \text{PhytoUpP}\left(\frac{\mu\text{gP}}{\text{L d}}\right) \quad (260)$$

Phytoplankton Nutrient Excretion

Phytoplankton excrete ammonium and inorganic phosphate. The following representation relates the change in alkalinity to phytoplankton excretion rates including the effect of pH on the nutrient's speciation,

$$\begin{aligned} S_{A,\text{PhytoExN}} &= F_i \times r_{AN} \times \text{PhytoExN}\left(\frac{\mu\text{gN}}{\text{L d}}\right) \\ &\quad - (\alpha_{\text{H2PO4}} + 2\alpha_{\text{HPO4}} + 3\alpha_{\text{PO4}}) \times r_{AP} \times \text{PhytoExP}\left(\frac{\mu\text{gP}}{\text{L d}}\right) \end{aligned} \quad (261)$$

Bottom Algae Alkalinity

The bottom algae generation of alkalinity are identical to the phytoplankton formulations with the exception that they are all divided by water depth, H (m), so that the units are correct:

$$S_{A,\text{BotAlgUpN}} = \left[-P_{ab}F_i + (1 - P_{ab})\right] \times r_{AN} \times \frac{\text{BotAlgUpN}\left(\frac{\text{mgN}}{\text{m}^2\ \text{d}}\right)}{H} \quad (262)$$

$$S_{A,\text{BotAlgUpP}} = (\alpha_{\text{H2PO4}} + 2\alpha_{\text{HPO4}} + 3\alpha_{\text{PO4}}) \times r_{AP} \times \frac{\text{BotAlgUptakeP}\left(\frac{\text{mgP}}{\text{m}^2\ \text{d}}\right)}{H} \quad (263)$$

$$\begin{aligned} S_{A,\text{BotAlgEx}} &= F_i \times r_{AN} \times \frac{\text{BotAlgExN}\left(\frac{\text{mgN}}{\text{m}^2\ \text{d}}\right)}{H} \\ &\quad - (\alpha_{\text{H2PO4}} + 2\alpha_{\text{HPO4}} + 3\alpha_{\text{PO4}}) \times r_{AP} \times \frac{\text{BotAlgExP}\left(\frac{\text{mgP}}{\text{m}^2\ \text{d}}\right)}{H} \end{aligned} \quad (264)$$

Pathogen (*X*)

Pathogens are subject to death and settling,

$$S_X = -\text{PathDeath} - \text{PathSettl} \quad \textbf{(265)}$$

Death

Pathogen death is due to natural die-off and light (Chapra 1997). The death of pathogens in the absence of light is modeled as a first-order temperature-dependent decay and the death rate due to light is based on the Beer-Lambert law,

$$\text{PathDeath} = k_{dX}(T)X + \alpha_{path}\frac{I(0)/24}{k_e H}\left(1 - e^{-k_e H}\right)X \quad \textbf{(266)}$$

where $k_{dX}(T)$ = temperature-dependent pathogen die-off rate [/d] and α_{path} = a light efficiency factor [dimensionless].

Settling

Pathogen settling is represented as

$$\text{PathSettl} = \frac{v_X}{H}x \quad \textbf{(267)}$$

where v_X = pathogen settling velocity [m/d].

Salinity (*S*)

We have included salinity as an explicit state variable for cases where it occurs at levels that would influence other model state variables and processes including impacts on dissolved oxygen saturation (p. 53), and algal salinity toxicity (p. 39). It is treated as a conservative substance,

$$S_S = 0 \quad \textbf{(268)}$$

Although there are a very few rivers that have elevated salinity[9], the most common systems where salinity is important are rivers that flow into the ocean. For those systems, salinity is also important as it provides a nice tracer to calibrate tidal dispersion.

An example of its use as a tracer can be developed as follows.

SOD/Nutrient Flux Model

Sediment nutrient fluxes and sediment oxygen demand (SOD) are based on a model developed by Di Toro (Di Toro et al. 1991, Di Toro and Fitzpatrick. 1993, Di Toro 2001). The present version also benefited from James Martin's (Mississippi State University, personal communication) efforts to incorporate the Di Toro approach into EPA's WASP modeling framework.

A schematic of the model is depicted in Figure 27. As can be seen, the approach allows oxygen and nutrient sediment-water fluxes to be computed based on the downward flux of particulate organic matter from the overlying water. The sediments are divided into 2 layers: a thin (≅ 1 mm) surface aerobic layer underlain by a thicker (10 cm) lower anaerobic layer. Organic carbon, nitrogen and phosphorus are delivered to the anaerobic sediments via the settling of

[9] for example, the Lower Colorado River (Mexico), and the Luni River (India), and the eponymously names Salt River (Arizona).

particulate organic matter (i.e., phytoplankton and POC). There they are transformed by mineralization reactions into dissolved methane, ammonium and inorganic phosphorus. These constituents are then transported to the aerobic layer where some of the methane and ammonium are oxidized. The flux of oxygen from the water required for these oxidations is the sediment oxygen demand. The following sections provide details on how the model computes this SOD along with the sediment-water fluxes of carbon, nitrogen and phosphorus that are also generated in the process.

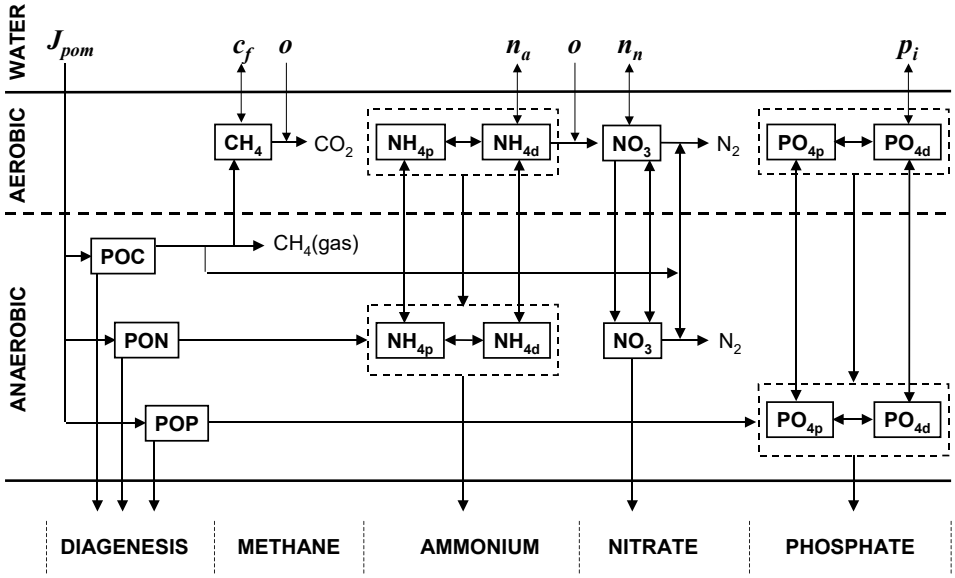

Figure 27 Schematic of SOD-nutrient flux model of the sediments.

Diagenesis

As summarized in Figure 28, the first step in the computation involves determining how much of the downward flux of particulate organic matter (POM) is converted into soluble reactive forms in the anaerobic sediments. This process is referred to as diagenesis. First the total downward flux of carbon, nitrogen and phosphorus are computed as the sums of the fluxes of settling phytoplankton and organic matter delivered to the sediments from the water column

$$J_{POC} = r_{ca} v_a a_p + r_{cd} v_{dt} m_o$$

$$J_{PON} = v_a IP_p + v_{pon} n_o \quad \textbf{(269)}$$

$$J_{POP} = v_a IN_p + v_{pop} p_o$$

where J_{POC} = the downward flux of POC [gC m^{-2} d^{-1}], v_a = phytoplankton settling velocity [m/d], a_p = phytoplankton concentration [gC/m^3], r_{cd} = the ratio of carbon to dry weight [gC/gD], v_{poc} = POC settling velocity [m/d], c_p = POC concentration [gD/m^3], J_{PON} = the downward flux of PON [mgN m^{-2} d^{-1}], q_{Np} = the phytoplankton nitrogen cell quota [mgN gC^{-1}], v_{pon} = particulate organic nitrogen settling velocity [m/d], n_o = organic nitrogen concentration [mgN m^{-3}], J_{POP} = the downward flux of POP [mgP m^{-2} d^{-1}], q_{Pp} = the phytoplankton phosphorus cell quota [mgP gC^{-1}], v_{pop} = particulate organic phosphorus settling velocity [m/d], and p_o = organic phosphorus concentration [mgP m^{-3}] .

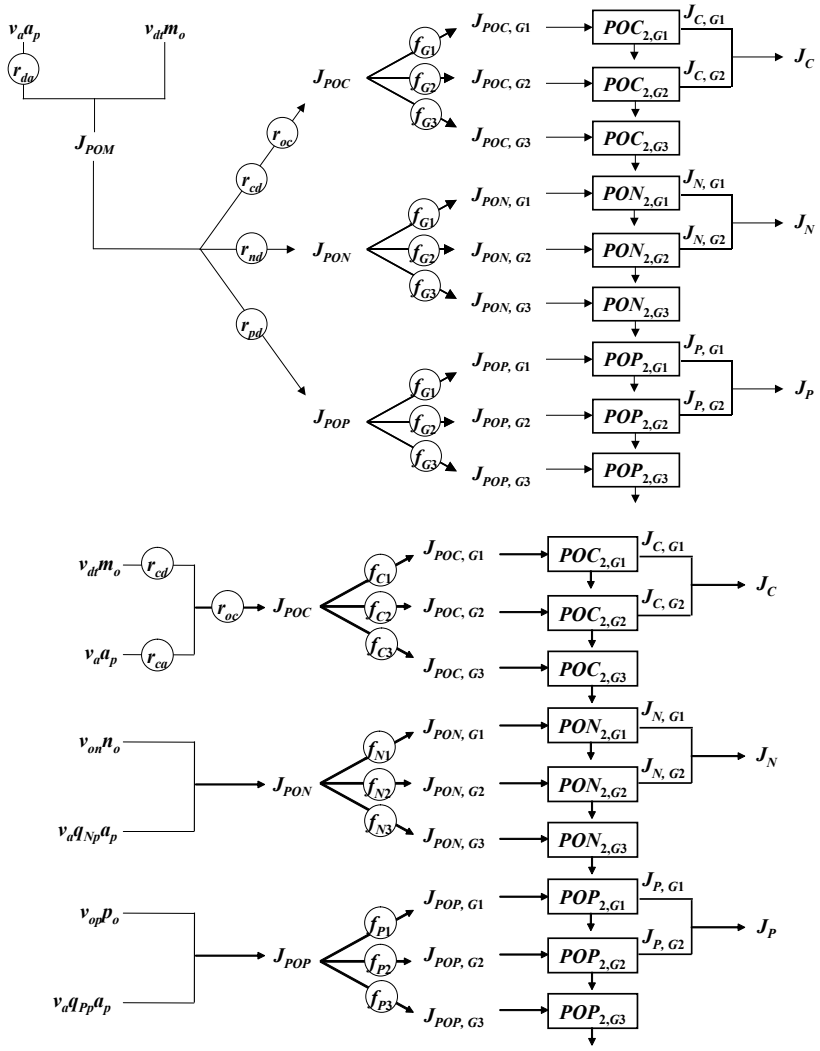

Figure 28 Representation of how settling particulate matter is transformed into fluxes of dissolved carbon (J_C) , nitrogen (J_N) and phosphorus (J_P) in the anaerobic sediments.

Note that for convenience, we express the particulate organic carbon (POC) as oxygen equivalents using the stoichiometric coefficient r_{oc}. Each of the nutrient fluxes is further broken down into three reactive fractions: labile (G1), slowly reacting (G2) and non-reacting (G3).

These fluxes are then entered into mass balances to compute the concentration of each fraction in the anaerobic layer. For example, for labile POC, a mass balance is written as

$$H_2 \frac{dPOC_{2,G1}}{dt} = J_{POC,G1} - k_{POC,G1}\theta_{POC,G1}^{T-20} H_2 POC_{2,G1} - w_2 POC_{2,G1} \quad (\mathbf{270})$$

where H_2 = the thickness of the anaerobic layer [m], $POC_{2,G1}$ = the concentration of the labile fraction of POC in the anaerobic layer [gO_2/m^3], $J_{POC,G1}$ = the flux of labile POC delivered to the anaerobic layer [$gO_2/m^2/d$], $k_{POC,G1}$ = the mineralization rate of labile POC [d^{-1}], $\theta_{POC,G1}$ = temperature correction factor for labile POC mineralization [dimensionless], and w_2 = the burial velocity [m/d]. At steady state, Eq. (214) can be solved for

$$POC_{2,G1} = \frac{J_{POC,G1}}{k_{POC,G1}\theta_{POC,G1}^{T-20}H_2 + v_b} \quad (271)$$

The flux of labile dissolved carbon, $J_{C,G1}$ [$gO_2/m^2/d$], can then be computed as

$$J_{C,G1} = k_{POC,G1}\theta_{POC,G1}^{T-20}H_2 POC_{2,G1} \quad (272)$$

In a similar fashion, a mass balance can be written and solved for the slowly reacting dissolved organic carbon. This result is then added to Eq. (216) to arrive at the total flux of dissolved carbon generated in the anaerobic sediments.

$$J_C = J_{C,G1} + J_{C,G2} \quad (273)$$

Similar equations are developed to compute the diagenesis fluxes of nitrogen, J_N [$gN/m^2/d$], and phosphorus J_P [$gP/m^2/d$].

Ammonium

Based on the mechanisms depicted in Figure 27, mass balances can be written for total ammonium in the aerobic layer and the anaerobic layers,

$$H_1\frac{dNH_{4,1}}{dt} = \omega_{12}\left(f_{pa2}NH_{4,2} - f_{pa1}NH_{4,1}\right) + K_{L12}\left(f_{da2}NH_{4,2} - f_{da1}NH_{4,1}\right) - w_2 NH_{4,1} + s\left(\frac{n_a}{1000} - f_{da1}NH_{4,1}\right) - \frac{\kappa_{NH4,1}^2}{s}\theta_{NH4}^{T-20}\frac{K_{NH4}}{K_{NH4} + NH_{4,1}}\frac{o}{2K_{NH4,O2} + o}f_{da1}NH_{4,1} \quad (274)$$

$$H_2\frac{dNH_{4,2}}{dt} = J_N + \omega_{12}\left(f_{pa1}NH_{4,1} - f_{pa2}NH_{4,2}\right) + K_{L12}\left(f_{da1}NH_{4,1} - f_{da2}NH_{4,2}\right) + w_2\left(NH_{4,1} - NH_{4,2}\right) \quad (275)$$

where H_1 = the thickness of the aerobic layer [m], $NH_{4,1}$ and $NH_{4,2}$ = the concentration of total ammonium in the aerobic layer and the anaerobic layers, respectively [gN/m^3], n_a = the ammonium concentration in the overlying water [mgN/m^3], $\kappa_{NH4,1}$ = the reaction velocity for nitrification in the aerobic sediments [m/d], θ_{NH4} = temperature correction factor for nitrification [dimensionless], K_{NH4} = ammonium half-saturation constant [gN/m^3], o = the dissolved oxygen concentration in the overlying water [gO_2/m^3], and $K_{NH4,O2}$ = oxygen half-saturation constant [mgO_2/L], and J_N = the diagenesis flux of ammonium [$gN/m^2/d$].

The fraction of ammonium in dissolved (f_{dai}) and particulate (f_{pai}) form are computed as

$$f_{dai} = \frac{1}{1 + m_i\pi_{ai}} \quad (276)$$

$$f_{pai} = 1 - f_{dai} \quad (277)$$

where m_i = the solids concentration in layer i [gD/m^3], and π_{ai} = the partition coefficient for ammonium in layer i [m^3/gD].

The mass transfer coefficient for particle mixing due to bioturbation between the layers, ω_{12} [m/d], is computed as

$$\omega_{12} = \frac{D_p \theta_{Dp}^{T-20}}{H_2} \frac{POC_{2,G1} / r_{oc}}{POC_R} \frac{o}{K_{M,Dp} + o} \quad (278)$$

where D_p = diffusion coefficient for bioturbation [m^2/d], θ_{Dp} = temperature coefficient [dimensionless], POC_R = reference G1 concentration for bioturbation [gC/m^3] and $K_{M,Dp}$ = oxygen half-saturation constant for bioturbation [gO_2/m^3].

The mass transfer coefficient for pore water diffusion between the layers, K_{L12} [m/d], is computed as,

$$K_{L12} = \frac{D_d \theta_{Dd}^{T-20}}{H_2 / 2} \quad (279)$$

where D_d = pore water diffusion coefficient [m^2/d], and θ_{Dd} = temperature coefficient [dimensionless].

The mass transfer coefficient between the water and the aerobic sediments, s [m/d], is computed as

$$s = \frac{SOD}{o} \quad (280)$$

where SOD = the sediment oxygen demand [$gO_2/m^2/d$].

At steady state, Eqs. (218) and (219) are two simultaneous nonlinear algebraic equations. The equations can be linearized by assuming that the $NH_{4,1}$ term in the Monod term for nitrification is constant. The simultaneous linear equations can then be solved for $NH_{4,1}$ and $NH_{4,2}$. The flux of ammonium to the overlying water can then be computed as

$$J_{NH4} = s\left(f_{da1} NH_{4,1} - \frac{n_a}{1000} \right) \quad (281)$$

Nitrate

Mass balances for nitrate can be written for the aerobic and anaerobic layers as

$$H_1 \frac{dNO_{3,1}}{dt} = K_{L12}\left(NO_{3,2} - NO_{3,1}\right) - w_2 NO_{3,1} + s\left(\frac{n_n}{1000} - NO_{3,1}\right) + \frac{\kappa_{NH4,1}^2}{s} \theta_{NH4}^{T-20} \frac{K_{NH4}}{K_{NH4} + NH_{4,1}} \frac{o}{2K_{NH4,O2} + o} f_{da1} NH_{4,1} - \frac{\kappa_{NO3,1}^2}{s} \theta_{NO3}^{T-20} NO_{3,1} \quad (282)$$

$$H_2 \frac{dNO_{3,2}}{dt} = J_N + K_{L12}\left(NO_{3,1} - NO_{3,2}\right) + w_2\left(NO_{3,1} - NO_{3,2}\right) - \kappa_{NO3,2} \theta_{NO3}^{T-20} NO_{3,2} \quad (283)$$

where $NO_{3,1}$ and $NO_{3,2}$ = the concentration of nitrate in the aerobic layer and the anaerobic layers, respectively [gN/m^3], n_n = the nitrate concentration in the overlying water [mgN/m^3], $\kappa_{NO3,1}$ and $\kappa_{NO3,2}$ = the reaction velocities for denitrification in the aerobic and anaerobic sediments, respectively [m/d], and θ_{NO3} = temperature correction factor for denitrification [dimensionless].

In the same fashion as for Eqs. (218) and (219), Eqs. (226) and (227) can be linearized and solved for $NO_{3,1}$ and $NO_{3,2}$. The flux of nitrate to the overlying water can then be computed as

$$J_{NO3} = s\left(NO_{3,1} - \frac{n_n}{1000}\right) \quad (284)$$

Denitrification requires a carbon source as represented by the following chemical equation,

$$5CH_2O + 4NO_3^- + 4H^+ \rightarrow 5CO_2 + 2N_2 + 7H_2O \quad (285)$$

The carbon requirement (expressed in oxygen equivalents per nitrogen) can therefore be computed as

$$r_{ondn} = 2.67\frac{gO_2}{gC}\frac{5\text{ moleC}\times 12\text{ gC/moleC}}{4\text{ moleN}\times 14\text{ gN/moleN}}\times\frac{1\text{ gN}}{1000\text{ mgN}} = 0.00286\frac{gO_2}{mgN} \quad (286)$$

Therefore, the oxygen equivalents consumed during denitrification, $J_{O2,dn}$ [$gO_2/m^2/d$], can be computed as

$$J_{O2,dn} = 1000\frac{mgN}{gN}\times r_{ondn}\left(\frac{\kappa_{NO3,1}^2}{s}\theta_{NO3}^{T-20}NO_{3,1} + \kappa_{NO3,2}\theta_{NO3}^{T-20}NO_{3,2}\right) \quad (287)$$

Methane

The dissolved carbon generated by diagenesis is converted to methane in the anaerobic sediments. Because methane is relatively insoluble, its saturation can be exceeded and methane gas produced. As a consequence, rather than write a mass balance for methane in the anaerobic layer, an analytical model developed by Di Toro et al. (1990) is used to determine the steady-state flux of dissolved methane corrected for gas loss delivered to the aerobic sediments.

First, the carbon diagenesis flux is corrected for the oxygen equivalents consumed during denitrification,

$$J_{CH4,T} = J_C - J_{O2,dn} \quad (288)$$

where $J_{CH4,T}$ = the carbon diagenesis flux corrected for denitrification [$gO_2/m^2/d$]. In other words, this is the total anaerobic methane production flux expressed in oxygen equivalents.

If $J_{CH4,T}$ is sufficiently large ($\geq 2K_{L12}C_s$), methane gas will form. In such cases, the flux can be corrected for the gas loss,

$$J_{CH4,d} = \sqrt{2K_{L12}C_sJ_{CH4,T}} \quad (289)$$

where $J_{CH4,d}$ = the flux of dissolved methane (expressed in oxygen equivalents) that is generated in the anaerobic sediments and delivered to the aerobic sediments [$gO_2/m^2/d$], C_s = the saturation concentration of methane expressed in oxygen equivalents [mgO_2/L]. If $J_{CH4,T} < 2K_{L12}C_s$, then no gas forms and

$$J_{CH4,d} = J_{CH4,T} \quad (290)$$

The methane saturation concentration is computed as

$$C_s = 100\left(1+\frac{H}{10}\right)1.024^{20-T} \quad (291)$$

where H = water depth [m] and T = water temperature [°C].

A methane mass balance can then be written for the aerobic layer as

$$H_1 \frac{dCH_{4,1}}{dt} = J_{CH4,d} + s\left(c_f - CH_{4,1}\right) - \frac{\kappa_{CH4,1}^2}{s}\theta_{CH4}^{T-20} CH_{4,1} \quad (292)$$

where $CH_{4,1}$ = methane concentration in the aerobic layer [gO_2/m^3], c_f = fast CBOD in the overlying water [gO_2/m^3], $\kappa_{CH4,1}$ = the reaction velocity for methane oxidation in the aerobic sediments [m/d], and θ_{CH4} = temperature correction factor [dimensionless]. At steady, state, this balance can be solved for

$$CH_{4,1} = \frac{J_{CH4,d} + sc_f}{s + \frac{\kappa_{CH4,1}^2}{s}\theta_{CH4}^{T-20}} \quad (293)$$

The flux of methane to the overlying water, JCH_4 [$gO_2/m^2/d$], can then be computed as

$$J_{CH4} = s\left(CH_{4,1} - c_f\right) \quad (294)$$

SOD

The SOD [$gO_2/m^2/d$] is equal to the sum of the oxygen consumed in methane oxidation and nitrification,

$$SOD = CSOD + NSOD \quad (295)$$

where $CSOD$ = the amount of oxygen demand generated by methane oxidation [$gO_2/m^2/d$] and $NSOD$ = the amount of oxygen demand generated by nitrification [$gO_2/m^2/d$]. These are computed as

$$CSOD = \frac{\kappa_{CH4,1}^2}{s}\theta_{CH4}^{T-20} CH_{4,1} \quad (296)$$

$$NSOD = r_{on}\frac{\kappa_{NH4,1}^2}{s}\theta_{NH4}^{T-20}\frac{K_{NH4}}{K_{NH4} + NH_{4,1}}\frac{o}{2K_{NH4,O2} + o} f_{da1} NH_{4,1} \quad (297)$$

where r_{on} = the ratio of oxygen to nitrogen consumed during nitrification [= 4.57 gO_2/gN].

Inorganic Phosphorus

Mass balances can be written total inorganic phosphorus in the aerobic layer and the anaerobic layers as

$$H_1 \frac{dPO_{4,1}}{dt} = \omega_{12}\left(f_{pp2}PO_{4,2} - f_{pp1}PO_{4,1}\right) + K_{L12}\left(f_{dp2}PO_{4,2} - f_{dp1}PO_{4,1}\right) - w_2 PO_{4,1} + s\left(\frac{p_i}{1000} - f_{da1}PO_{4,1}\right) \quad (298)$$

$$H_2 \frac{dPO_{4,2}}{dt} = J_P + \omega_{12}\left(f_{pp1}PO_{4,1} - f_{pp2}PO_{4,2}\right) + K_{L12}\left(f_{dp1}PO_{4,1} - f_{dp2}PO_{4,2}\right) + w_2\left(PO_{4,1} - PO_{4,2}\right) \quad (299)$$

where $PO_{4,1}$ and $PO_{4,2}$ = the concentration of total inorganic phosphorus in the aerobic layer and the anaerobic layers, respectively [gP/m^3], p_i = the inorganic phosphorus in the overlying water [mgP/m^3], and J_P = the diagenesis flux of phosphorus [gP/m^2/d].

The fraction of phosphorus in dissolved (f_{dpi}) and particulate (f_{ppi}) form are computed as

$$f_{dpi} = \frac{1}{1 + m_i \pi_{pi}} \quad \textbf{(300)}$$

$$f_{ppi} = 1 - f_{dpi} \quad \textbf{(301)}$$

where π_{pi} = the partition coefficient for inorganic phosphorus in layer i [m^3/gD].

The partition coefficient in the anaerobic layer is set to an input value. For the aerobic layer, if the oxygen concentration in the overlying water column exceeds a critical concentration, o_{crit} [gO$_2$/m^3], then the partition coefficient is increased to represent the sorption of phosphorus onto iron oxyhydroxides as in

$$\pi_{p1} = \pi_{p2}\left(\Delta\pi_{PO4,1}\right) \quad \textbf{(302)}$$

where $\Delta\pi_{PO4,1}$ is a factor that increases the aerobic layer partition coefficient relative to the anaerobic coefficient.

If the oxygen concentration falls below o_{crit} then the partition coefficient is decreased smoothly until it reaches the anaerobic value at zero oxygen,

$$\pi_{p1} = \pi_{p2}\left(\Delta\pi_{PO4,1}\right)^{o/o_{crit}} \quad \textbf{(303)}$$

Equations (242) and (243) can be solved for $PO_{4,1}$ and $PO_{4,2}$. The flux of phosphorus to the overlying water can then be computed as

$$J_{PO4} = s\left(PO_{4,1} - \frac{p_i}{1000}\right) \quad \textbf{(304)}$$

Solution Scheme

Although the foregoing sequence of equations can be solved, a single computation will not yield a correct result because of the interdependence of the equations. For example, the surface mass transfer coefficient s depends on SOD. The SOD in turn depends on the ammonium and methane concentrations which themselves are computed via mass balances that depend on s. Hence, an iterative technique must be used. The procedure used in QUAL2K is

1. Determine the diagenesis fluxes: J_C, J_N and J_P.
2. Start with an initial estimate of SOD,

$$SOD_{init} = J_C + r'_{on} J_N \quad \textbf{(305)}$$

   where r_{on}' = the ratio of oxygen to nitrogen consumed for total conversion of ammonium to nitrogen gas via nitrification/denitrification [= 1.714 gO$_2$/gN]. This ratio accounts for the carbon utilized for denitrification.

3. Compute s using

$$s = \frac{SOD_{init}}{o} \quad (306)$$

4. Solve for ammonium, nitrate and methane, and compute the *CSOD* and *NSOD*.
5. Make a revised estimate of SOD using the following weighted average

$$SOD = \frac{SOD_{init} + CSOD + NSOD}{2} \quad (307)$$

6. Check convergence by calculating an approximate relative error

$$\varepsilon_a = \left| \frac{SOD - SOD_{init}}{SOD} \right| \times 100\% \quad (308)$$

7. If ε_a is greater than a prespecified stopping criterion ε_s then set $SOD_{init} = SOD$ and return to step 2.
8. If convergence is adequate ($\varepsilon_a \le \varepsilon_s$), then compute the inorganic phosphorus concentrations.
9. Compute the ammonium, nitrate, methane and phosphate fluxes.

Supplementary Fluxes

Because of the presence of organic matter deposited prior to the summer steady-state period (e.g., during spring runoff), it is possible that the downward flux of particulate organic matter is insufficient to generate the observed SOD. In such cases, a supplementary SOD can be prescribed,

$$SOD_t = SOD + SOD_s \quad (309)$$

where SOD_t = the total sediment oxygen demand [$gO_2/m^2/d$], and SOD_s = the supplementary SOD [$gO_2/m^2/d$]. In addition, prescribed ammonia and methane fluxes can be used to supplement the computed fluxes.

REFERENCES

Adams, E.E., D. J. Cosler, and K.R. Helfrich. 1987. "Analysis of evaporation data from heated ponds", cs-5171, research project 2385-1, Electric Power Research Institute, Palo Alto, California 94304. April.

Andrews and Rodvey, 1980. Heat exchange between water and tidal flats. D.G.M 24(2). (in German).

APHA, 2017. Standard Methods for the Examination of Water and Wastewater, 23rd ed.; Rice, E.W., Baird, R.B., Eaton, A.D., Eds.; American Public Health Association: Washington, D.C., USA.

Asaeda, T., and T. Van Bon, Modelling the effects of macrophytes on algal blooming in eutrophic shallow lakes. Ecol. Model., 104: 261-287, 1997.

Baker, K.S. and Frouin, R. 1987. Relation between photosynthetically available radiation and total insolation at the ocean surface under clear skies. Limnol. Oceanogr. 1370-1377.

Baly, E.C.C. 1935. The Kinetics of Photosynthesis. *Proc. Royal Soc. London Ser. B*, 117:218-239.

Banks, R. B. 1975. "Some Features of Wind Action on Shallow Lakes." *J. Environ Engr. Div. ASCE.* 101(EE5): 813-827.

Banks, R. B. and Herrera, F. F. 1977. "Effect of Wind and Rain on Surface Reaeration." *J. Environ Engr. Div. ASCE.* 103(EE3): 489-504.

Barnwell, T.O., Brown, L.C., and Mareck, W. 1989). Application of Expert Systems Technology in Water Quality Modeling. *Water Sci. Tech.* 21(8-9):1045-1056.

Bejan, A. 1993. *Heat Transfer.* Wiley, New York, NY.

Beven, K.J., Gilman, K., and Newson, M. 1979. Flow and flow routing in upland channel networks. Hydrol. Sci. Bull. 24:43-69.

Bowie, G.L., Mills, W.B., Porcella, D.B., Campbell, C.L., Pagenkopf, J.R., Rupp, G.L., Johnson, K.M., Chan, P.W.H., Gherini, S.A. and Chamberlin, C.E. 1985. Rates, Constants, and Kinetic Formulations in Surface Water Quality Modeling. U.S. Envir. Prot. Agency, ORD, Athens, GA, ERL, EPA/600/3-85/040.

Brady, D.K., Graves, W.L., and Geyer, J.C. 1969. Surface Heat Exchange at Power Plant Cooling Lakes, Cooling Water Discharge Project Report, No. 5, Edison Electric Inst. Pub. No. 69-901, New York, NY.

Bras, R.L. 1990. Hydrology. Addison-Wesley, Reading, MA.

Brown, L.C., and Barnwell, T.O. 1987. The Enhanced Stream Water Quality Models QUAL2E and QUAL2E-UNCAS, EPA/600/3-87-007, U.S. Environmental Protection Agency, Athens, GA, 189 pp.

Brunt, D. 1932. Notes on radiation in the atmosphere: I. Quart J Royal Meteorol Soc 58:389-420.

Brutsaert, W. 1982. Evaporation into the atmosphere: theory, history, and applications. D.Reidel Publishing Co., Hingham MA, 299 p.

Butts, T. A. and Evans, R. L. 1983. Effects of Channel Dams on Dissolved Oxygen Concentrations in Northeastern Illinois Streams, Circular 132, State of Illinois, Dept. of Reg. and Educ., Illinois Water Survey, Urbana, IL.

Camacho, L.A., Chapra, S.C. 2021. Impact of global warming on the oxygen concentration and BOD assimilative capacity of the world's rivers: 2. Application to a high elevation river. *Water*.

Carslaw, H.S. and Jaeger, J.C. 1959. Conduction of Heat in Solids, Oxford Press, Oxford, UK, 510 pp.

Cengel, Y.A. 1998 *Heat Transfer: A Practical Approach.* New York, McGraw-Hill.

Chapra and Canale 2006. *Numerical Methods for Engineers*, *5th Ed.* New York, McGraw-Hill.

Chapra, S.C. 1997. *Surface water quality modeling*. New York, McGraw-Hill.

Chapra, S.C. 2022. Applied Numerical Methods with MATLAB for Engineering and Science, 5th Ed., WCB/McGraw-Hill, New York, N.Y.

Chow, V.T., Maidment, D.R., and Mays, L.W. 1988. *Applied Hydrology*. New York, McGraw-Hill, 592 pp.

Churchill, M.A., Elmore, H.L., and Buckingham, R.A. 1962. The prediction of stream reaeration rates. *J. Sanit. Engrg. Div.* , ASCE, 88{4),1-46.

Coffaro, G. and A. Sfriso, Simulation model of Ulva rigida growth in shallow water of the Lagoon of Venice. Ecol. Model., 102: 55-66, 1997.

Covar, A. P. 1976. "Selecting the Proper Reaeration Coefficient for Use in Water Quality Models." Presented at the U.S. EPA Conference on Environmental Simulation and Modeling, April 19-22, 1976, Cincinnati, OH.

Di Toro, D. M. and J. F. Fitzpatrick. 1993. Chesapeake Bay sediment flux model. Tech. Report EL-93-2, U.S. Army Corps of Engineers, Waterways Experiment Station, Vicksburg, Mississippi, 316 pp.

Di Toro, D.M, Paquin, P.R., Subburamu, K. and Gruber, D.A. 1991. Sediment Oxygen Demand Model: Methane and Ammonia Oxidation. *J. Environ. Eng.*, 116(5):945-986.

Di Toro, D.M. 1978. Optics of Turbid Estuarine Waters: Approximations and Applications. *Water Res*. 12:1059-1068.

Di Toro, D.M. 2001. *Sediment Flux Modeling*. Wiley-Interscience, New York, NY.

Droop, M.R. 1974. The nutrient status of algal cells in continuous culture. *J.Mar.Biol.Assoc. UK,* 54:825-855.

Ecology. 2003. Shade.xls - a tool for estimating shade from riparian vegetation. Washington State Department of Ecology. http://www.ecy.wa.gov/programs/eap/models/

Edinger, J.E., Brady, D.K., and Geyer, J.C. 1974. Heat Exchange and Transport in the Environment. Report No. 14, EPRI Pub. No. EA-74-049-00-3, Electric Power Research Institute, Palo Alto, CA.

Fischer, H.B., List, E.J., Koh, R.C.Y., Imberger, J., and Brooks, N.H. (1979) *Mixing in inland and coastal waters*, San Diego, Academic Press, 483 p.

Finlayson-Pitts, B.J. and Pitts, J.N., Jr. 2000. *Chemistry of the upper and lower atmosphere : theory, experiments, and applications*. Academic Press.

Finnemore, E.J. and Franzini, J.B. 2002. *Fluid Mechanics with Engineering Applications*, 10th Ed. New York, McGraw, Hill.

Flynn, K.F., Chudyk, W., Watson, V., Chapra, S.C., Suplee, M.W. 2018. Influence of biomass and water velocity on light attenuation of Cladophora glomerata L. (Kuetzing) in rivers. *Aquatic Botany*, 151:62–70

Geiger, R. 1965. The climate near the ground. Harvard University Press. Cambridge, MA.

Gordon, N.D, T.A. McMahon, and B.L. Finlayson. 1992. Stream Hydrology, An Introduction for Ecologists. Published by John Wiley and Sons.

Grigull, U. and Sandner, H. 1984. *Heat Conduction.* Springer-Verlag, New York, NY.

Hamilton, D.P., and S.G. Schladow, Prediction of water quality in lakes and reservoirs. 1. Model description. Ecol. Model., 96: 91-110, 1997.

Harbeck, G. E., 1962, A practical field technique for measuring reservoir evaporation utilizing mass-transfer theory. US Geological Survey Professional Paper 272-E, 101-5.

Harned, H.S., and Hamer, W.J. 1933. "The Ionization Constant of Water." *J. Am., Chem. Soc.* 51:2194.

Helfrich, K.R., E.E. Adams, A.L. Godbey, and D.R.F. Harleman. 1982. Evaluation of models for predicting evaporative water loss in cooling impoundments. Report CS-2325, Research project 1260-17. Electric Power Research Institute, Pala Alto, CA. March 1982.

Hellweger, F/L., K.L. Farley, U. Lall, and D.M. DiToro. 2003. Greedy algae reduce arsenate. Limnol.Oceanogr. 48(6), 2003, 2275-2288.

Holland, H.D. 1978. The Chemistry of the Atmosphere and Oceans. Wiley-Interscience, NY, NY.

Hutchinson, G.E. 1957. *A Treatise on Limnology, Vol. 1, Physics and Chemistry*. Wiley, New York, NY.
Jassby, A. D., Platt, T. (1976). Mathematical formulation of the relationship between photosynthesis and light for phytoplankton. Limnol. Oceanogr. 21: 540-547
Jaworski, N. S., Lear, D. W., Jr., Villa, O., Jr. (1972). Nutrient
Jobson, H.E. 1977. Bed Conduction Computation for Thermal Models. J. Hydraul. Div. ASCE. 103(10):1213-1217.
Koberg, G.E. 1964. Methods to compute long-wave radiation from the atmosphere and reflected solar radiation from a water surface. US Geological Survey Professional Paper 272-F.
Kreith, F. and Bohn, M.S. 1986. *Principles of Heat Transfer, 4th Ed.* Harper and Row, New York, NY.
Laws, E. A. and Chalup, M. S. 1990. A Microalgal Growth Model. *Limnol. Oceanogr.* 35(3):597-608.
LI-COR, 2003. Radiation Measurement Instruments, LI-COR, Lincoln, NE, 30 pp.
Likens, G. E., and Johnson, N. M. (1969). "Measurements and analysis of the annual heat budget for sediments of two Wisconsin lakes." *Limnol. Oceanogr.*, 14(1):115-135.
Mackay, D. and Yeun, A.T.K. 1983. Mass Transfer Coefficient Correlations for Volatilization of Organic Solutes from Water. *Environ. Sci. Technol.* 17:211-233.
Marciano, J.K. and G.E. Harbeck. 1952. Mass transfer studies in water loss investigation: Lake Hefner studies. Geological Circular 229. U.S. Geological Survey, Washington DC.
McIntyre, C. D. 1973. Periphyton dynamics in laboratory streams: A simulation model and its implications. *Ecol. Monogr.* 43:399-420.
Meeus, J. 1999. Astronomical algorithms. Second edition. Willmann-Bell, Inc. Richmond, VA.
Melching, C.S. and Flores, H.E. 1999. Reaeration Equations from U.S. Geological Survey Database. *J. Environ. Engin.*, 125(5):407-414.
Mills, A.F. 1992. *Heat Transfer.* Irwin, Homewood, IL.
Moog, D.B., and Iirka, G.H. 1998. Analysis of reaeration equations using mean multiplicative error. *J. Environ. Engrg.*, ASCE, 124(2), 104-110.
Munson, B.R., Young, D.F., Okiishi, T.H., and W.D. Huebsch. 2009. *Fundamentals of Fluid Mechanics*. 6th edition. John Wiley & Sons. Hoboken, NJ.
Nakshabandi, G.A. and H. Kohnke. 1965. Thermal conductivity and diffusivity of soils as related to moisture tension and other physical properties. Agr. Met. Vol 2.
O'Connor, D.J., and Dobbins, W.E. 1958. Mechanism of reaeration in natural streams. *Trans. ASCE*, 123, 641-684.
Owens, M., Edwards, R.W., and Gibbs, J.W. 1964. Some reaeration studies in streams. *Int. J. Air and Water Pollution*, 8, 469-486.
Plummer, L.N. and Busenberg, E. 1982. The Solubilities of Calcite, Aragonite and Vaterite in CO2-H2O Solutions Between 0 and 90 oC, and an Evaluation of the Aqueous Model for the System CaCO3–CO2–H2O. *Geochim. Cosmochim.* 46:1011-1040.
Raudkivi, A. I. 1979. *Hydrology*. Pergamon, Oxford, England.
Redfield, A.C., Ketchum, B.H. and Richards, F.A. 1963. The Influence of Organisms on the Composition of Seawater, in *The Sea*, M.N. Hill, ed. Vol. 2, pp. 27-46, Wiley-Interscience, NY.
Riley, G.A. 1956. Oceanography of Long Island Sound 1952-1954. II. Physical Oceanography, Bull. Bingham Oceanog. Collection 15, pp. 15-16.
Rosgen, D. 1996. Applied river morphology. Wildland Hydrology publishers. Pagosa Springs, CO.
Rutherford, J.C., Scarsbrook, M.R. and Broekhuizen, N. 2000. Grazer Control of Stream Algae: Modeling Temperature and Flood Effects. *J. Environ. Eng.* 126(4):331-339.

Ryan, P.J. and D.R.F. Harleman. 1971. Prediction of the annual cycle of temperature changes in a stratified lake or reservoir. Mathematical model and user's manual. Ralph M. Parsons Laboratory Report No. 137. Massachusetts Institute of Technology. Cambridge, MA.

Ryan, P.J. and K.D. Stolzenbach.1972. Engineering aspects of heat disposal from power generation, (D.R.F. Harleman, ed.). R.M. Parson Laboratory for Water Resources and Hydrodynamics, Department of Civil Engineering, Massachusetts Institute of Technology, Cambridge, MA

Schwarzenbach, R.P., Gschwend, P.M., and Imboden, D.M. 1993. Environmental Organic Chemistry. Wiley-Interscience, 681 pp.

Shanahan, P. 1984. Water temperature modeling: a practical guide. In: Proceedings of stormwater and water quality model users group meeting, April 12-13, 1984. USEPA, EPA-600/9-85-003. (users.rcn.com/shanahan.ma.ultranet/TempModeling.pdf).

Smith, E.L. 1936. Photosynthesis in Relation to Light and Carbon Dioxide. Proc. Natl. Acad. Sci. 22:504-511.

Steele, J.H. 1962. Environmental Control of Photosynthesis in the Sea. Limnol. Oceanogr. 7:137-150.

Stumm, W. and Morgan, J.J. 1996. Aquatic Chemistry, 3rd Ed., New York, Wiley-Interscience, 1022 pp.

Szeicz, G. 1974. Solar radiation for plant growth. J. Appl. Ecol. 11:617-636.

Thackston, E. L., and Dawson, J.W. III. 2001. Recalibration of a Reaeration Equation. *J. Environ. Engrg.*, 127(4):317-320.

Thackston, E. L., and Krenkel, P. A. 1969. Reaeration-prediction in natural streams. *J. Sanit. Engrg. Div.*, ASCE, 95(1), 65-94.

Tsivoglou, E. C., and Neal, L.A. 1976. Tracer Measurement of Reaeration. III. Predicting the Reaeration Capacity of Inland Streams. Journal of the Water Pollution Control Federation, 48(12):2669-2689.

Thomann, R.V. and Mueller, J.A. 1987. Principles of Surface Water Quality Modeling and Control. New York, Harper-Collins.

TVA, 1972. Heat and mass transfer between a water surface and the atmosphere. Water Resources Research, Laboratory Report No. 14. Engineering Laboratory, Division of Water Control Planning, Tennessee Valley Authority, Norris TN.

Wanninkhof, R., Ledwell, I. R., and Crusius, I. 1991. "Gas Transfer Velocities on Lakes Measured with Sulfur Hexafluoride." In Symposium Volume of the Second International Conference on Gas Transfer at Water Surfaces, S.C. Wilhelms and I.S. Gulliver, eds., Minneapolis, MN .

Wood, K.G. 1974. Carbon Dioxide Diffusivity Across the Air-Water Interface. *Archiv für Hydrobiol.* 73(1):57-69.

APPENDIX A: NOMENCLATURE

Symbol	**Definition**	Units
$A_{acres,i}$	surface area of element i	acres
$[CO_2]_s$	saturation concentration of carbon dioxide	mole/L
$[CO_3^{2-}]$	carbonate ion	mole/L
$[H^+]$	hydronium ion	mole/L
$[H_2CO_3^*]$	sum of dissolved carbon dioxide and carbonic acid	mole/L
$[HCO_3^-]$	bicarbonate ion	mole/L
$[OH^-]$	hydroxyl ion	mole/L
a	velocity rating curve coefficient	dimensionless
a"	mean atmospheric transmission coefficient after scattering and absorption	Dimensionless
a'	mean atmospheric transmission coefficient	Dimensionless
a_1	atmospheric molecular scattering coefficient for radiation transmission	dimensionless
A_a	atmospheric long-wave radiation coefficient	dimensionless
A_b	atmospheric long-wave radiation coefficient	$mmHg^{-0.5}$ or $mb^{-0.5}$
a_b	bottom algae	gC/m^2
A_c	cross-sectional area	m^2
a_d	coefficient correcting dam reaeration for water quality	dimensionless
Alk	alkalinity	eq L^{-1} or $mgCaCO_3/L$
a_p	phytoplankton concentration	gC/m^3
a_t	atmospheric attenuation	dimensionless
a_{tc}	atmospheric transmission coefficient	dimensionless
B	average element width	m
b	velocity rating curve exponent	dimensionless
B_0	bottom width	m
b_d	coefficient correcting dam reaeration for dam type	dimensionless
c_1	Bowen's coefficient	0.47 mmHg/ºC
c_f	fast reacting CBOD	mgO_2/L
$C_{gb}(T)$	temperature-dependent maximum photosynthesis rate	$gC/(m^2\ d)$
$CH_{4,1}$	methane concentration in the aerobic sediment layer	gO_2/m^3
C_L	fraction of sky covered with clouds	dimensionless
$c_{nps,i,j}$	j^{th} non-point source concentration for element i	ºC
C_p	specific heat of water	cal/(g ºC)
$c_{ps,i,j}$	j^{th} point source concentration for element i	ºC
c_s	slowly reacting CBOD	mgO_2/L
C_s	saturation concentration of methane	mgO_2/L
$CSOD$	the amount of oxygen demand generated by methane oxidation	$gO_2/m^2/d$
c_T	total inorganic carbon	mole L^{-1}
$c'_{T,i-1}$	concentration of inorganic carbon entering element below a dam	mgO_2/L
d	dust attenuation coefficient	dimensionless
D_d	pore water diffusion coefficient	m^2/d

D_p	diffusion coefficient for bioturbation	m^2/d
E'_i	bulk dispersion coefficient between elements i and $i + 1$	m^3/d
*e*air	air vapor pressure	mm Hg
elev	elevation above sea level	m
E_n	numerical dispersion	m^2/d
$E_{p,i}$	longitudinal dispersion between elements i and $i + 1$	m^2/s
eqtime	equation of time: the difference between mean solar time and true solar time when located on the reference longitude of the time zone considered	minutes
e_s	saturation vapor pressure at water surface	mmHg
f	photoperiod	fraction of day
f_{dai}	fraction of ammonium in dissolved form in sediment layer i	dimensionless
f_{dpi}	fraction of inorganic phosphorus in dissolved form in sediment layer i	dimensionless
F_{Lp}	phytoplankton growth attenuation due to light	dimensionless
F_{oxc}	attenuation of CBOD oxidation due to low oxygen	dimensionless
F_{oxdn}	enhancement of denitrification at low oxygen concentration	dimensionless
F_{oxna}	attenuation due to low oxygen	dimensionless
F_{oxrb}	attenuation due to low oxygen	dimensionless
F_{oxrp}	attenuation due to low oxygen	dimensionless
f_{pai}	fraction of ammonium in particulate form in sediment layer i	dimensionless
f_{ppi}	fraction of inorganic phosphorus in particulate form in sediment layer i	dimensionless
F_u	fraction unionized ammonia	
g	acceleration due to gravity	$= 9.81\ m/s^2$
gX	mass of element X	g
H	water depth	m
H_d	drop in water elevation for a dam	m
H_i	thickness of sediment layer i	m
H_{sed}	sediment thickness	cm
$I(0)$	solar radiation at water surface	$cal/cm^2/d$
I_0	extraterrestrial radiation	$cal/cm^2/d$
J_{an}	net atmospheric longwave radiation flux	$cal/(cm^2\ d)$
J_{br}	longwave back radiation flux from water	$cal/(cm^2\ d)$
J_c	conduction flux	$cal/(cm^2\ d)$
$J_{C,G1}$	flux of labile dissolved carbon	$gO_2/m^2/d$
J_{CH4}	flux of methane from sediments to the overlying water	$gO_2/m^2/d$
$J_{CH4,d}$	flux of dissolved methane generated in anaerobic sediments corrected for methane gas formation	$gO_2/m^2/d$
$J_{CH4,T}$	carbon diagenesis flux corrected for denitrification	$gO_2/m^2/d$
J_e	evaporation flux	$cal/(cm^2\ d)$
J_a	air-water heat flux	$cal/(cm^2\ d)$
J_N	nitrogen diagenesis flux	$gN/m^2/d$
$J_{O2,dn}$	oxygen equivalents consumed during denitrification	$gO_2/m^2/d$
J_P	phosphorus diagenesis flux	$gP/m^2/d$

$J_{POC,G1}$	the flux of labile POC delivered to the anaerobic sediment layer	$gO_2/m^2/d$
J_{POM}	the downward flux of POM	$gD\ m^{-2}\ d^{-1}$
J_{si}	sediment-water heat flux	$cal/(cm^2\ d)$
J_{sn}	net solar shortwave radiation flux	$cal/(cm^2\ d)$
$k(T)$	temperature dependent first-order reaction rate	/d
K_1	acidity constant for dissociation of carbonic acid	
K_2	acidity constant for dissociation of bicarbonate	
K_a	equilibrium coefficient for ammonium dissociation	
$k_a(T)$	temperature-dependent oxygen reaeration coefficient	/d
$k_{ac}(T)$	temperature-dependent carbon dioxide reaeration coefficient	/d
$k_{db}(T)$	temperature-dependent bottom algae death rate	/d
$k_{dc}(T)$	temperature-dependent fast CBOD oxidation rate	/d
$k_{dn}(T)$	temperature-dependent denitrification rate	/d
$k_{dp}(T)$	temperature-dependent phytoplankton death rate	/d
$k_{poc}(T)$	temperature-dependent POC dissolution rate	/d
$k_{dX}(T)$	temperature-dependent pathogen die-off rate	/d
k_e	light extinction coefficient	$/m^1$
k_{eb}	a background coefficient accounting for extinction due to water and color	/m
$k_{gp}(T)$	maximum photosynthesis rate at temperature T	/d
K_H	Henry's constant	mole/(L atm)
$k_{hc}(T)$	temperature-dependent slow CBOD hydrolysis rate	/d
$k_{hn}(T)$	temperature-dependent organic nitrogen hydrolysis rate	/d
k_{hnxb}	preference coefficient of bottom algae for ammonium	mgN/m^3
k_{hnxp}	preference coefficient of phytoplankton for ammonium	mgN/m^3
$k_{hp}(T)$	temperature-dependent organic phosphorus hydrolysis rate	/d
K_{L12}	pore water diffusion mass transfer coefficient	m/d
K_{Lb}	bottom algae light parameter	
K_{Lp}	phytoplankton light parameter	ly/d
$K_{M,Dp}$	oxygen half-saturation constant for bioturbation	gO_2/m^3
$k_{na}(T)$	temperature-dependent nitrification rate for ammonia nitrogen	/d
K_{NH4}	ammonium half-saturation constant	gN/m^3
$K_{NH4,O2}$	oxygen half-saturation constant	mgO_2/L
$k_{POC,G1}$	the mineralization rate of labile POC	d^{-1}
$k_{rb}(T)$	temperature-dependent bottom algae respiration rate	/d
$k_{rp}(T)$	temperature-dependent phytoplankton respiration/excretion rate	/d
k_{sNb}	nitrogen half-saturation constant for bottom algae	µgN/L
k_{sNp}	nitrogen half-saturation constant for phytoplankton	µgN/L
K_{socf}	parameter for oxygen dependency of fast CBOD oxidation	
K_{sodn}	parameter for oxygen dependency of denitrification	
K_{sona}	parameter for oxygen dependency of nitrification	
k_{sPb}	phosphorus half-saturation constant for bottom algae	µgP/L

k_{sPp}	phosphorus half-saturation constant for phytoplankton	μgP/L
K_w	acidity constant for dissociation of water	
Lat	latitude	radians
Llm	longitude of local meridian	degrees
localTime	local standard time	minutes
L_{sm}	longitude of standard meridian	degrees
m	optical air mass	dimensionless
mgY	mass of element Y	mg
m_i	inorganic suspended solids	mgD/L
c_p	POC	mgD/L
n	Manning roughness coefficient	
n_a	the ammonium concentration in the overlying water	mgN/m^3
n_{au}	unionized ammonia nitrogen	mgN/m^3
n_{fac}	atmospheric turbidity factor	dimensionless
$NH_{4,i}$	the concentration of total ammonium in sediment layer *i*	gN/m^3
n_n	nitrate concentration in the overlying water	mgN/m^3
n_o	organic nitrogen	mgN/m^3
$NO_{3,i}$	nitrate concentration in layer *i*	gN/m^3
n_{pai}	total number of non-point withdrawals outflows from element *i*	dimensionless
n_{psi}	total number of non-point sources inflows to element *i*	dimensionless
NSOD	the amount of oxygen demand generated by nitrification	$gO_2/m^2/d$
o	dissolved oxygen	mgO_2/L
o'_{i-1}	oxygen concentration entering an element below a dam	mgO_2/L
o_{crit}	critical oxygen concentration for sediment phosphorus sorption	gO_2/m^3
$o_s(T, \text{elev})$	saturation concentration of oxygen at temperature, *T*, and elevation above sea level, *elev*	mgO_2/L
P	wetted perimeter	m
P_{ab}	preference for ammonium as a nitrogen source for bottom algae	dimensionless
p_{ai}	total number of point withdrawals from element *i*	dimensionless
P_{ap}	preference for ammonium as a nitrogen source for phytoplankton	dimensionless
$PAR(z)$	photosynthetically active radiation (PAR) at depth *z* below water surface	ly/d
p_{atm}	atmospheric pressure	mm Hg
p_{CO_2}	atmospheric partial pressure of carbon dioxide	atm
p_i	inorganic phosphorus	μgP/L
p_o	organic phosphorus	μgP/L
$PO_{4,i}$	the concentration of total inorganic phosphorus in sediment layer *i*	gP/m^3
$POC_{2,G1}$	the concentration of the labile fraction of POC in the anaerobic sediment layer	gO_2/m^3
POC_R	reference G1 concentration for bioturbation	gC/m^3
psi	total number of point sources to element *i*	dimensionless
p_{wc}	mean daily atmospheric precipitable water content	
Q	flow	m^3/s or m^3/d

$Q_{out,i}$	total outflow from element due to point and nonpoint withdrawals	m^3/d
Q_i	outflow from element i into element $i + 1$	m^3/d
$Q_{in,i}$	total inflow into element from point and nonpoint sources	m^3/d
$Q_{npa,i,j}$	j^{th} non-point withdrawal outflow from element i	m^3/d
$Q_{nps,i,j}$	j^{th} non-point source inflow to element i	m^3/d
$Q_{pa,i,j}$	j^{th} point withdrawal outflow from element i	m^3/d
$Q_{ps,i,j}$	j^{th} point source inflow to element i	m^3/d
r	normalized radius of earth's orbit (i.e., ratio of actual earth-sun distance to mean earth-sun distance	dimensionless
r_{cndn}	ratio of oxygen equivalents lost per nitrate nitrogen that is denitrified	gO_2/gN
r_d	ratio of deficit above and below dam	dimensionless
r_{da}	the ratio of dry weight to chlorophyll *a*	gD/gC
R_L	longwave reflection coefficient	dimensionless
r_{oc}	ratio of oxygen consumed per organic carbon oxidized to carbon dioxide	gO_2/gC
r_{oca}	ratio of oxygen generated per organic carbon produced by photosynthesis when nitrate is taken up	gO_2/gC
r_{ocn}	ratio of oxygen generated per organic carbon produced by photosynthesis when ammonia is taken up	gO_2/gC
r_{on}	the ratio of oxygen to nitrogen consumed during nitrification	= 4.57 gO_2/gN
r_{on}'	the ratio of oxygen to nitrogen consumed for total conversion of ammonium to nitrogen gas via nitrification/denitrification	= 1.714 gO_2/gN
R_s	albedo or reflectivity (fraction of solar radiation that is reflected)	dimensionless
S	channel slope	dimensionless
s	chloride	mgCl/L
s	mass transfer coefficient between the water and the aerobic sediments	m/d
S_0	bottom slope	m/m
$S_{b,i}$	sources and sinks of constituent due to reactions for bottom algae	$gC/m^2/d$
S_i	sources and sinks of constituent due to reactions and mass transfer mechanisms for water constituents	$g/m^3/d$ or $mg/m^3/d$
SOD	the sediment oxygen demand	$gO_2/m^2/d$
SOD_s	the supplemental sediment oxygen demand	$gO_2/m^2/d$
SOD_t	total sediment oxygen demand = $SOD + SODs$	$gO_2/m^2/d$
s_s	channel side slope	m/m
t	time	d
T	water temperature	°C
$T_{w,f}$	water temperature	°F
T_a	absolute temperature	K
T_{air}	air temperature	°C
$T_{air,f}$	air temperature	°F
T_d	dew-point temperature	°C

T_i	temperature in element *i*	°C
timezone	time zone indicates local time zone in relation to Greenwich Mean Time (GMT) (negative in western hemisphere)	hours
$T_{nps,i,j}$	j^{th} non-point source temperature for element *i*	°C
$T_{ps,i,j}$	j^{th} point source temperature for element *i*	°C
trueSolarTime	time determined from actual position of the sun in the sky	minutes
$T_{s,i}$	temperature of bottom sediment	°C
t_{sr}	time of sunrise	Hr
t_{ss}	time of sunset	Hr
T_{std}	standard time	Hr
$t_{t,i}$	travel time from headwater to end of element *i*	D
U	mean velocity	m/s
U^*	shear velocity	m/s
U_w	wind speed	m/s
$U_{w,mph}$	wind speed	mph
$U_{w,z}$	wind speed at height z_w above water surface	m/s
v_a	phytoplankton settling velocity	m/d
v_{poc}	POC settling velocity	m/d
v_i	inorganic suspended solids settling velocity	m/d
V_i	volume of i^{th} element	m^3
v_p	pathogen settling velocity	m/d
v_{pc}	non-living particulate organic carbon settling velocity	m/d
W_0	solar constant	2851 cal/cm^2/d
w_2	burial velocity	m/d
$W_{h,i}$	net heat load from point and non-point sources into element *i*	cal/d
W_i	external loading of constituent to element *i*	g/d or mg/d
X	pathogen	cfu/100 mL
z_w	height above water for wind speed measurements	m

Greek:

Symbol	**Definition**	Units
α	depth rating curve coefficient	dimensionless
α	sun's altitude	radians
α_d	sun's altitude	degrees
α_s	sediment thermal diffusivity	cm^2/s
α_0	fraction of total inorganic carbon in carbon dioxide	dimensionless
α_1	fraction of total inorganic carbon in bicarbonate	dimensionless
α_2	fraction of total inorganic carbon in carbonate	dimensionless
α_i	effect of inorganic suspended solids on light attenuation	L/mgD/m
α_o	effect of particulate organic matter on light attenuation	L/mgD/m
α_p	linear effect of chlorophyll on light attenuation	L/μgA/m
α_{pn}	non-linear effect of chlorophyll on light attenuation	$(L/\mu gA)^{2/3}/m$
α_{path}	pathogen light efficiency factor	dimensionless
β	depth rating curve exponent	dimensionless

δ	solar declination	radians
$\Delta\pi PO_{4,1}$	factor that increases the aerobic sediment layer partition coefficient relative to the anaerobic coefficient	dimensionless
$\Delta\theta_v$	virtual temperature difference between the water and air	°F
Δts	difference between standard and local civic time	hr
Δx_i	length of *i*th element	m
ε	emissivity of water	dimensionless
ε_{clear}	emissivity of longwave radiation from the sky with no clouds	0-1
ε_{sky}	emissivity of longwave radiation from the sky with clouds	0-1
ε_a	estimated error	%
ϕ_{Lb}	bottom algae light attenuation	0-1
ϕ_{Lp}	phytoplankton light attenuation	0-1
ϕ_{Nb}	bottom algae nutrient attenuation factor	0-1
ϕ_{Np}	phytoplankton nutrient attenuation factor	0-1
μ_p	phytoplankton photosynthesis rate	/d
ρ	density of water	g/cm^3
σ	Stefan-Boltzmann constant	11.7×10^{-8} cal/(cm^2 d K^4)
τ	local hour angle of sun	radians
τ_i	residence time of *i*th element	d
θ	temperature coefficient for zero and first-order reactions	dimensionless
θ_{am}	elevation adjusted optical air mass	
ω_{12}	the bioturbation mass transfer coefficient between the sediment layers	m/d
π_{ai}	the partition coefficient for ammonium in sediment layer *i*	m^3/gD
π_{pi}	the partition coefficient for inorganic phosphorus in sediment layer *i*	m^3/gD
θ_{CH4}	temperature correction factor for sediment methane oxidation	dimensionless
θ_{Dd}	temperature coefficient for porewater diffusion	dimensionless
θ_{Dp}	temperature coefficient for bioturbation diffusion	dimensionless
θ_{NH4}	temperature correction factor for sediment nitrification	dimensionless
θ_{NO3}	sediment denitrification temperature correction factor	dimensionless
$\theta_{POC,G1}$	temperature correction factor for labile POC mineralization	dimensionless
$\kappa_{CH4,1}$	the reaction velocity for methane oxidation in the aerobic sediments	m/d
$\kappa_{NH4,1}$	the reaction velocity for nitrification in the aerobic sediments	m/d
$\kappa_{NO3,i}$	denitrification reaction velocity sediment layer *i*	m/d

APPENDIX B: SOLAR POSITION, SUNRISE, AND SUNSET CALCULATIONS

The sunrise/sunset and solar position functions are a VBA translation of NOAA's sunrise/sunset calculator and NOAA's solar position calculator at the following web pages:

- http://www.srrb.noaa.gov/highlights/sunrise/sunrise.html
- http://www.srrb.noaa.gov/highlights/sunrise/azel.html

The calculations in the NOAA Sunrise/Sunset and Solar Position Calculators are based on equations from Astronomical Algorithms, by Jean Meeus. The sunrise and sunset results have been verified by NOAA to be accurate to within a minute for locations between +/- 72° latitude, and within 10 minutes outside of those latitudes.

Five main functions are included for use from Excel worksheets or VBA programs:

- sunrise (lat, lon, year, month, day, timezone, dlstime) calculates the local time of sunrise for a location and date
- solarnoon (lat, lon, year, month, day, timezone, dlstime) calculates the local time of solar noon for a location and date (the time when the sun crosses the meridian)
- sunset (lat, lon, year, month, day, timezone, dlstime) calculates the local time of sunset for a location and date
- solarazimuth (lat, lon, year, month, day, hour, minute, second, timezone, dlstime) calculates the solar azimuth for a location, date, and time (degrees clockwise from north to the point on the horizon directly below the sun)
- solarelevation (lat, lon, year, month, day, hour, minute, second, timezone, dlstime) calculates the solar elevation for a location, date, and time (degrees vertically from horizon to the sun)

A subroutine is also provided that calculates solar azimuth (az), solar elevation (el):

- solarposition (lat, lon, year, month, day, hour, minute, second, timezone, dlstime, az, el, earthRadiusVector)

The sign convention for the main functions and subroutine is:

- positive latitude decimal degrees for northern hemisphere
- negative longitude degrees for western hemisphere
- negative time zone hours for western hemisphere

The Excel/VBA functions and subroutines for solar position, sunrise, and sunset times are as follows:

```
Option Explicit

Function radToDeg(angleRad)
'// Convert radian angle to degrees

        radToDeg = (180# * angleRad / Application.WorksheetFunction.Pi())

End Function

Function degToRad(angleDeg)
'// Convert degree angle to radians

        degToRad = (Application.WorksheetFunction.Pi() * angleDeg / 180#)
```

```
End Function

Function calcJD(year, month, day)

'*********************************************************************/
'* Name:    calcJD
'* Type:    Function
'* Purpose: Julian day from calendar day
'* Arguments:
'*   year : 4 digit year
'*   month: January = 1
'*   day  : 1 - 31
'* Return value:
'*   The Julian day corresponding to the date
'* Note:
'*   Number is returned for start of day.  Fractional days should be
'*   added later.
'*********************************************************************/

Dim A As Double, B As Double, jd As Double

        If (month <= 2) Then
         year = year - 1
         month = month + 12
        End If

        A = Application.WorksheetFunction.Floor(year / 100, 1)
        B = 2 - A + Application.WorksheetFunction.Floor(A / 4, 1)

        jd = Application.WorksheetFunction.Floor(365.25 * (year + 4716), 1) + _
             Application.WorksheetFunction.Floor(30.6001 * (month + 1), 1) + day + B -
1524.5
        calcJD = jd

'gp put the year and month back where they belong
        If month = 13 Then
         month = 1
         year = year + 1
        End If
        If month = 14 Then
         month = 2
         year = year + 1
        End If

End Function

Function calcTimeJulianCent(jd)

'*********************************************************************/
'* Name:    calcTimeJulianCent
'* Type:    Function
'* Purpose: convert Julian Day to centuries since J2000.0.
'* Arguments:
'*   jd : the Julian Day to convert
'* Return value:
'*   the T value corresponding to the Julian Day
'*********************************************************************/

Dim t As Double

        t = (jd - 2451545#) / 36525#
        calcTimeJulianCent = t
```

```
End Function

Function calcJDFromJulianCent(t)

'*************************************************************************/
'* Name:    calcJDFromJulianCent
'* Type:    Function
'* Purpose: convert centuries since J2000.0 to Julian Day.
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   the Julian Day corresponding to the t value
'*************************************************************************/

Dim jd As Double

        jd = t * 36525# + 2451545#
        calcJDFromJulianCent = jd

End Function

Function calcGeomMeanLongSun(t)

'*************************************************************************/
'* Name:    calGeomMeanLongSun
'* Type:    Function
'* Purpose: calculate the Geometric Mean Longitude of the Sun
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   the Geometric Mean Longitude of the Sun in degrees
'*************************************************************************/

Dim l0 As Double

        l0 = 280.46646 + t * (36000.76983 + 0.0003032 * t)
        Do
           If (l0 <= 360) And (l0 >= 0) Then Exit Do
           If l0 > 360 Then l0 = l0 - 360
           If l0 < 0 Then l0 = l0 + 360
        Loop

        calcGeomMeanLongSun = l0

End Function

Function calcGeomMeanAnomalySun(t)

'*************************************************************************/
'* Name:    calGeomAnomalySun
'* Type:    Function
'* Purpose: calculate the Geometric Mean Anomaly of the Sun
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   the Geometric Mean Anomaly of the Sun in degrees
'*************************************************************************/

Dim m As Double

        m = 357.52911 + t * (35999.05029 - 0.0001537 * t)
        calcGeomMeanAnomalySun = m
```

```
End Function

Function calcEccentricityEarthOrbit(t)

'*************************************************************************/
'* Name:    calcEccentricityEarthOrbit
'* Type:    Function
'* Purpose: calculate the eccentricity of earth's orbit
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   the unitless eccentricity
'*************************************************************************/

Dim e As Double

        e = 0.016708634 - t * (0.000042037 + 0.0000001267 * t)
        calcEccentricityEarthOrbit = e

End Function

Function calcSunEqOfCenter(t)

'*************************************************************************/
'* Name:    calcSunEqOfCenter
'* Type:    Function
'* Purpose: calculate the equation of center for the sun
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   in degrees
'*************************************************************************/

Dim m As Double, mrad As Double, sinm As Double, sin2m As Double, sin3m As Double
Dim c As Double

        m = calcGeomMeanAnomalySun(t)

        mrad = degToRad(m)
        sinm = Sin(mrad)
        sin2m = Sin(mrad + mrad)
        sin3m = Sin(mrad + mrad + mrad)

        c = sinm * (1.914602 - t * (0.004817 + 0.000014 * t)) _
            + sin2m * (0.019993 - 0.000101 * t) + sin3m * 0.000289

        calcSunEqOfCenter = c

End Function

Function calcSunTrueLong(t)

'*************************************************************************/
'* Name:    calcSunTrueLong
'* Type:    Function
'* Purpose: calculate the true longitude of the sun
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   sun's true longitude in degrees
'*************************************************************************/

Dim l0 As Double, c As Double, O As Double
```

```
        l0 = calcGeomMeanLongSun(t)
        c = calcSunEqOfCenter(t)

        O = l0 + c
        calcSunTrueLong = O

End Function

Function calcSunTrueAnomaly(t)

'*************************************************************************/
'* Name:    calcSunTrueAnomaly (not used by sunrise, solarnoon, sunset)
'* Type:    Function
'* Purpose: calculate the true anamoly of the sun
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   sun's true anamoly in degrees
'*************************************************************************/

Dim m As Double, c As Double, v As Double

        m = calcGeomMeanAnomalySun(t)
        c = calcSunEqOfCenter(t)

        v = m + c
        calcSunTrueAnomaly = v

End Function

Function calcSunRadVector(t)

'*************************************************************************/
'* Name:    calcSunRadVector (not used by sunrise, solarnoon, sunset)
'* Type:    Function
'* Purpose: calculate the distance to the sun in AU
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   sun radius vector in AUs
'*************************************************************************/

Dim v As Double, e As Double, r As Double

        v = calcSunTrueAnomaly(t)
        e = calcEccentricityEarthOrbit(t)

        r = (1.000001018 * (1 - e * e)) / (1 + e * Cos(degToRad(v)))
        calcSunRadVector = r

End Function

Function calcSunApparentLong(t)

'*************************************************************************/
'* Name:    calcSunApparentLong (not used by sunrise, solarnoon, sunset)
'* Type:    Function
'* Purpose: calculate the apparent longitude of the sun
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   sun's apparent longitude in degrees
```

```
'*************************************************************************/

Dim O As Double, omega As Double, lambda As Double

        O = calcSunTrueLong(t)

        omega = 125.04 - 1934.136 * t
        lambda = O - 0.00569 - 0.00478 * Sin(degToRad(omega))
        calcSunApparentLong = lambda

End Function

Function calcMeanObliquityOfEcliptic(t)

'*************************************************************************/
'* Name:    calcMeanObliquityOfEcliptic
'* Type:    Function
'* Purpose: calculate the mean obliquity of the ecliptic
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   mean obliquity in degrees
'*************************************************************************/

Dim seconds As Double, e0 As Double

        seconds = 21.448 - t * (46.815 + t * (0.00059 - t * (0.001813)))
        e0 = 23# + (26# + (seconds / 60#)) / 60#
        calcMeanObliquityOfEcliptic = e0

End Function

Function calcObliquityCorrection(t)

'*************************************************************************/
'* Name:    calcObliquityCorrection
'* Type:    Function
'* Purpose: calculate the corrected obliquity of the ecliptic
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   corrected obliquity in degrees
'*************************************************************************/

Dim e0 As Double, omega As Double, e As Double

        e0 = calcMeanObliquityOfEcliptic(t)

        omega = 125.04 - 1934.136 * t
        e = e0 + 0.00256 * Cos(degToRad(omega))
        calcObliquityCorrection = e

End Function

Function calcSunRtAscension(t)

'*************************************************************************/
'* Name:    calcSunRtAscension (not used by sunrise, solarnoon, sunset)
'* Type:    Function
'* Purpose: calculate the right ascension of the sun
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
```

```
'*   sun's right ascension in degrees
'*************************************************************************/

Dim e As Double, lambda As Double, tananum As Double, tanadenom As Double
Dim alpha As Double

        e = calcObliquityCorrection(t)
        lambda = calcSunApparentLong(t)

        tananum = (Cos(degToRad(e)) * Sin(degToRad(lambda)))
        tanadenom = (Cos(degToRad(lambda)))

'original NOAA code using javascript Math.Atan2(y,x) convention:
'        var alpha = radToDeg(Math.atan2(tananum, tanadenom));
'        alpha = radToDeg(Application.WorksheetFunction.Atan2(tananum, tanadenom))

'translated using Excel VBA Application.WorksheetFunction.Atan2(x,y) convention:
        alpha = radToDeg(Application.WorksheetFunction.Atan2(tanadenom, tananum))

        calcSunRtAscension = alpha

End Function

Function calcSunDeclination(t)

'*************************************************************************/

'* Name:    calcSunDeclination
'* Type:    Function
'* Purpose: calculate the declination of the sun
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   sun's declination in degrees
'*************************************************************************/

Dim e As Double, lambda As Double, sint As Double, theta As Double

        e = calcObliquityCorrection(t)
        lambda = calcSunApparentLong(t)

        sint = Sin(degToRad(e)) * Sin(degToRad(lambda))
        theta = radToDeg(Application.WorksheetFunction.Asin(sint))
        calcSunDeclination = theta

End Function

Function calcEquationOfTime(t)

'*************************************************************************/
'* Name:    calcEquationOfTime
'* Type:    Function
'* Purpose: calculate the difference between true solar time and mean
'*     solar time
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'* Return value:
'*   equation of time in minutes of time
'*************************************************************************/

Dim epsilon As Double, l0 As Double, e As Double, m As Double
Dim y As Double, sin2l0 As Double, sinm As Double
Dim cos2l0 As Double, sin4l0 As Double, sin2m As Double, Etime As Double
```

```
        epsilon = calcObliquityCorrection(t)
        l0 = calcGeomMeanLongSun(t)
        e = calcEccentricityEarthOrbit(t)
        m = calcGeomMeanAnomalySun(t)

        y = Tan(degToRad(epsilon) / 2#)
        y = y ^ 2

        sin2l0 = Sin(2# * degToRad(l0))
        sinm = Sin(degToRad(m))
        cos2l0 = Cos(2# * degToRad(l0))
        sin4l0 = Sin(4# * degToRad(l0))
        sin2m = Sin(2# * degToRad(m))

        Etime = y * sin2l0 - 2# * e * sinm + 4# * e * y * sinm * cos2l0 _
                - 0.5 * y * y * sin4l0 - 1.25 * e * e * sin2m

        calcEquationOfTime = radToDeg(Etime) * 4#

End Function

Function calcHourAngleSunrise(lat, SolarDec)

'*************************************************************************/
'* Name:    calcHourAngleSunrise
'* Type:    Function
'* Purpose: calculate the hour angle of the sun at sunrise for the
'*          latitude
'* Arguments:
'*   lat : latitude of observer in degrees
'* solarDec : declination angle of sun in degrees
'* Return value:
'*   hour angle of sunrise in radians
'*************************************************************************/

Dim latrad As Double, sdRad As Double, HAarg As Double, HA As Double

        latrad = degToRad(lat)
        sdRad = degToRad(SolarDec)

        HAarg = (Cos(degToRad(90.833)) / (Cos(latrad) * Cos(sdRad)) - Tan(latrad) *
Tan(sdRad))

        HA = (Application.WorksheetFunction.Acos(Cos(degToRad(90.833)) _
              / (Cos(latrad) * Cos(sdRad)) - Tan(latrad) * Tan(sdRad)))

        calcHourAngleSunrise = HA

End Function

Function calcHourAngleSunset(lat, SolarDec)

'*************************************************************************/
'* Name:    calcHourAngleSunset
'* Type:    Function
'* Purpose: calculate the hour angle of the sun at sunset for the
'*          latitude
'* Arguments:
'*   lat : latitude of observer in degrees
'* solarDec : declination angle of sun in degrees
'* Return value:
'*   hour angle of sunset in radians
```

```
'************************************************************************/

Dim latrad As Double, sdRad As Double, HAarg As Double, HA As Double

        latrad = degToRad(lat)
        sdRad = degToRad(SolarDec)

        HAarg = (Cos(degToRad(90.833)) / (Cos(latrad) * Cos(sdRad)) - Tan(latrad) *
Tan(sdRad))

        HA = (Application.WorksheetFunction.Acos(Cos(degToRad(90.833)) _
               / (Cos(latrad) * Cos(sdRad)) - Tan(latrad) * Tan(sdRad)))

        calcHourAngleSunset = -HA

End Function

Function calcSunriseUTC(jd, Latitude, longitude)

'************************************************************************/
'* Name:    calcSunriseUTC
'* Type:    Function
'* Purpose: calculate the Universal Coordinated Time (UTC) of sunrise
'*         for the given day at the given location on earth
'* Arguments:
'*   JD  : julian day
'*   latitude : latitude of observer in degrees
'*   longitude : longitude of observer in degrees
'* Return value:
'*   time in minutes from zero Z
'************************************************************************/

Dim t As Double, eqtime As Double, SolarDec As Double, hourangle As Double
Dim delta As Double, timeDiff As Double, timeUTC As Double
Dim newt As Double

        t = calcTimeJulianCent(jd)

'        // *** First pass to approximate sunrise

        eqtime = calcEquationOfTime(t)
        SolarDec = calcSunDeclination(t)
        hourangle = calcHourAngleSunrise(Latitude, SolarDec)

        delta = longitude - radToDeg(hourangle)
        timeDiff = 4 * delta
' in minutes of time
        timeUTC = 720 + timeDiff - eqtime
' in minutes

' *** Second pass includes fractional jday in gamma calc

        newt = calcTimeJulianCent(calcJDFromJulianCent(t) + timeUTC / 1440#)
        eqtime = calcEquationOfTime(newt)
        SolarDec = calcSunDeclination(newt)
        hourangle = calcHourAngleSunrise(Latitude, SolarDec)
        delta = longitude - radToDeg(hourangle)
        timeDiff = 4 * delta
        timeUTC = 720 + timeDiff - eqtime
' in minutes

        calcSunriseUTC = timeUTC
```

```
End Function

Function calcSolNoonUTC(t, longitude)

'*************************************************************************/
'* Name:    calcSolNoonUTC
'* Type:    Function
'* Purpose: calculate the Universal Coordinated Time (UTC) of solar
'*     noon for the given day at the given location on earth
'* Arguments:
'*   t : number of Julian centuries since J2000.0
'*   longitude : longitude of observer in degrees
'* Return value:
'*   time in minutes from zero Z
'*************************************************************************/

Dim newt As Double, eqtime As Double, solarNoonDec As Double, solNoonUTC As Double

        newt = calcTimeJulianCent(calcJDFromJulianCent(t) + 0.5 + longitude / 360#)

        eqtime = calcEquationOfTime(newt)
        solarNoonDec = calcSunDeclination(newt)
        solNoonUTC = 720 + (longitude * 4) - eqtime

        calcSolNoonUTC = solNoonUTC

End Function

Function calcSunsetUTC(jd, Latitude, longitude)

'*************************************************************************/
'* Name:    calcSunsetUTC
'* Type:    Function
'* Purpose: calculate the Universal Coordinated Time (UTC) of sunset
'*         for the given day at the given location on earth
'* Arguments:
'*   JD  : julian day
'*   latitude : latitude of observer in degrees
'*   longitude : longitude of observer in degrees
'* Return value:
'*   time in minutes from zero Z
'*************************************************************************/

Dim t As Double, eqtime As Double, SolarDec As Double, hourangle As Double
Dim delta As Double, timeDiff As Double, timeUTC As Double
Dim newt As Double

        t = calcTimeJulianCent(jd)

'        // First calculates sunrise and approx length of day

        eqtime = calcEquationOfTime(t)
        SolarDec = calcSunDeclination(t)
        hourangle = calcHourAngleSunset(Latitude, SolarDec)

        delta = longitude - radToDeg(hourangle)
        timeDiff = 4 * delta
        timeUTC = 720 + timeDiff - eqtime

'        // first pass used to include fractional day in gamma calc

        newt = calcTimeJulianCent(calcJDFromJulianCent(t) + timeUTC / 1440#)
        eqtime = calcEquationOfTime(newt)
```

```
        SolarDec = calcSunDeclination(newt)
        hourangle = calcHourAngleSunset(Latitude, SolarDec)

        delta = longitude - radToDeg(hourangle)
        timeDiff = 4 * delta
        timeUTC = 720 + timeDiff - eqtime
'        // in minutes

        calcSunsetUTC = timeUTC

End Function

Function sunrise(lat, lon, year, month, day, timezone, dlstime)

'**********************************************************************/
'* Name:    sunrise
'* Type:    Main Function called by spreadsheet
'* Purpose: calculate time of sunrise  for the entered date
'*     and location.
'* For latitudes greater than 72 degrees N and S, calculations are
'* accurate to within 10 minutes.  For latitudes less than +/- 72°
'* accuracy is approximately one minute.
'* Arguments:
'   latitude = latitude (decimal degrees)
'   longitude = longitude (decimal degrees)
'    NOTE: longitude is negative for western hemisphere for input cells
'          in the spreadsheet for calls to the functions named
'          sunrise, solarnoon, and sunset.  Those functions convert the
'          longitude to positive for the western hemisphere for calls to
'          other functions using the original sign convention
'          from the NOAA javascript code.
'   year = year
'   month = month
'   day = day
'   timezone = time zone hours relative to GMT/UTC (hours)
'   dlstime = daylight savings time (0 = no, 1 = yes) (hours)
'* Return value:
'*   sunrise time in local time (days)
'**********************************************************************/

Dim longitude As Double, Latitude As Double, jd As Double
Dim riseTimeGMT As Double, riseTimeLST As Double

' change sign convention for longitude from negative to positive in western hemisphere
            longitude = lon * -1
            Latitude = lat
            If (Latitude > 89.8) Then Latitude = 89.8
            If (Latitude < -89.8) Then Latitude = -89.8

            jd = calcJD(year, month, day)

'            // Calculate sunrise for this date
            riseTimeGMT = calcSunriseUTC(jd, Latitude, longitude)

'            //  adjust for time zone and daylight savings time in minutes
            riseTimeLST = riseTimeGMT + (60 * timezone) + (dlstime * 60)

'            //  convert to days
            sunrise = riseTimeLST / 1440

End Function

Function solarnoon(lat, lon, year, month, day, timezone, dlstime)
```

```
'*********************************************************************/
'* Name:    solarnoon
'* Type:    Main Function called by spreadsheet
'* Purpose: calculate the Universal Coordinated Time (UTC) of solar
'*     noon for the given day at the given location on earth
'* Arguments:
'    year
'    month
'    day
'*   longitude : longitude of observer in degrees
'    NOTE: longitude is negative for western hemisphere for input cells
'          in the spreadsheet for calls to the functions named
'          sunrise, solarnoon, and sunset.  Those functions convert the
'          longitude to positive for the western hemisphere for calls to
'          other functions using the original sign convention
'          from the NOAA javascript code.
'* Return value:
'*   time of solar noon in local time days
'*********************************************************************/

Dim longitude As Double, Latitude As Double, jd As Double
Dim t As Double, newt As Double, eqtime As Double
Dim solarNoonDec As Double, solNoonUTC As Double

' change sign convention for longitude from negative to positive in western hemisphere
        longitude = lon * -1
        Latitude = lat
        If (Latitude > 89.8) Then Latitude = 89.8
        If (Latitude < -89.8) Then Latitude = -89.8

        jd = calcJD(year, month, day)
        t = calcTimeJulianCent(jd)

        newt = calcTimeJulianCent(calcJDFromJulianCent(t) + 0.5 + longitude / 360#)

        eqtime = calcEquationOfTime(newt)
        solarNoonDec = calcSunDeclination(newt)
        solNoonUTC = 720 + (longitude * 4) - eqtime

'            //  adjust for time zone and daylight savings time in minutes
        solarnoon = solNoonUTC + (60 * timezone) + (dlstime * 60)

'            //  convert to days
        solarnoon = solarnoon / 1440

End Function

Function sunset(lat, lon, year, month, day, timezone, dlstime)

'*********************************************************************/
'* Name:    sunset
'* Type:    Main Function called by spreadsheet
'* Purpose: calculate time of sunrise and sunset for the entered date
'*     and location.
'* For latitudes greater than 72 degrees N and S, calculations are
'* accurate to within 10 minutes.  For latitudes less than +/- 72°
'* accuracy is approximately one minute.
'* Arguments:
'   latitude = latitude (decimal degrees)
'   longitude = longitude (decimal degrees)
'    NOTE: longitude is negative for western hemisphere for input cells
'          in the spreadsheet for calls to the functions named
```

```
'          sunrise, solarnoon, and sunset.  Those functions convert the
'          longitude to positive for the western hemisphere for calls to
'          other functions using the original sign convention
'          from the NOAA javascript code.
'   year = year
'   month = month
'   day = day
'   timezone = time zone hours relative to GMT/UTC (hours)
'   dlstime = daylight savings time (0 = no, 1 = yes) (hours)
'* Return value:
'*   sunset time in local time (days)
'*********************************************************************/

Dim longitude As Double, Latitude As Double, jd As Double
Dim setTimeGMT As Double, setTimeLST As Double

' change sign convention for longitude from negative to positive in western hemisphere
            longitude = lon * -1
            Latitude = lat
            If (Latitude > 89.8) Then Latitude = 89.8
            If (Latitude < -89.8) Then Latitude = -89.8

            jd = calcJD(year, month, day)

'           // Calculate sunset for this date
            setTimeGMT = calcSunsetUTC(jd, Latitude, longitude)

'            //  adjust for time zone and daylight savings time in minutes
            setTimeLST = setTimeGMT + (60 * timezone) + (dlstime * 60)

'            //  convert to days
            sunset = setTimeLST / 1440

End Function

Function solarazimuth(lat, lon, year, month, day, _
                      hours, minutes, seconds, timezone, dlstime)

'*********************************************************************/
'* Name:    solarazimuth
'* Type:    Main Function
'* Purpose: calculate solar azimuth (deg from north) for the entered
'*          date, time and location.  Returns -999999 if darker than twilight
'*
'* Arguments:
'*   latitude, longitude, year, month, day, hour, minute, second,
'*   timezone, daylightsavingstime
'* Return value:
'*   solar azimuth in degrees from north
'*
'* Note: solarelevation and solarazimuth functions are identical
'*       and could be converted to a VBA subroutine that would return
'*       both values.
'*
'*********************************************************************/

Dim longitude As Double, Latitude As Double
Dim Zone As Double, daySavings As Double
Dim hh As Double, mm As Double, ss As Double, timenow As Double
Dim jd As Double, t As Double, r As Double
Dim alpha As Double, theta As Double, Etime As Double, eqtime As Double
Dim SolarDec As Double, earthRadVec As Double, solarTimeFix As Double
Dim trueSolarTime As Double, hourangle As Double, harad As Double
```

```
Dim csz As Double, zenith As Double, azDenom As Double, azRad As Double
Dim azimuth As Double, exoatmElevation As Double
Dim step1 As Double, step2 As Double, step3 As Double
Dim refractionCorrection As Double, Te As Double, solarzen As Double

            longitude = lon * -1
            Latitude = lat
            If (Latitude > 89.8) Then Latitude = 89.8
            If (Latitude < -89.8) Then Latitude = -89.8

            Zone = timezone * -1
            daySavings = dlstime * 60
            hh = hours - (daySavings / 60)
            mm = minutes
            ss = seconds

'//    timenow is GMT time for calculation in hours since 0Z
            timenow = hh + mm / 60 + ss / 3600 + Zone

            jd = calcJD(year, month, day)
            t = calcTimeJulianCent(jd + timenow / 24#)
            r = calcSunRadVector(t)
            alpha = calcSunRtAscension(t)
            theta = calcSunDeclination(t)
            Etime = calcEquationOfTime(t)

            eqtime = Etime
            SolarDec = theta '//    in degrees
            earthRadVec = r

            solarTimeFix = eqtime - 4# * longitude + 60# * Zone
            trueSolarTime = hh * 60# + mm + ss / 60# + solarTimeFix
            '//    in minutes

            Do While (trueSolarTime > 1440)
                trueSolarTime = trueSolarTime - 1440
            Loop

            hourangle = trueSolarTime / 4# - 180#
            '//    Thanks to Louis Schwarzmayr for the next line:
            If (hourangle < -180) Then hourangle = hourangle + 360#

            harad = degToRad(hourangle)

            csz = Sin(degToRad(Latitude)) * _
                  Sin(degToRad(SolarDec)) + _
                  Cos(degToRad(Latitude)) * _
                  Cos(degToRad(SolarDec)) * Cos(harad)

            If (csz > 1#) Then
                csz = 1#
            ElseIf (csz < -1#) Then
                csz = -1#
            End If

            zenith = radToDeg(Application.WorksheetFunction.Acos(csz))

            azDenom = (Cos(degToRad(Latitude)) * Sin(degToRad(zenith)))

            If (Abs(azDenom) > 0.001) Then
                azRad = ((Sin(degToRad(Latitude)) * _
                    Cos(degToRad(zenith))) - _
                    Sin(degToRad(SolarDec))) / azDenom
```

```
                If (Abs(azRad) > 1#) Then
                    If (azRad < 0) Then
                        azRad = -1#
                    Else
                        azRad = 1#
                    End If
                End If

                azimuth = 180# - radToDeg(Application.WorksheetFunction.Acos(azRad))

                If (hourangle > 0#) Then
                    azimuth = -azimuth
                End If
            Else
                If (Latitude > 0#) Then
                    azimuth = 180#
                Else
                    azimuth = 0#
                End If
            End If
            If (azimuth < 0#) Then
                azimuth = azimuth + 360#
            End If

            exoatmElevation = 90# - zenith

            If (exoatmElevation > 85#) Then
                refractionCorrection = 0#
            Else
                Te = Tan(degToRad(exoatmElevation))
                If (exoatmElevation > 5#) Then
                    refractionCorrection = 58.1 / Te - 0.07 / (Te * Te * Te) + _
                        0.000086 / (Te * Te * Te * Te * Te)
                ElseIf (exoatmElevation > -0.575) Then
                    step1 = (-12.79 + exoatmElevation * 0.711)
                    step2 = (103.4 + exoatmElevation * (step1))
                    step3 = (-518.2 + exoatmElevation * (step2))
                    refractionCorrection = 1735# + exoatmElevation * (step3)
                Else
                    refractionCorrection = -20.774 / Te
                End If
                refractionCorrection = refractionCorrection / 3600#
            End If

            solarzen = zenith - refractionCorrection
            solarazimuth = azimuth

End Function

Function solarelevation(lat, lon, year, month, day, _
                    hours, minutes, seconds, timezone, dlstime)

'*************************************************************************/
'* Name:    solarazimuth
'* Type:    Main Function
'* Purpose: calculate solar azimuth (deg from north) for the entered
'*          date, time and location.  Returns -999999 if darker than twilight
'*
'* Arguments:
'*   latitude, longitude, year, month, day, hour, minute, second,
'*   timezone, daylightsavingstime
'* Return value:
'*   solar azimuth in degrees from north
```

```
'*
'* Note: solarelevation and solarazimuth functions are identical
'*       and could converted to a VBA subroutine that would return
'*       both values.
'*
'*********************************************************************/

Dim longitude As Double, Latitude As Double
Dim Zone As Double, daySavings As Double
Dim hh As Double, mm As Double, ss As Double, timenow As Double
Dim jd As Double, t As Double, r As Double
Dim alpha As Double, theta As Double, Etime As Double, eqtime As Double
Dim SolarDec As Double, earthRadVec As Double, solarTimeFix As Double
Dim trueSolarTime As Double, hourangle As Double, harad As Double
Dim csz As Double, zenith As Double, azDenom As Double, azRad As Double
Dim azimuth As Double, exoatmElevation As Double
Dim step1 As Double, step2 As Double, step3 As Double
Dim refractionCorrection As Double, Te As Double, solarzen As Double

            longitude = lon * -1
            Latitude = lat
            If (Latitude > 89.8) Then Latitude = 89.8
            If (Latitude < -89.8) Then Latitude = -89.8

            Zone = timezone * -1
            daySavings = dlstime * 60
            hh = hours - (daySavings / 60)
            mm = minutes
            ss = seconds

'//    timenow is GMT time for calculation in hours since 0Z
            timenow = hh + mm / 60 + ss / 3600 + Zone

            jd = calcJD(year, month, day)
            t = calcTimeJulianCent(jd + timenow / 24#)
            r = calcSunRadVector(t)
            alpha = calcSunRtAscension(t)
            theta = calcSunDeclination(t)
            Etime = calcEquationOfTime(t)

            eqtime = Etime
            SolarDec = theta '//    in degrees
            earthRadVec = r

            solarTimeFix = eqtime - 4# * longitude + 60# * Zone
            trueSolarTime = hh * 60# + mm + ss / 60# + solarTimeFix
            '//    in minutes

            Do While (trueSolarTime > 1440)
                trueSolarTime = trueSolarTime - 1440
            Loop

            hourangle = trueSolarTime / 4# - 180#
            '//    Thanks to Louis Schwarzmayr for the next line:
            If (hourangle < -180) Then hourangle = hourangle + 360#

            harad = degToRad(hourangle)

            csz = Sin(degToRad(Latitude)) * _
                  Sin(degToRad(SolarDec)) + _
                  Cos(degToRad(Latitude)) * _
                  Cos(degToRad(SolarDec)) * Cos(harad)
```

```
    If (csz > 1#) Then
        csz = 1#
    ElseIf (csz < -1#) Then
        csz = -1#
    End If

    zenith = radToDeg(Application.WorksheetFunction.Acos(csz))

    azDenom = (Cos(degToRad(Latitude)) * Sin(degToRad(zenith)))

    If (Abs(azDenom) > 0.001) Then
        azRad = ((Sin(degToRad(Latitude)) * _
            Cos(degToRad(zenith))) - _
            Sin(degToRad(SolarDec))) / azDenom
        If (Abs(azRad) > 1#) Then
            If (azRad < 0) Then
                azRad = -1#
            Else
                azRad = 1#
            End If
        End If

        azimuth = 180# - radToDeg(Application.WorksheetFunction.Acos(azRad))

        If (hourangle > 0#) Then
            azimuth = -azimuth
        End If
    Else
        If (Latitude > 0#) Then
            azimuth = 180#
        Else
            azimuth = 0#
        End If
    End If
    If (azimuth < 0#) Then
        azimuth = azimuth + 360#
    End If

    exoatmElevation = 90# - zenith

    If (exoatmElevation > 85#) Then
        refractionCorrection = 0#
    Else
        Te = Tan(degToRad(exoatmElevation))
        If (exoatmElevation > 5#) Then
            refractionCorrection = 58.1 / Te - 0.07 / (Te * Te * Te) + _
                0.000086 / (Te * Te * Te * Te * Te)
        ElseIf (exoatmElevation > -0.575) Then
            step1 = (-12.79 + exoatmElevation * 0.711)
            step2 = (103.4 + exoatmElevation * (step1))
            step3 = (-518.2 + exoatmElevation * (step2))
            refractionCorrection = 1735# + exoatmElevation * (step3)
        Else
            refractionCorrection = -20.774 / Te
        End If
        refractionCorrection = refractionCorrection / 3600#
    End If

    solarzen = zenith - refractionCorrection
    solarelevation = 90# - solarzen

End Function
```

```
Sub solarposition(lat, lon, year, month, day, _
        hours, minutes, seconds, timezone, dlstime, _
        solarazimuth, solarelevation, earthRadVec)

'*************************************************************************/
'* Name:    solarposition
'* Type:    Subroutine
'* Purpose: calculate solar azimuth (deg from north)
'*          and elevation (deg from horizon) for the entered
'*          date, time and location.
'*
'* Arguments:
'*   latitude, longitude, year, month, day, hour, minute, second,
'*   timezone, daylightsavingstime
'* Return value:
'*   solar azimuth in degrees from north
'*   solar elevation in degrees from horizon
'*   earth radius vector (distance to the sun in AU)
'*
'* Note: solarelevation and solarazimuth functions are identical
'*       and could converted to a VBA subroutine that would return
'*       both values.
'*
'*************************************************************************/

Dim longitude As Double, Latitude As Double
Dim Zone As Double, daySavings As Double
Dim hh As Double, mm As Double, ss As Double, timenow As Double
Dim jd As Double, t As Double, r As Double
Dim alpha As Double, theta As Double, Etime As Double, eqtime As Double
'Dim SolarDec As Double, earthRadVec As Double, solarTimeFix As Double
Dim SolarDec As Double, solarTimeFix As Double
Dim trueSolarTime As Double, hourangle As Double, harad As Double
Dim csz As Double, zenith As Double, azDenom As Double, azRad As Double
Dim azimuth As Double, exoatmElevation As Double
Dim step1 As Double, step2 As Double, step3 As Double
Dim refractionCorrection As Double, Te As Double, solarzen As Double

            longitude = lon * -1
            Latitude = lat
            If (Latitude > 89.8) Then Latitude = 89.8
            If (Latitude < -89.8) Then Latitude = -89.8

            Zone = timezone * -1
            daySavings = dlstime * 60
            hh = hours - (daySavings / 60)
            mm = minutes
            ss = seconds

'//    timenow is GMT time for calculation in hours since 0Z
            timenow = hh + mm / 60 + ss / 3600 + Zone

            jd = calcJD(year, month, day)
            t = calcTimeJulianCent(jd + timenow / 24#)
            r = calcSunRadVector(t)
            alpha = calcSunRtAscension(t)
            theta = calcSunDeclination(t)
            Etime = calcEquationOfTime(t)

            eqtime = Etime
            SolarDec = theta '//    in degrees
            earthRadVec = r
```

```
solarTimeFix = eqtime - 4# * longitude + 60# * Zone
trueSolarTime = hh * 60# + mm + ss / 60# + solarTimeFix
'//    in minutes

Do While (trueSolarTime > 1440)
    trueSolarTime = trueSolarTime - 1440
Loop

hourangle = trueSolarTime / 4# - 180#
'//    Thanks to Louis Schwarzmayr for the next line:
If (hourangle < -180) Then hourangle = hourangle + 360#

harad = degToRad(hourangle)

csz = Sin(degToRad(Latitude)) * _
      Sin(degToRad(SolarDec)) + _
      Cos(degToRad(Latitude)) * _
      Cos(degToRad(SolarDec)) * Cos(harad)

If (csz > 1#) Then
    csz = 1#
ElseIf (csz < -1#) Then
    csz = -1#
End If

zenith = radToDeg(Application.WorksheetFunction.Acos(csz))

azDenom = (Cos(degToRad(Latitude)) * Sin(degToRad(zenith)))

If (Abs(azDenom) > 0.001) Then
    azRad = ((Sin(degToRad(Latitude)) * _
        Cos(degToRad(zenith))) - _
        Sin(degToRad(SolarDec))) / azDenom
    If (Abs(azRad) > 1#) Then
        If (azRad < 0) Then
            azRad = -1#
        Else
            azRad = 1#
        End If
    End If

    azimuth = 180# - radToDeg(Application.WorksheetFunction.Acos(azRad))

    If (hourangle > 0#) Then
        azimuth = -azimuth
    End If
Else
    If (Latitude > 0#) Then
        azimuth = 180#
    Else
        azimuth = 0#
    End If
End If
If (azimuth < 0#) Then
    azimuth = azimuth + 360#
End If

exoatmElevation = 90# - zenith

If (exoatmElevation > 85#) Then
    refractionCorrection = 0#
Else
    Te = Tan(degToRad(exoatmElevation))
```

```
            If (exoatmElevation > 5#) Then
                refractionCorrection = 58.1 / Te - 0.07 / (Te * Te * Te) + _
                    0.000086 / (Te * Te * Te * Te * Te)
            ElseIf (exoatmElevation > -0.575) Then
                step1 = (-12.79 + exoatmElevation * 0.711)
                step2 = (103.4 + exoatmElevation * (step1))
                step3 = (-518.2 + exoatmElevation * (step2))
                refractionCorrection = 1735# + exoatmElevation * (step3)
            Else
                refractionCorrection = -20.774 / Te
            End If
            refractionCorrection = refractionCorrection / 3600#
        End If

        solarzen = zenith - refractionCorrection
        solarazimuth = azimuth
        solarelevation = 90# - solarzen

End Sub
```

APPENDIX C: SEDIMENT-WATER HEAT EXCHANGE

Although the omission of sediment-water heat exchange is usually justified for deeper systems, it can have a significant impact on the heat balance for shallower streams. Consequently, sediment-water heat exchange is included in QUAL2K.

A major impediment to its inclusion is that incorporating sediment heat transfer often carries a heavy computational burden. This is because the sediments are usually represented as a vertically segmented distributed system. Thus, inclusion of the mechanism results in the addition of numerous sediment segments for each overlying water element.

In the present appendix, I derive a computationally-efficient lumped approach that yields comparable results to the distributed methods.

The conduction equation is typically used to simulate the vertical temperature distribution in a distributed sediment (Figure 29*a*)

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2} \tag{310}$$

This model can be subjected to the following boundary conditions:

$$T(0,t) = \overline{T} + T_a \cos\left[\omega(t-\phi)\right]$$

$$T(\infty,t) = \overline{T}$$

where T = sediment temperature [ºC], t = time [s], α = sediment thermal diffusivity [m^2 s^{-1}], and z = depth into the sediments [m], where $z = 0$ at the sediment-water interface and z increases downward, $\overline{T}$ = mean temperature of overlying water [ºC], T_a = amplitude of temperature of overlying water [ºC], ω = frequency [s^{-1}] = $2\pi/T_p$, T_p = period [s], and ϕ = phase lag [s]. The first boundary condition specifies a sinusoidal Dirichlet boundary condition at the sediment-water interface. The second specifies a constant temperature at infinite depth. Note that the mean of the surface sinusoid and the lower fixed temperature are identical.

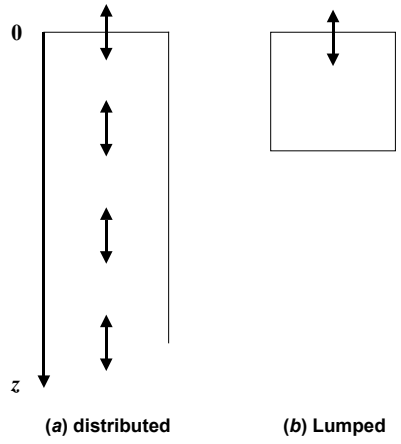

Figure 29. Alternate representations of sediments: (a) distributed and (b) lumped.

Applying these boundary conditions, Eq. (254) can be solved for (Carslaw and Jaeger 1959)

$$T(z,t) = \overline{T} + T_a e^{-\omega' z} \cos\left[\omega(t-\phi) - \omega' z\right] \tag{311}$$

where ω' [m^{-1}] is defined as

$$\omega' = \sqrt{\frac{\omega}{2\alpha}} \tag{312}$$

The heat flux at the sediment water interface can then be determined by substituting the derivative of Eq. (255) into Fourier's law and evaluating the result at the sediment-water interface ($z = 0$) to yield

$$J(0,t) = \rho C_p \sqrt{\omega\alpha}\ T_a \cos\left[\omega(t-\phi) + \pi/4\right] \tag{313}$$

where $J(0, t)$ = flux [W/m^2].

An alternative approach can be developed using a first-order lumped model (Figure 29*b*),

$$H_s \rho_s C_{ps} \frac{dT_s}{dt} = \frac{\alpha_s \rho_s C_{ps}}{H_s/2}\left[\overline{T} + T_a \cos\left[\omega(t-\phi)\right] - T_s\right]$$

where H_{sed} = the thickness of the sediment layer [m], ρ_s = sediment density [kg/m^3], and C_{ps} = sediment specific heat [joule (kg ^{o}C)]$^{-1}$]. Collecting terms gives,

$$\frac{dT}{dt} + k_h T = k_h \overline{T} + k_h T_a \cos\left[\omega(t-\phi)\right]$$

where $k_h = 2\alpha_s/H_{sed}^2$. After initial transient have died out, this solution to this equation is

$$T = \overline{T} + \frac{k_h}{\sqrt{k_h^2 + \omega^2}} T_a \cos\left[\omega(t-\phi) - \tan^{-1}(\omega/k_h)\right] \tag{314}$$

which can be used to determine the flux as

$$J = \frac{2\alpha}{H_{sed}} \rho C_p T_a \left[\cos\left[\omega(t-\phi)\right] - \frac{k_h}{\sqrt{k_h^2 + \omega^2}} \cos\left[\omega(t-\phi) - \tan^{-1}(\omega/k_h)\right]\right] \tag{315}$$

It can be shown that Eqs. (231) and (234) yield identical results if the depth of the single layer is set at

$$H_{sed} = \frac{1}{\omega'} \tag{316}$$

Water quality models typically consider annual, weekly and diel variations. Using $\alpha = 0.0035$ cm^2/s (Hutchinson 1957), the single-layer depth that would capture these frequencies can be calculated as 2.2 m, 30 cm and 12 cm, respectively.

Because QUAL2K resolves diel variations, a value on the order of 12 cm should be selected for the sediment thickness. We have chosen of value of 10 cm as being an adequate first estimate because of the uncertainties of the river sediment thermal properties (Table 4).