

# Water Temperature Model

EcoSwarm software developed by:  
Steve Railsback, Lang Railsback & Assoc. (LRA@Northcoast.com)  
Steve Jackson, Jackson Scientific Computing (jackson3@humboldt1.com)

## Overview and Conventions

WaterTemperatureModel is a class calculating heat fluxes and temperature changes for waterbodies. It is designed especially for shallow, still waterbodies such as ponds or river backwaters. Except as noted, equations and parameters are from: *Theurer, F. D., K. D. Voos, and W. J. Miller. 1984. Instream water temperature model. Instream Flow Information Paper 16 FWS/OBS-84/15, U. S. Fish and Wildlife Service, Washington, D.C.* These equations and parameters were used by Theurer et al. for a daily time step model (with daily mean weather data as input), but WaterTemperatureModel allows the user to choose the time step.

WaterTemperatureModel assumes that the waterbody it is applied to is well-mixed over each time step.

The class calculates the flux of heat into the water body due to a number of processes. Fluxes are in units of joules per square meter of surface area per second ( $J/m^2/s$ ). Fluxes are positive for transfer of heat into the waterbody, so processes such as evaporative cooling that cause heat to leave the waterbody produce negative values.

All temperature parameters are in °C. Humidity is relative humidity, fraction of saturation (between 0.0 and 1.0). Wind speed is in meters per second (m/s).

One of the major heat fluxes is insolation (solar radiation), which is not calculated by WaterTemperatureModel. Instead, the EcoSwarm class SolarManager can be used to calculate insolation and provide it to WaterTemperatureModel.

At least one heat flux often included in water temperature models is not included in WaterTemperatureModel: convection of heat between the water and the earth and groundwater.

A typical application of WaterTemperatureModel uses only the method `getNewTemperatureWithSurfaceArea:`, but other “get” methods can provide results for each individual heat flux.

## Creating

```
+ create: aZone
```

Creates and returns a waterTemperatureModel object.

## Setting

(None)

## Using

```
- (double) getNewTemperatureWithSurfaceArea:  
    (double) aSurfaceArea  
    withVolume: (double) aVolume  
    withTimeInterval: (double) anInterval  
    withInsolation: (double) aSolarFlux  
    withAirT: (double) anAirTemperature  
    withWaterT: (double) aWaterTemperature  
    withHumidity: (double) aHumidity  
    withWindSpeed: (double) aWindSpeed
```

Calculates a new water temperature (newWaterT) for a water body, determining the net heat flux through the specified surface area, the total change in heat over the specified time interval, and the corresponding new temperature. Parameter aSurfaceArea is the water body's surface area over which heat exchange occurs, in square meters; aVolume is the water body's volume in cubic meters; and anInterval is the length of time simulated, in seconds.

The method first calculates the net heat flux (netHeatFlux), using method **getNetFluxWithInsolation:**. The total heat exchanged (heatChange, in Joules) is then calculated as:

$$\text{heatChange} = \text{netHeatFlux} \times \text{aSurfaceArea} \times \text{anInterval}.$$

Then the waterbody's new total heat content is calculated as the heat content at the initial temperature plus heatChange:

$$\text{totalHeat} = (\text{aVolume} \times \text{aWaterTemperature} \times 4,182,000) + \text{heatChange}$$

where 4,182,000 is the number of Joules per cubic meter of water per °C.

Then the new temperature is calculated:

$$\text{newWaterT} = \text{totalHeat} / (\text{aVolume} \times 4,182,000).$$

Finally, if the value of newWaterT is calculated to be less than zero, it is set to zero. This model does not represent the heat of fusion and ice formation, so is particularly unreliable as water temperature approaches zero.

An error condition is raised if aVolume is not greater than zero.

- **(double) getNetFluxWithInsolation:** (double) aSolarFlux  
withAirT: (double) anAirTemperature  
withWaterT: (double) aWaterTemperature  
withHumidity: (double) aHumidity  
withWindSpeed: (double) aWindSpeed

Calculates the total heat flux for five processes, in J/m<sup>2</sup>/s. The parameter aSolarFlux provides the flux due to insolation, which can be calculated by a separate EcoSwarm class, SolarManager. Fluxes for atmospheric radiation, water back-radiation, evaporative cooling, and convection are calculated by messaging the following methods. The net flux is calculated by simply summing the five fluxes.

- **(double) getAtmosphericRadFluxWithAirT:**  
(double) anAirTemperature

Calculates heat flux due to atmospheric longwave radiation. The following simple formulation, which neglects shading and cloud cover, is used.

$$\text{atmosRadFlux} = (1 - r) e \sigma (\text{anAirTemperature} + 273)^4$$

where  $r = 0.03$ ,

$$e = 9.062\text{E-}6 (\text{anAirTemperature} + 273)^2, \text{ and}$$

$$\sigma = 5.67\text{E-}8.$$

- **(double) getWaterBackRadiationFluxWithWaterT:**  
(double) aWaterTemperature

Calculates the (negative) heat flux due to radiation from water back to the atmosphere.

$$\text{waterBackRadFlux} = -0.9526 \sigma (\text{aWaterTemperature} + 273)^4, \text{ where}$$

$$\sigma = 5.67\text{E-}8.$$

- **(double) getEvaporativeFluxWithAirT:** (double) anAirTemperature  
withWaterT: (double) aWaterTemperature  
withHumidity: (double) aHumidity  
withWindSpeed: (double) aWindSpeed

Calculates the (negative) heat flux due to evaporative cooling. A relatively simple model for ponds is corrected from: *Culberson, S. D., and R. H. Piedrahita. 1996. Aquaculture pond ecosystem model: temperature and dissolved oxygen prediction - mechanism and application. Ecological Modelling 89:231-258.*

$$\text{evapFlux} = -1.405 \text{ aWindSpeed} [\text{saturatedVaporPress} - \text{vaporPress}], \text{ where}$$

saturatedVaporPress = 25.37 exp[17.62 – 5271/(aWaterTemperature + 273)], and

vaporPress = aHumidity 25.37 exp[17.62 – 5271/(aWaterTemperature + 273)].

Daily mean water temperature is often calibrated by adjusting wind speed.

- **(double) getConvectionFluxWithAirT:** (double) anAirTemperature  
**withWaterT:** (double) aWaterTemperature  
**withWindSpeed:** (double) aWindSpeed

Calculates the heat flux due to convection from air to water. The method simplifies the lake-type model of Theurer et al. by assuming atmospheric pressure is constant at 760 millibars.

convectFlux = –0.00255 aWindSpeed atmPress  
(aWaterTemperature – anAirTemperature)

where atmPress = 760.

## Document History

First draft: SFRailsback, 4/30/2003.

Check for negative values of newWaterT added: SFR 5/9/2003.

Code review: SFRailsback 6/18/2003.

Documentation of division by zero check in **getNewTemperatureWithSurfaceArea:**  
SFRailsback 10/21/2003.

Document revision: SFRailsback 7/16/2004.