

Ecography

ECOG-02360

Kearney, M. R. and Porter, W. P. 2016. NicheMapR – an R package for biophysical modelling: the microclimate model. – Ecography doi: 10.1111/ecog.02360

Supplementary material

Appendix 1 Underlying theory and equations of the NicheMapR microclimate model

Michael R. Kearney

2016-07-23

Contents

1. Introduction	2
2. Solar radiation calculations	2
2.1 Extra-terrestrial Radiation	2
2.2 Terrestrial Radiation	4
2.3 Slope, aspect and hill-shade effects	8
3. Longwave radiation	8
4. Hourly interpolation of weather data	10
5. Vertical air temperature and wind speed profiles	11
6. Hydric and thermal properties of air, and surface evaporation	12
7. Soil heat balance and soil thermal properties	12
8. Soil water balance	13
9. Snow	14
10. References	15

1. Introduction

This document is Appendix 1 for the NicheMapR microclimate model software note (Kearney and Porter 2016). It provides a detailed explanation of the underlying theory and equations of the microclimate model used in NicheMapR. All of the calculations described here are coded in Fortran in a set of subroutines and functions depicted in Figure 1. The Fortran code is compiled as a dynamic link library that can be called from R, as described in Kearney and Porter (2016).

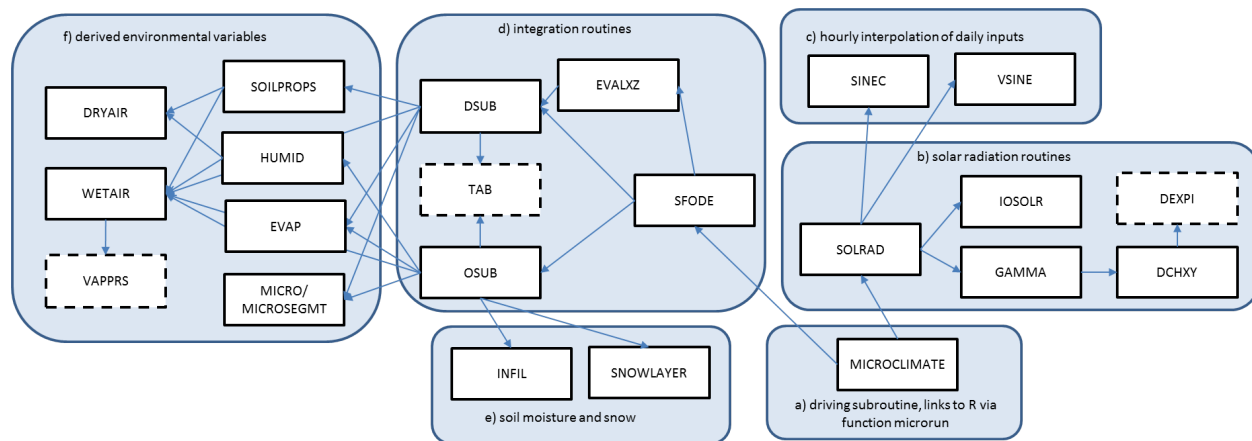


Figure 1: Structure of the NicheMapR microclimate model Fortran program. Solid boxes are subroutines and dashed boxes are functions.

2. Solar radiation calculations

Solar radiation has a dominating effect on the available microclimates across the earth. The total solar irradiance (Wm^{-2}) reaching the earth at a given location depends on

1. the radiation reaching the outer edge of the atmosphere at the location of interest (extra-terrestrial radiation - affected by the earth's orbit, time of year and day, and location on the earth);
2. the effect of the atmosphere as the radiation travels through it to the ground (terrestrial radiation - affected by attenuation due to Rayleigh scattering, aerosols, water vapour, ozone and cloud);
3. the terrain (vegetation shade, hill-shade, slope, aspect).

The SOLRAD subroutine used in the NicheMapR microclimate model for calculating clear sky solar radiation is based on McCullough and Porter (1971) with modifications to incorporate skylight before sunrise and after sunset (Rozenberg 1966, p. 19) and to include the effects of sloping surfaces and hill-shade.

2.1 Extra-terrestrial Radiation

The total solar irradiance reaching an imaginary plane perpendicular to the sun's rays at the outer edge of the atmosphere is obtained by integrating across all wavelengths of radiation emitted by the sun after correcting for the distance of the earth from the sun and for the angle at which the sun is shining on that plane. Thus, the wavelength-specific irradiance reaching this plane is

$$I_{\lambda} = S_{\lambda} \left(\frac{a}{r}\right)^2 \cos Z$$

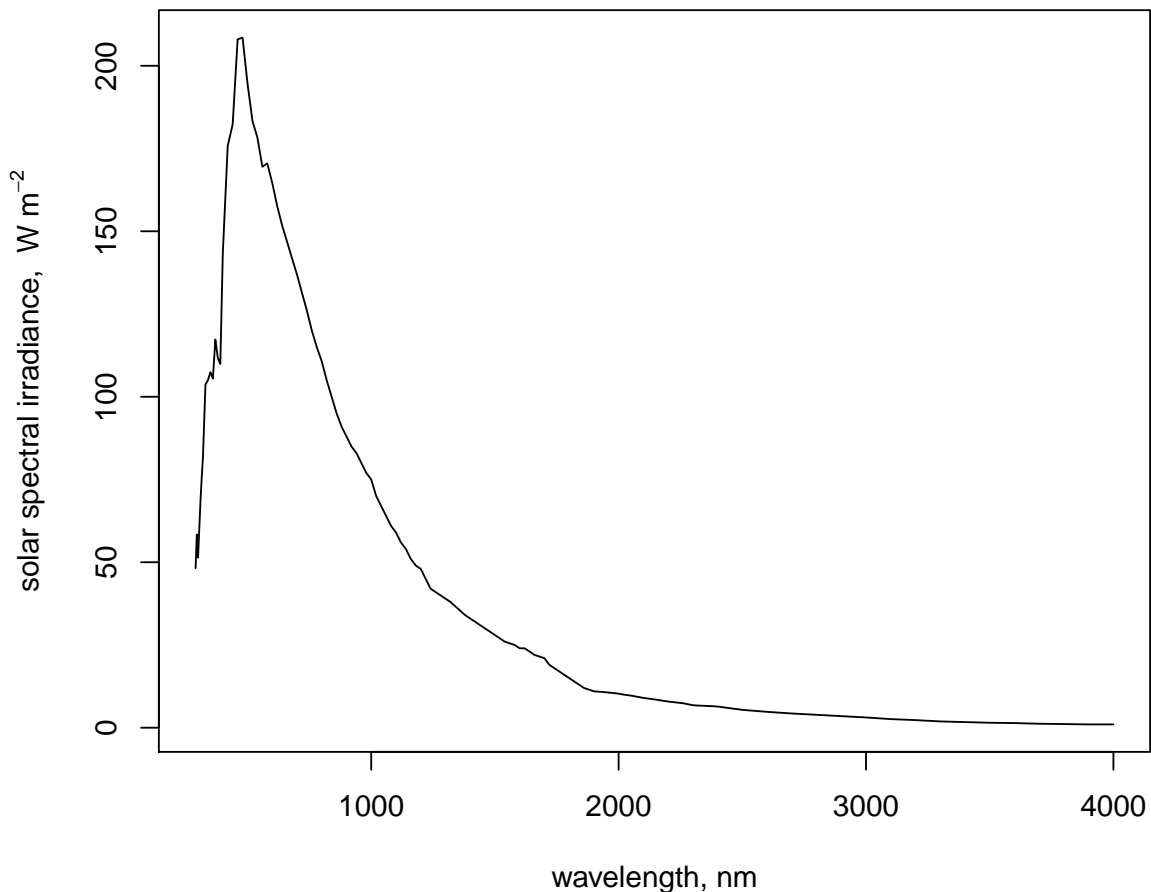


Figure 2: Solar spectral irradiance as a function of wavelength

where S_λ is solar spectral irradiance reaching a plane perpendicular to the incoming radiation at one astronomical unit from the sun (Fig. 2, data stored internally in SOLRAD), a is the length of the long (semi-major) axis of the earth's elliptical orbit around the sun, r is the distance between the earth and the sun and Z is the angle between the sun's rays and a line extending perpendicular to the imaginary plane (so no direct radiation is received by this plane if $Z > 90^\circ$, but see below for twilight conditions).

The factor $(\frac{a}{r})^2$ is approximated in the model as $1 + 2\epsilon \cos(\omega d)$ where $\omega = \frac{2\pi}{365}$, ϵ is the eccentricity of the earth's orbit (default value of 0.01675) and d is the day of the year (1-365). From the astronomical triangle, the term $\cos Z = \cos \phi \cos \delta h + \sin \phi \sin \delta$, where ϕ is the latitude, δ is the solar declination and h is the solar hour angle. The solar declination (the latitude on earth where the sun is directly overhead on a given day) is $\delta = \arcsin(0.39784993 \sin \zeta)$ where the ecliptic longitude of the earth in its orbit is $\zeta = \omega(d - 80) + 2\epsilon[\sin(\omega d) - \sin(\omega 80)]$. The solar hour angle (the angular distance of the sun relative to the zenith crossing; the right ascension for an observer at a particular location and time of day) is $h = 15(t_d - t_{sn})$ degrees, with t_d the local standard time of day and t_{sn} the local standard time of true solar noon. The value of t_{sn} is calculated by adding/subtracting 4 min for each degree of longitude west/east of the reference longitude for the local time zone. The hour angle at sunset $H_+ = \arccos[-\tan \delta \tan \phi]$ while the hour angle at sunrise $H_- = -H_+$. In the polar zones, where the (absolute) latitude $\phi > 66^\circ 30'$, there may be no sunrise/sunset but

rather a long day ($H_+ = \pi$) or a long night ($H_+ = 0$).

2.2 Terrestrial Radiation

Solar radiation reaching the earth, i.e. terrestrial radiation, varies from extra-terrestrial radiation because of scattering, reflection and absorption of radiation by the atmosphere. It may thus include a direct I_λ and a scattered component D_λ (skylight or sky radiation), where global terrestrial radiation $G_\lambda = I_\lambda + D_\lambda$. The total radiant energy from the sun G is obtained by integrating across all emitted wavelengths, thus $G = I + D$. If the zenith angle Z is between 90° and 88° , twilight conditions are assumed and $G = D = 10^{(41.34615384 - 0.423076923Z)0.00146}$ based on the regression in (Rozenberg 1966, p. 19). Otherwise, G is computed by calculating monochromatic irradiance for the direct and diffuse components and then integrating across all wavelengths using the trapezoidal rule.

2.2.1 Direct Component of Terrestrial Radiation

The direct, horizontal plane component of terrestrial solar radiation is calculated as

$$I_\lambda = S_\lambda \left(\frac{a}{r}\right)^2 \cos Z_a \exp[-m(Z_a)_\lambda \tau(h)]$$

where the apparent zenith angle, Z_a , is the angle between the beams of direct radiation and the local zenith direction (i.e. vertical) at the surface of the earth, h is the elevation above sea level at which I_λ is to be evaluated, $m(Z_a)$ is the relative optical air mass, and $_\lambda \tau(h)$ is the total vertical monochromatic extinction coefficient at elevation h . The term $m(Z_a)_\lambda \tau(h)$ represents the “total monochromatic extinction coefficient” and is often abbreviated as $_\lambda \tau(Z_a)$. The term $m(Z_a) = \sec Z_a$ for $Z_a < 80^\circ$. Atmospheric refraction means that, for large values of the true zenith angle Z ($\geq 88^\circ$), the apparent zenith angle Z_a will differ by a factor $R = 16 + (Z_a - 88)10$ minutes of an arc. For $80^\circ \leq Z_a \leq 90^\circ$ the term $m(Z_a)$ can be computed as $m(Z_a) = [\cos Z_a + 0.025 \exp(-11 \cos Z_a)]^{-1}$. Variation in $m(Z_a)$ with altitude is ignored as it is negligible up to at least 6 km above sea level.

The total vertical monochromatic extinction depends on Rayleigh scattering (due to molecules, especially oxygen and nitrogen) $_\lambda \tau_R$, scattering due to aerosols (dust, water droplets, etc.) $_\lambda \tau_A$, and absorption due to ozone $_\lambda \tau_O$ and water vapour $_\lambda \tau_W$, thus

$$_\lambda \tau(h) = _\lambda \tau_R(h) + _\lambda \tau_A(h) + _\lambda \tau_O(h) + _\lambda \tau_W(h)$$

The separate extinction factors are first obtained for sea level and then adjusted for elevation h . Wavelength-specific sea level Rayleigh and ozone optical thicknesses are taken from Elterman (1968, 1970) for atmospheric pressure $P = 1013$ mb, a sea level meteorological range MR_0 (visibility at 0.55μ) of 25 km and a total atmospheric ozone content X of 0.34 cm NTP (normal temperature and pressure; $T = 18^\circ \text{C}$ and $P = 1013$ mb) (Fig. 3a,b). Rayleigh optical thickness is subsequently adjusted by the ratio of the barometric pressure to the reference pressure. Wavelength-specific sea level optical thickness for 1 cm precipitable water vapour in the air column is taken from Gates and Harrop (1963) (Fig. 3c).

Total atmospheric ozone shows significant seasonal and latitudinal variations, and is assumed to follow the variation in Figure 4 (data from Robinson 1966, p. 114). A correction factor of the ratio of the latitude/season-specific ozone to the reference level (0.34 cm) is thus applied to the elevation-adjusted extinction term for ozone (Fig. 4).

Aerosol attenuation profiles vary significantly with latitude and longitude. The original version of the microclimate model applied data from Elterman (1968, 1970) on wavelength-specific attenuation (Fig. 3d), but this is based on observations in North America which can be strongly divergent from other regions of the globe such as Australia. The microclimate model in NicheMapR now includes the option of using the [Global Aerosol Data Set \(GADS\)](#) (Koepke et al. 1997), which is a Fortran program that can produce profiles of aerosol attenuation on a 5° by 5° grid of the globe. GADS has been modified to give output for the full

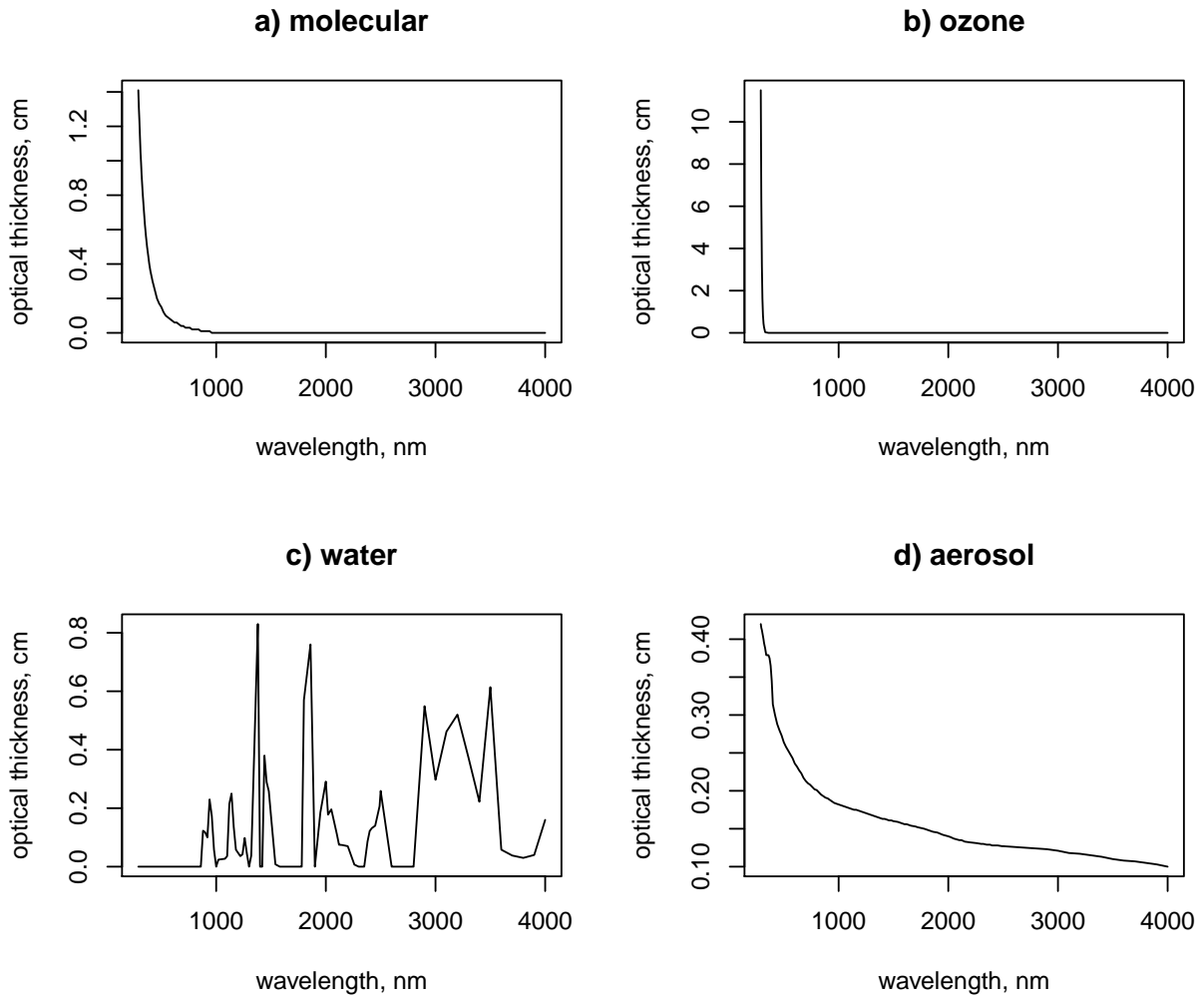


Figure 3: Wavelength-specific optical thickness for different atmospheric components

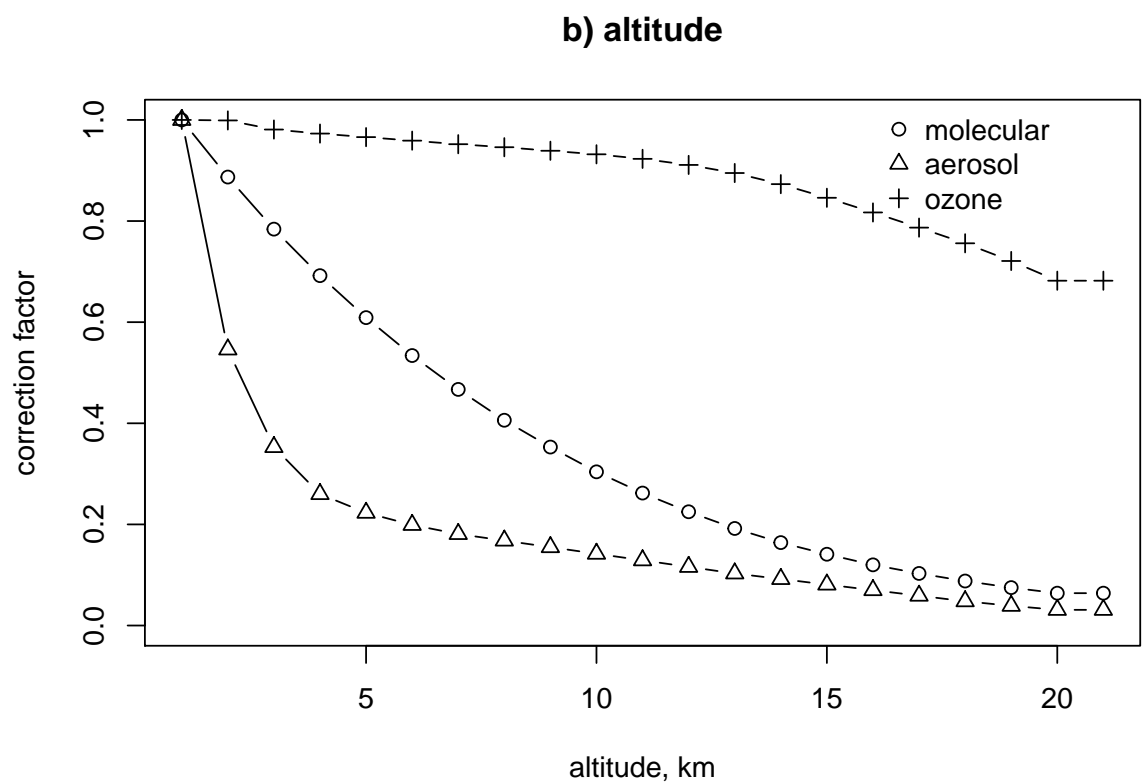
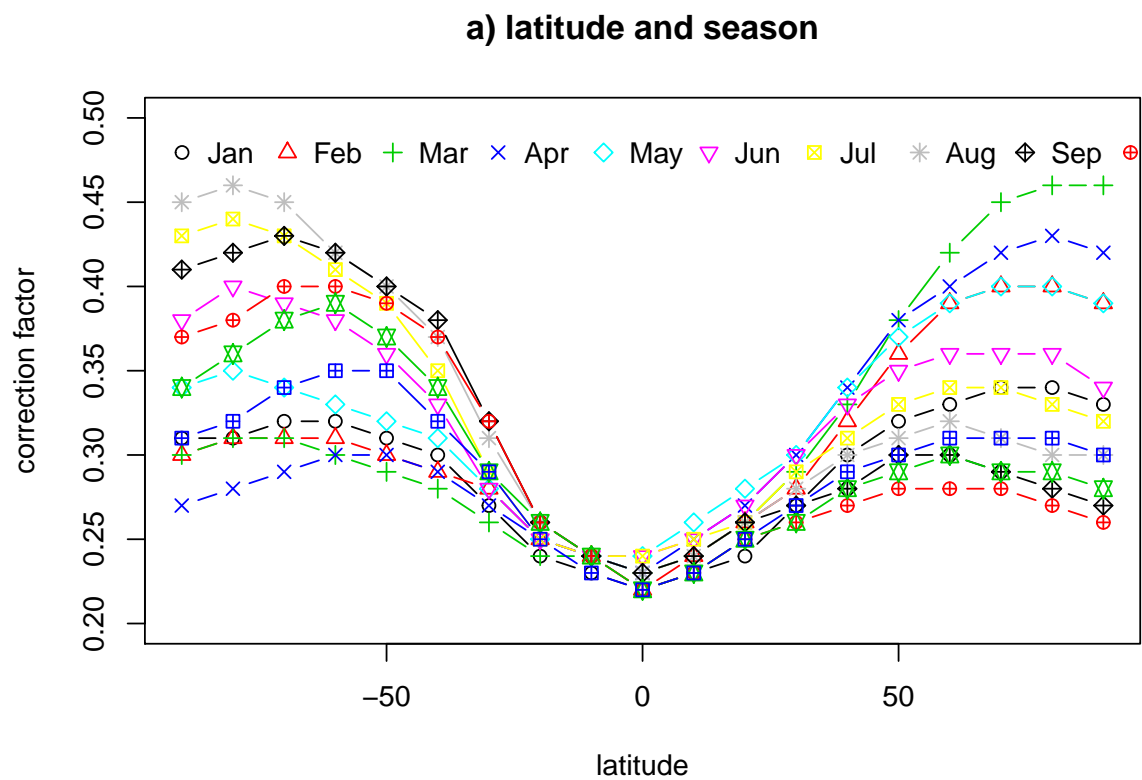


Figure 4: Seasonal, latitudinal and altitudinal correction factors for atmospheric ozone

spectral profile for a single location (rather than a single wavelength for the all co-ordinates in the database), compiled to be called within the R environment, and included as part of the NicheMapR package. The data for GADS is installed in the folder ‘extdata’ and the Fortran program is called with the function ‘get.gads’ which returns 25 values for optical depths between 250 and 4000 nm for a user specified season (summer or winter) and relative humidity level (8 values ranging from 0-100%). These values can then be splined across the 111 wavelengths required by the microclimate model (see example in function `micro_global`).

Finally, the extinction factors must be adjusted for elevation h . For $\lambda\tau_R$, $\lambda\tau_A$ and $\lambda\tau_O$ these adjustments A_R , A_A and A_O , respectively, are made via the following polynomials fitted to the profiles provided by (Elterman 1968):

$$\begin{aligned} A_R(h) &= 7.277E - 05h^3 + 0.00507293h^2 - 0.12482149h + 1.11687469 \\ A_A(h) &= 8.35656E - 07h^6 - 6.26384E - 05h^5 + 1.86967E - 03h^4 - 2.82585E - 02h^3 + 2.26739E - 01h^2 \\ &\quad - 9.25268E - 01h + 1.71321 \\ A_O(h) &= 1.07573E - 06h^5 - 5.14511E - 05h^4 + 7.97960E - 04h^3 - 4.90904E - 03h^2 + 2.99258E - 03h + 1.00238 \end{aligned}$$

The data on which these curves are fitted (Fig. 5) are stored internally in SOLRAD.

In the model, no generic altitudinal correction is made for the effect of precipitable water vapour as its vertical distribution is highly variable, and $A_W(h)$ is assumed by default to equal 1 cm for all elevations but can be changed by the user. Using the Gates and Harrop (1963) coefficients, the total monochromatic extinction coefficient for water vapour absorption is

$$\lambda\tau_W(Z_{a,h}) = \lambda\tau_W \sqrt{m(Z_a)wA_W(h)}$$

In summary, the total monochromatic extinction coefficient $\lambda\tau(Z_a) = m(Z_a)\lambda\tau(h) =$

$$m(Z_a) \left[\left(\frac{P}{1013} \right)_{\lambda\tau_R} A_R(h) + \left(\frac{25}{MR_0} \right)_{\lambda} \tau_A(h) A_A(h) + \left(\frac{X}{0.34} \right)_{\lambda} \tau_O(h) A_O(h) \right] + \lambda\tau_W \sqrt{m(Z_a)wA_W(h)}$$

This value is constrained to be no greater than 80 in cases where low sun angles may make $m(Z_a)$ too large.

2.2.2 Scattered Component of Terrestrial Radiation

Scattered radiation D_λ is the solar radiation that is diffusely transmitted by the earth’s atmosphere and depends on the wavelength of the radiation, the solar zenith angle, the optical properties of the terrestrial atmosphere and the reflecting properties of the surface underlying the atmosphere. For wavelengths greater than 360 nanometres,

$$D_\lambda(\lambda\tau_R, Z, A) = I_{0,\lambda} \left[\frac{\gamma_l(Z, \lambda\tau_R) + \gamma_r(Z, \lambda\tau_R)}{2(1 - A(\lambda)\bar{s}(\lambda\tau_R))} - e^{-\lambda\tau_R m(Z_a)} \right]$$

where $A(\lambda)$ is the albedo of the underlying surface and the functions γ_l , γ_r and \bar{s} are calculated using the Dave-Warten subroutine (Dave and Warten 1968). In the ultraviolet region, ≤ 360 nanometers,

$$D_\lambda = \frac{S_\lambda}{\pi} \left(\frac{a}{r} \right)^2 \left[F_d(p, Z, \lambda) + \frac{F'_d(p, Z, \lambda)}{Q} \frac{A(\lambda)}{(1 - A(\lambda)\bar{s}(\lambda\tau_R + \lambda\tau_O))} \right]$$

where the functions F_d and $\frac{F'_d}{Q}$ are generated from tables in Dave and Furakawa (1967) and are stored in SOLRAD.

2.3 Slope, aspect and hill-shade effects

For sloping surfaces, a correction is made to the zenith angle, Z based on the angle of the slope SL , the azimuth of the slope AZ_{SL} , and the azimuth of the sun AZ_{SUN} such that the zenith angle (in radians) for the slope

$$Z_{SL} = \arccos(\cos(Z) \cos(SL) + \sin(Z) \sin(SL) \cos(AZ_{SUN} - AZ_{SL})).$$

In the Southern Hemisphere, the azimuth of the sun

$$AZ_{SUN,S} = \arccos\left(\frac{(\sin(\phi) \sin\left(\frac{90\pi}{180} - Z\right) - \sin(\delta))}{\cos(\phi) \cos\left(\frac{90\pi}{180} - Z\right)}\right)$$

where ϕ is the latitude and δ is the solar declination. The equivalent formula for the Northern Hemisphere is

$$AZ_{SUN,N} = \arccos\left(\frac{\sin(\delta) - \sin(\phi) \sin\left(\frac{90\pi}{180} - Z\right)}{\cos(\phi) \cos\left(\frac{90\pi}{180} - Z\right)}\right)$$

This value ranges from 0-180° and is thus subtracted from 360° to obtain afternoon angles.

Local terrain may also obscure direct and diffuse radiation by increasing the horizon angle and thus causing ‘hill-shade’ (Dozier et al. 1981). Horizon angles in 24 directions can be entered into NicheMapR and, when the altitude of the sun is less than the horizon angle in that direction, direct radiation is set to zero. The [r.horizon function](#) of the GIS package GRASS (Neteler et al. 2012) can be used to compute horizon angles from a DEM.

2.2.3 Total solar radiation reaching a surface

The total or ‘global’ solar radiation finally reaching the earth’s surface can now be summarised as

$$Q_{solar} = G = \int_0^{\text{inf}} G_{\lambda} d\lambda = \int_0^{\text{inf}} (I_{\lambda} + D_{\lambda}) d\lambda = I + D$$

and depends on the solar zenith angle, the elevation (atmospheric pressure), the albedo of the underlying surface A and the optical properties of the atmosphere. In the NicheMapR microclimate model, it is summed over 111 wavelengths from 0.290 to 4 μm . Adjustments for cloud cover are made according to the Angstrom formula (formula 5.33 on P. 177 of Linacre 1992) $Q_{solar,cloud} = Q_{solar}(0.36 + 0.64(\frac{1-P_{cloud}}{100}))$ where P_{cloud} is the percentage cloud cover.

3. Longwave radiation

Longwave radiation is a major route of heat exchange in terrestrial environments. All objects in a habitat emit longwave radiation at a rate proportional to the 4th power of their temperature, $Q_{IR} = \sigma \epsilon T^4$ where σ is the Stefan-Boltzmann constant ($Wm^{-2}K^{-4}$), ϵ is the emissivity and T is the object’s temperature in Kelvin. The heat budget at the ground surface depends in part on the longwave radiation received from the sky and surrounding objects (e.g. hills and shade from vegetation) as well as the longwave radiation it emits. The following equations are used for computing infra-red radiation exchange in the NicheMapR microclimate model.

For longwave radiation from clear skies, A_{RAD} , the sky emissivity is approximated as $\epsilon_{SKY} = 1.72 \left(\frac{\epsilon_A}{T_A}\right)^{\frac{1}{7}}$ (Campbell Gaylson S. and Norman 1998, eqn 10.10), where T_A is the shaded air temperature (at reference

height, i.e. 1.2 m) in Kelvin and e_A is the vapour pressure of the air in kilopascals (see section 5). Thus, given a percentage cloud cover CLD ,

$$A_{RAD} = \sigma \epsilon_{SKY} (T_A + 273)^4 \left(1 - \frac{CLD}{100}\right).$$

Cloud longwave radiation C_{RAD} is approximated from $T_A - 2$ assuming an emissivity ϵ of 1, thus

$$C_{RAD} = \sigma \epsilon_{CLD} (T_A - 2 + 273)^4 \frac{CLD}{100}$$

and the total radiation from the sky, given a percentage shade from vegetation or other objects SHD , is

$$Q_{IR,SKY} = (A_{RAD} + C_{RAD}) \left(1 - \frac{SHD}{100}\right).$$

Similarly, assuming objects casting shade are radiating at shaded air temperature, total infra-red radiation from shading

$$Q_{IR,VEG} = C_{RAD} \frac{SHD}{100}.$$

Given a set of n horizon angles in degrees $HORI$, the view factor from ground to sky not obscured by hills (or other objects like buildings or trees)

$$VIEWF = 1 - \sum_{i=1}^n \frac{\sin\left(\frac{\pi}{180} HORI(i)\right)}{n}.$$

Again, assuming that hills radiate at shaded air temperature, total radiation from hill-shade

$$Q_{IR,HILL} = \sigma \epsilon_{HILL} (T_A + 273)^4 (1 - VIEWF).$$

Finally, given a computed ground surface temperature T_{surf} , and assuming shaded ground radiates at shaded air temperature, radiation emitted from the ground

$$Q_{IR,GND} = \sigma \epsilon_{GND} (T_{surf} + 273)^4 \left(1 - \frac{SHD}{100}\right) + \sigma \epsilon_{GND} (T_A + 273)^4 \frac{SHD}{100}.$$

The net longwave radiation gain for the substrate heat budget is therefore

$$Q_{IR} = (Q_{IR,SKY} + Q_{IR,VEG})VIEWF + Q_{IR,HILL}(1 - VIEWF) - Q_{IR,GND}$$

and the effective ‘sky temperature’, an output of the microclimate model,

$$T_{SKY} = \left[\frac{(Q_{IR,SKY} + Q_{IR,VEG})VIEWF + Q_{IR,HILL}(1 - VIEWF)}{\sigma} \right]^{\frac{1}{4}}$$

(assuming ϵ of 1).

4. Hourly interpolation of weather data

If hourly weather station input data for air temperature, relative humidity, wind speed and cloud cover are not available (as is often the case), hourly profiles must instead be generated from maximum and minimum values for the day. The subroutines VSINE and SINEC do this based on the user inputs of when the daily minima and maxima occur relative to sunrise and solar noon and the timings of sunrise, sunset and solar noon as computed by SOLRAD.

VSINE works with relative humidity, wind speed and cloud cover. It first assumes the midnight value is the average of the given minimum and maximum, and computes three slopes: a slope from the midnight point to the relative-to-sunrise maximum (cloud cover, relative humidity) or minimum (wind speed) value, a slope from the sunrise value to the relative-to-midday minimum (cloud cover, relative humidity) or maximum (wind speed) value, and slope from the relative-to-midday value back to the assumed midnight value. The model then uses these slopes to compute the values on the hour through the entire day. Note, however, that if the model is running for every day of the year (and is not doing the first day of the simulation) it uses the value for the last hour of the previous day as the initial value for midnight on the current day. Otherwise, it uses the average of the minimum and maximum value as the starting point. The result is a day length-specific linear interpolation.

SINEC uses a slightly more sophisticated approach to create hourly profiles of air temperature to capture the initial rapid decline in air temperature from its maximum (typically) prior to sunset, and the more gradual decline that occurs through the nighttime. The temperature at the user-specified time relative to sunrise T_{sr} is assumed to be the minimum temperature, while the temperature at sunset $T_{ss} = AZ_s + T_{min} + A$, where A is the average of the minimum and maximum air temperature and

$$Z_s = \sin \left(\frac{360(t_{ss} - t_{ref})}{\frac{2(t_{Tmax} - t_{sr})}{57.29577}} \right)$$

where t_{ss} and t_{sr} are the times (in US military time) of sunrise and sunset, respectively, t_{Tmax} is the time of the maximum air temperature and $t_{ref} = (t_{max} - t_{sr})/2 + t_{sr}$. From midnight to sunrise, if all days of the year are being run (and it is not the first day), air temperature is assumed to change linearly from the last temperature of the previous day to the minimum temperature at t_{sr} . Otherwise, air temperature changes with an exponential decay as

$$T_{air} = \frac{(T_{ss} - T_{sr})}{\exp(E)} + T_{SR}$$

where $E = \tau[(2400 - t_{ss}) + t]$ and

$$\tau = \frac{3}{2400 - t_{ss} + t_{sr}}.$$

Similarly, an exponential decay occurs from the temperature at the user-specified time relative to sunset towards the midnight temperature, where the latter is assumed to be either the minimum temperature of the next day if all days or the year are being run, or the temperature at sunrise. Otherwise, if time is between sunrise and sunset, the air temperature is assumed to follow a sine wave $T_{air} = AZ + T_{min} + A$ where

$$Z_s = \sin \left(\frac{360(t - t_{ref})}{\frac{2(t_{Tmax} - t_{sr})}{57.29577}} \right)$$

5. Vertical air temperature and wind speed profiles

Air temperature and wind speed vary with height above the ground as a function of the air temperature and wind speed at the reference height, the surface temperature, and the presence and spacing of objects on the surface (surface roughness). In the NicheMapR microclimate model, the MICRO subroutine computes a single unsegmented wind velocity u and temperature profile based on the user-specified roughness height z_0 and reference height z while MICROSEGMT computes a three segment velocity and temperature profile based on user-specified, experimentally determined values for roughness heights $z_{0,1}$ and $z_{0,2}$ at two segment heights z_1 and z_2 .

The MICRO subroutine first checks for free convection (lapse) conditions. The wind speed at the new local height $u_{loc} = 2.5u^* \ln(\frac{z_{loc}}{z_0} + 1)$, where the friction velocity $u^* = \frac{0.4u}{\ln(\frac{z}{z_0} + 1)}$, with u the wind speed at reference height z and 0.4 being the von Karman constant.

Given the reference air temperature T_{ref} and the current estimate of the substrate surface temperature T_{sub} , the fictitious temperature at the roughness height

$$T_{z_0} = \frac{T_{ref}S_{t_b} + T_{sub}S_{t_s}}{S_{t_b} + S_{t_s}},$$

with the sublayer Stanton number $S_{t_s} = 0.62 / \left(\frac{z_0 u^*}{12}\right)^{0.45}$ and the bulk Stanton number $S_{t_b} = 0.64 / \ln(\frac{z}{z_0} + 1)$. Then the air temperature at a new local height T_{loc} can then be computed as

$$T_{loc} = T_{z_0} + (T_{ref} - T_{z_0}) \ln\left(\frac{z_{loc}}{z_0} + 1\right).$$

If experimental data are specified for the three segment velocity profile, i.e. z_1 , z_2 , $z_{0,1}$ and $z_{0,2}$ are non-zero, then MICROSEGMT is called instead of MICRO. The computations are similar except that now

$$u_1^* = 0.4u_1 / \ln\left(\frac{z_1}{z_{0,1}} + 1\right),$$

$$u_2^* = 0.4u_2 / \ln\left(\frac{z_2}{z_0} + 1\right),$$

with

$$u_1 = \frac{u_2^*}{0.4} \ln\left(\frac{z_1}{z_{0,1}} + 1\right)$$

and

$$u_2 = \frac{u}{0.4} \ln\left(\frac{z_2}{z_{0,2}} + 1\right).$$

Then

$$S_{t_s} = 0.62 / \left(\frac{z_0 u_2^*}{12}\right)^{0.45}$$

and

$$S_{t_b} = 0.64 / \ln\left(\frac{z_2}{z_0} + 1\right).$$

The wind speed at height i is now $u(i) = 2.5u_s^* \ln(\frac{z_i}{z_{0,s}} + 1)$, where if $z_1 \leq z_i$, $u_s^* = u^*$ and $z_{0,s} = z_{0,1}$ whereas if, $z_1 > z_i$ then if $z_2 \leq z_i$, $u_s^* = u_1^*$ and $z_{0,s} = z_{0,2}$ but if $z_2 > z_i$, $u_s^* = u_2^*$ and $z_{0,s} = z_0$. The air temperature profile is as specified above for subroutine MICRO, but with the modified values of the Stanton numbers S_{t_s} and S_{t_b} .

Finally, convective heat transfer at the surface is computed as

$$Q_{conv} = \frac{0.08472}{T_{ave}} (T_{ref} - T_{sub}) u^* \frac{S_{t_b}}{\left(1 + \frac{S_{t_b}}{S_{t_s}}\right)}$$

where T_{ave} is the average of the reference and substrate temperatures, in Kelvin.

6. Hydric and thermal properties of air, and surface evaporation

Subroutines DRYAIR, WETAIR, HUMID and EVAP, together with function VAPPRS, compute the hydric and thermal properties of air, and the implication for evaporation from the surface. DRYAIR, WETAIR and VAPPRS are now available as R functions in the NicheMapR package and are described in detail in the technical manual ‘‘Properties of Air’’ (Tracy et al. 2016, Tracy et al. 1980) which is now available as a vignette in NicheMapR.

DRYAIR computes atmospheric pressure $P = 101325(1 - \frac{0.0065A}{288})^{\frac{1}{0.190284}}$ where A is altitude (m), together with the air density $\rho_{air} = \frac{101325}{287.04(T_{air}+273.15)}$ where T_{air} is the air temperature ($^{\circ}C$), the dynamic viscosity $v_{dyn} = 1.8325 \times 10^{-5} \frac{296.16+C}{\rho_{air}+273.15+120} (\frac{T_{air}+273.15}{296.16})^{1.5}$, the kinematic viscosity $v_{kin} = \frac{v_{dyn}}{\rho_{air}}$, diffusivity of water vapour in air $\alpha = 2.26 \times 10^{-5} (\frac{T_{air}+273.15}{273.15})^{1.81} \frac{1 \times 10^5}{P}$, the thermal conductivity of air $k_{air} = 0.02425 + (7.038 \times 10^{-5} T_{air})$, and the latent heat of vapourisation of water $\delta H_{vap} = 2.5012 \times 10^6 - 2.3787 \times 10^3 T_{air}$.

Function VAPPRS computes the saturation water vapour pressure for a given T_{air} as

$$e^* = 10^{a(\frac{T_o}{T_{air}} - 1) + b \log_{10}(\frac{T_o}{T_{air}}) - c \left(10^{(d(1 - \frac{T_{air}}{T_o}))} - 1\right) + e \left(10^{(f(\frac{T_o}{T_{air}} - 1))} - 1\right) + \log_{10}(P_0)}$$

where T_{air} is in Kelvin, $P_0 = 1013.246$, $T_0 = 373.16$, $a = -7.90298$, $b = 5.02808$, $c = 1.3816 \times 10^{-7}$, $d = 11.344$, $e = 8.1328 \times 10^{-3}$ and $f = -3.49149$ when T_{air} is positive and

$$e^* = 10^{a(\frac{T_o}{T_{air}} - 1) + b \log_{10}(\frac{T_o}{T_{air}}) + c(1 - \frac{T_{air}}{T_o}) + \log_{10}(d)}$$

where $T_0 = 273.16$, $a = -9.09718$, $b = 3.56654$, $c = 0.876793$, and $d = 6.1071$, when T_{air} is zero or negative.

WETAIR uses VAPPRS to compute vapour pressure for the current relative humidity $e = (e^* RH)/100$ where RH is relative humidity (%), the vapour density $v_d = e^* 0.018016 / (0.998 R T_{air})$ where $R = 8.31434$ and T_{air} is in Kelvin, the specific heat $c_p = \frac{1004.84 + r_w 1846.40}{1.0 + r_w}$ where the mixing ratio $r_w = \frac{(0.62197(1.0053e))}{P - 1.0053e}$, the density of air $\rho_{air} = 0.0034838 \frac{P}{0.999 T_v}$ where the virtual temperature $T_v = T_{air}((1.0 + r_w / \frac{18.016}{28.966}) / (1.0 + r_w))$ and T_{air} is in Kelvin, and some additional properties of humid air not used in the microclimate model. Further details can be found in Tracy et al. (2016).

HUMID uses DRYAIR and WETAIR to adjust relative humidity from the reference height to the local height. EVAP uses DRYAIR and WETAIR to compute the vapour density at 100% relative humidity and at the current humidity, and then determines evaporation from the surface according to the equation $m_{evap} = \frac{P_{wet}}{100} h_d (v_{d,surf} - v_{d,air})$, where P_{wet} is the % of the unit surface area that is acting as a free water surface.

The mass transfer coefficient $h_d = \frac{h_c}{c_p \rho_{air}} (\frac{0.71}{0.60})^{0.666}$ and the convective heat transfer coefficient $h_c = \max(|\frac{Q_{conv}}{T_{loc} - T_{ref}}|, 0.5)$ are both computed in DSUB. h_d is computed from the Chilton-Colborn analogy (Bird et al, 2002), which assumes that the temperature and humidity profile shapes are the same shape.

Finally, m_{evap} is converted to energy lost/gained as heat by evaporation/condensation $Q_{evap} = m_{evap} \Delta H_{vap}$ where the latent heat of vaporisation is computed as $\Delta H_{vap} = 10^3(2500.8 - 2.36 T_{surf} + 0.0016 T_{surf}^2 - 0.00006 T_{surf}^3)$ if the surface temperature $T_{surf} > 0^{\circ}C$, and $\Delta H_{vap} = 10^3(2834.1 - 0.29 T_{surf} + 0.004 T_{surf}^2)$ if $T_{surf} \leq 0^{\circ}C$.

7. Soil heat balance and soil thermal properties

The soil temperature change through depth z and time t is described by the one-dimensional partial differential equation

$$\frac{dT}{dt} = \frac{k}{\rho c} \frac{\delta^2 T}{\delta z^2}$$

(Carslaw and Jaeger 1959) and requires one initial condition and two boundary conditions for its solution. The initial condition (soil temperature profile) is either 1) the mean daily reference air temperature for all soil nodes or 2) (when doing daily simulations) the temperature profile at the end of the previous day. In the former case, three replicate days are run with the ending profile of the first day being the initial condition for the second day and the end of the second day being the initial condition for the third day to establish a steady periodic temperature profile for the soil. The deep soil boundary condition is obtained by assuming it is the mean annual air temperature. The remaining boundary condition at the soil surface is obtained by equating the energy conducted to the soil surface to the net heat transfer to the surface by solar radiation, infrared radiation, convection and evaporation, i.e. a unit area energy balance for the soil surface

$$Q_{cond} = -k_{so} \frac{\delta T}{\delta z} \Big|_{z_0} = Q_{solar} + Q_{IR} + Q_{conv} - Q_{evap}$$

This equation is solved by breaking the soil profile into nodes and solving separate ODEs for each using the Adams Predictor-Corrector algorithm with Runge Kutta starting, with the ODE for the surface $\frac{dT_{s,1}}{dt} = \frac{Q_{solar} + Q_{IR} + Q_{conv} - Q_{evap}}{C_{s,1}}$ and the subsequent ODEs $\frac{dT_{s,i}}{dt} = \frac{K_{s,i-1}(T_{s,i-1} - T_{s,i}) + K_{s,i}(T_{s,i+1} - T_{s,i})}{C_{s,i}}$ where for a given node i the heat capacity $C_i = \rho_{s,i} c_{s,i} \frac{z_{i+1} - z_{i-1}}{2}$, and $K_{s,i} = k_{s,i}(z_{i+1} - z_i)$ where $k_{s,i}$ is the thermal conductivity. The computation of the terms Q_{solar} , Q_{IR} , Q_{conv} and Q_{evap} have been described earlier in this document.

In the NicheMapR microclimate model, 10 soil nodes are specified, including the surface. The spacing is under the control of the user but must be closer near the surface where changes are most rapid, to ensure that the numerical integration is successful. The key soil thermal properties in the equations above are the density ρ , the specific heat capacity c_s and the thermal conductivity k_s , all of which may vary with time as a function of soil temperature and moisture, and with depth as a function of geology. The original version of the microclimate model did not permit soil properties to vary with depth or time. In the present version the user can specify up to 10 different soil properties with depth (i.e. unique properties for each node). The user specifies the conductivity, specific heat capacity, density and water-holding capacity of the mineral components of the different soil layers together with the bulk density and a time-vector for each layer of relative soil moisture values. The overall soil conductivity, specific heat and density values are then adjusted for bulk density and soil moisture and soil temperature, using the methods described in Campbell et al. (1994, equations 8, 9) and Campbell and Norman (Campbell Gaylson S. and Norman 1998, equations 8.13, 8.17, 8.20 and data in tables 8.2 and 9.1) in the subroutine SOILPROPS (Fig. 1) which is called by DSUB and utilises the routines in DRYAIR, WETAIR and VAPPRS. If the soil temperature is between -0.45 and 0.4 °C and there is moisture in the soil, the latent heat of fusion ($333.5 Jg^{-1}$) is added to the computation of specific heat.

8. Soil water balance

The NicheMapR microclimate model now has the capacity for the user to provide daily inputs for soil moisture levels, or they can be calculated from first principles as a function of rainfall, soil type and transpiration. To calculate the soil water budget, Campbell's (1985) Program 11.1 (henceforth C11.1) has been integrated. It is used for simulating depth-specific soil moisture, water potential and humidity gradients in the presence of vegetation. C11.1 takes the approach of linearising the differential equation for flow in space, and using a Newton-Raphson procedure to solve the non-linear equations through time. The linearisation through space involves representing the soil as a series of capacitors and resistors (see Campbell G. S. 1985 Fig. 8.5). It uses matric potential as the dependent variable (rather than matric flux potential) and hence is most computationally efficient when soil is relatively dry. The model is well described in Campbell (1985) thus only

details specific to the integration with the NicheMapR microclimate model and driving data are provided here.

C11.1 is included as Fortran subroutine INFIL which is called from the hourly output OSUB (Fig. 1), with the soil temperature computed for a given hour being used to drive the soil moisture calculations. In turn, the calculated soil moisture profile was used in the computation of the next hour’s soil temperatures. The INFIL routine is iterated with a 6 minute step size to increase the accuracy of the solutions (shorter step sizes did not substantially alter the predictions). All daily rainfall is assumed to have occurred in the first hour of the day and the depth of water pooling on the surface is kept track of by the subtracting the amount computed by C11.1 to have infiltrated or evaporated, with a maximum possible pool depth set by the user. If at any stage water was pooled on the surface, the first soil node is set to be saturated, which effects infiltration.

Potential evaporation was computed by simulating evaporation from a completely wet surface using code of the original version of the NicheMapR model. Following Campbell (1985, equation 12.30), this was partitioned among evaporation and transpiration as a function of leaf area index LAI (a user input). The fractional surface wetness for the next hour’s heat budget was set as the ratio of potential to actual evapotranspiration calculated by C11.1.

The microclimate model uses 10 user-specified nodes for computing the heat budget. C11.1 was modified to insert extra nodes half way between the original 10 nodes, such that a total of 19 nodes were used for soil moisture calculations. The user input variables specific to C11.1 include, for each of these 19 nodes, the Campbell (or Clapp and Hornberger) exponent B , the air entry potential P_e (Jkg^{-1}), the hydraulic conductivity K_s ($kgsm^{-3}$), the bulk density ρ_b ($kg m^{-3}$) and the root density (mm^{-3}) in addition to the leaf area index LAI already mentioned. All other model parameters for C11.1 were fixed at those suggested by Campbell (1985).

9. Snow

The capacity to model the growth and decay of a snowpack, and its influence on substrate temperatures, has been incorporated into the NicheMapR microclimate model by allowing an additional 8 nodes of snow to be added to the 10 substrate nodes, with their thermal properties set to that of ice when snow is present. The snow nodes are at 2.5, 5, 10, 20, 50, 100, 200 and 300cm, with the original 0cm node continuing to act as the surface node and the connection between the 300cm node and the original soil surface able to grow to an arbitrary large extent. As the snow pack grows, subroutine SNOWLAYER (Fig. 1) checks to see whether sufficient snow has accreted or melted to increase or decrease the number of snow nodes in use. Snow is not permitted to be below 2.5 cm depth. Subroutine OSUB contains the formulae for computing snow growth and melt, and calls SNOWLAYER (Fig. 1)

Snow growth is driven by inputted precipitation for each day and the user specified thresholds for the air temperature at which rain becomes snow, and the snow density (either a fixed value or a linear change of density with day of year). All snow for a given day is assumed to fall at midnight, and the user can specify a multiplier to account for under-catch if necessary (Rasmussen et al. 2012). When snow is present, the surface albedo A changes as a function of the age of the snow pack as $A = \frac{-9.8740\ln(d_{snow})+78.3434}{100}$ where d_{snow} is the days since the last snowfall based on regressions fitted to Fig A-4 of Anderson (2006). Snow thermal conductivity is estimated from Djachkova’s formula following Anderson (2006) $k_{snow} = 0.0442e^{5.181\rho_{snow}}$.

Snow melt occurs according to the heat load as computed by the heat budget for the substrate (including the snow), as well as melting due to rain. If the mean temperature between a given pair of soil nodes is greater than 0.4 °C, the change in mean temperature between nodes is computed and multiplied by the snow heat capacity (including the heat of fusion), snow density and distance between the nodes to obtain the heat input, and then divided by the heat of fusion to compute the mass of snow melted. Rain-melt is simulated following Anderson (2006) by multiplying the product of the rainfall and mean air temperature for the day by a rain melt factor (default value 0.0125) whenever rain falls at air temperatures above the snow threshold.

10. References

- Anderson E. 2006. Snow Accumulation and Ablation Model - SNOW-17
- Bird, R. B. et al. 2002. Transport Phenomena. - Wiley and Sons.
- Campbell GS. 1985. Soil Physics with Basic: Transport Models for Soil-Plant Systems. Amsterdam: Elsevier.
- Campbell GS, Jungbauer JDJ, Bidlake WR, Hungerford RD. 1994. Predicting the effect of temperature on soil thermal conductivity. *Soil Science* 158:307-313.
- Campbell GS, Norman JM. 1998. Environmental Biophysics. Springer.
- Carslaw HS, Jaeger JC. 1959. Conduction of heat in solids. Oxford University Press.
- Dave JV, Furakawa PM. 1967. Scattered radiation in the ozone absorption bands at selected levels of a terrestrial Rayleigh atmosphere. American Meteorological Society.
- Dave JV, Warten RM. 1968. Program for computing the Stokes parameters of the radiation emerging from a plane-parallel, non-absorbing, Rayleigh atmosphere. Palo Alto, California: IBM Scientific Center.
- Dozier J, Bruno J, Downey P. 1981. A faster solution to the horizon problem. *Computers and Geosciences* 7:145-151.
- Elterman L. 1968. UV, visible and IR attenuation for altitudes to 59 km. Bedford, Mass.: U. S. Airforce Cambridge Research Laboratory.
- . 1970. Vertical-attenuation model with eight surface meteorological ranges 2 to 13 kilometers. Bedford, Mass.: U. S. Airforce Cambridge Research Laboratory.
- Gates DM, Harrop WJ. 1963. Infrared transmission of the atmosphere to solar radiation. *Applied Optics* 2:887-898.
- Kearney MR, Porter WP. 2016. NicheMapR - an R package for biophysical modelling: the microclimate model. *Ecography* in review.
- Koepke P, Hess M, Schult I, Shettle EP. 1997. Global Aerosol Data Set. Hamburg: Max-Planck-Institut für Meteorologie.
- Linacre E. 1992. Climate data and resources. Routledge.
- McCullough EC, Porter WP. 1971. Computing clear day solar radiation spectra for the terrestrial ecological environment. *Ecology* 52:1008-1015.
- Neteler M, Bowman MH, Landa M, Metz M. 2012. GRASS GIS: A multi-purpose open source GIS. *Environmental Modelling & Software* 31:124-113.
- Rasmussen R, et al. 2012. How Well Are We Measuring Snow: The NOAA/FAA/NCAR Winter Precipitation Test Bed. *Bulletin of the American Meteorological Society* 93:811-829.
- Robinson N. 1966. Solar Radiation. Elsevier.
- Rozenberg GV. 1966. Twilight: A Study in Atmospheric Optics. Plenum Press.
- Tracy CR, Welch WR, Pinshow B, Kearney MR, Porter WP. 2016. Properties of air: A manual for use in biophysical ecology. Madison: The University of Wisconsin.
- Tracy CR, Welch WR, Porter WP. 1980. Properties of air: A manual for use in biophysical ecology. Madison: The University of Wisconsin.

Appendix 2 Microclimate Model Input Data

Michael R. Kearney

2016-04-18

Contents

Model mode settings	1
Time and location parameters	1
Air and wind vertical profile parameters	2
Radiation-related parameters	2
Terrain and shading parameters	2
Substrate profile settings	3
Time varying environmental data	3
Substrate properties	4
Soil moisture parameters	4
Snow model parameters	5
Intertidal simulation parameters (currently experimental and untested)	5

Model mode settings

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
writcsv	-	0 (off) or 1 (on)	make Fortran program write output as csv files
microdaily	-	0 (off) or 1 (on)	run in daily mode (initial conditions from previous day)
runshade	-	0 (off) or 1 (on)	run the model twice, once for each shade level
runmoist	-	0 (off) or 1 (on)	run soil moisture model
snowmodel	-	0 (off) or 1 (on)	run the snow model
hourly	-	0 (off) or 1 (on)	run the model from hourly weather inputs

Time and location parameters

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
julnum	days	positive integer	number of days to run the model
julday	day of year	1-365	vector of julian days (length must equal julnum)
idayst	-	1-julnum	start day (usually 1)
ida	-	1-julnum	end day (usually value of julnum)
HEMIS	-	1 (N) or 2 (S)	hemisphere to run
ALAT	degrees	0-90	latitude (degrees)
AMINUT	dec. minutes	0-60	latitude (minutes)
ALONG	degrees	0-180	longitude (degrees)
ALMINT	dec. minutes	0-60	longitude (minutes)
ALREF	degrees	0-180	reference longitude (degrees) for time zone

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
EC	-	0.0034 to 0.058	eccentricity of the earth's orbit (presently 0.0167238)

Air and wind vertical profile parameters

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
RUF	m	0.01-200	roughness height
Refhyt	m	0.50-10	reference height for air temp, wind speed and humidity input data
Usrhyt	m	> 0.005, < Refhyt	local height at which to compute air temp, wind speed and humidity
Z01	m	0 or (> Z02, < ZH2)	1st segment, roughness height ¹
Z02	m	0 or (> RUF, < Z01)	2nd segment, roughness height ¹
ZH1	m	0 or (> ZH2, < Refhyt)	1st segment, height above surface ¹
ZH2	m	0 or (> RUF, < ZH1)	2nd segment, height above surface ¹

¹Set to 0 if no experimental data

Radiation-related parameters

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
SLES	-	0-1	substrate longwave IR emissivity ¹
REFLS	-	0-1	substrate solar reflectivity ¹
CMH2O	cm	0.1-2	precipitable cm H ₂ O in air column
TAI	-	1-julnum	aerosol optical extinction coefficients ^{1,2}

¹Vector of length = julnum

²Vector of length 111 for wavelengths between 290 and 4000 nm. Wavelength increments are: 290,295,300,305,310,315,320,330,340,350,360,370,380,390,400,420,440,460,480,500, 520,540,560,580,600,620,640,660,680,700,720, 1120,1140,1160,1180,1200,1220,1240,1260,1280,1300,1320,1380,1400,1420,1440,1460,1480,1500,1540,1580,1600,1620,1640, 1660,1700,1720,1780,1800,1860,1900,1950,2000,2020,2050,2100,2120,2150,2200,2260,2300,2320,2350,2380,2400,2420,2450, 2490,2500,2600,2700,2800,2900,3000,3100,3200,3300,3400,3500,3600,3700,3800,3900,4000

Terrain and shading parameters

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
ALTT	m	0-10,000	altitude
slope	decimal degrees	0-90	slope
azmuth	decimal degrees	0-360	aspect (0 is north)
hori	decimal degrees	0-90	horizon angles ¹
VIEWF	-	0-1	view factor to sky ²
MINSHADES	%	0-100	minimum shade ³
MAXSHADES	%	0-100	maximum shade ^{3,4}
PCTWETS	%	0-100	percentage of substrate unit area that is wet ^{3,5}

¹angle to the effective horizon, due to e.g. hills, buildings, in 24 directions from 0 degrees azimuth (north) clockwise in 15 degree intervals

²Fraction of sky obscured by terrain

³vector of length = julnum

⁴must be greater than minimum shade

⁵drives evaporation - set internally when runmoist==1

Substrate profile settings

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
DEP	cm	0-1000	vector of 10 depths ¹
ERR	-	>0	integrator error (typically 1.5-2)
tannul	°C	-80 - +60	annual mean temperature
soilinit	°C	-80 - +60	initial substrate temperature profile ¹

¹Acting as nodes for substrate heat budget calculations. Must start at 0 and nodes must be closely spaced near the surface. Typical depth profile c(0., 2.5, 5., 10., 15, 20, 30, 50, 100, 200)

²Vector of length 10, corresponding to depths specified in variable DEP

Time varying environmental data

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
TIMINS	h	0-23	time of minima for air temp, wind, humidity and cloud cover ¹
TIMAXS	h	0-23	time of maxima for air temp, wind, humidity and cloud cover ²
TMINN	°C	-80 - +60	minimum air temperature (at reference height, Refhyt) ³
TMAXX	°C	-80 - +60	maximum air temperature (at reference height, Refhyt) ³
RHMINN	%	0-100	minimum relative humidity (at reference height, Refhyt) ³
RHMAXX	%	0-100	maximum relative humidity (at reference height, Refhyt) ³
WNMINN	m s ⁻¹	0-100	minimum wind speed (at reference height, Refhyt) ³
WNMAXX	m s ⁻¹	0-100	maximum wind speed (at reference height, Refhyt) ³
CCMINN	%	0-100	minimum cloud cover; vector of length = julnum
CCMAXX	%	0-100	maximum cloud cover; vector of length = julnum
RAINFALL	mm	0-2000	daily total rainfall; vector of length = julnum
tannulrun	°C	-80 - +60	daily deep soil temperature; vector of length = julnum
moists	decimal %	0-1	predefined soil daily moisture profile through time ⁴
TAIRhr	°C	-80 - +60	hourly air temperature (at reference height, Refhyt) ⁵
RHhr	%	0-100	hourly relative humidity (at reference height, Refhyt) ⁵
WNhr	m s ⁻¹	0-100	hourly wind speed (at reference height, Refhyt) ⁵
CLDhr	%	0-100	hourly cloud cover ⁵
SOLRhr	W m ²	0-1367	hourly solar radiation ⁵
RAINhr	mm	0-2000	hourly rainfall ⁵

¹Vector of 4 integers, air temp & wind mins relative to sunrise, humidity and cloud cover mins relative to solar noon - typical TIMINS vector c(0, 0, 1, 1).

²Vector of 4 integers, air temp & wind maxs relative to solar noon, humidity and cloud cover maxs relative

to sunrise - typical TIMAXS vector c(1, 1, 0, 0).

³Vector of length = julnum.

⁴Matrix of 10 rows (depths) and julnum columns. The first column is used to specify the initial soil moisture when running the soil moisture model.

⁵Vector of length = julnum*24.

Substrate properties

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
Numtypes	-	1-10	number of substrate types
Nodes ^{1,2}	-	1-10	nodes where substrate type transitions occur
soilprops[,1] ³	Mg m ⁻³	>0	bulk density (must not exceed mineral density)
soilprops[,2] ³	m ³ m ⁻³	>0	volumetric water content at saturation ⁴
soilprops[,3] ³	%	0-100	clay content
soilprops[,4] ³	W m ⁻¹ K ⁻¹	>0	thermal conductivity
soilprops[,5] ³	J kg ⁻¹ K ⁻¹	>0	specific heat capacity
soilprops[,6] ³	Mg m ⁻³	>0	mineral density

¹Matrix of 10 rows (depths) and julnum columns

²The number of nodes specified should correspond with the variable numtypes, and in turn with the number of substrate types specified in soilprops in the soil props matrix. E.g. for a uniform profile, the first row of Nodes would be 10 (deepest soil node - the subsequent nodes all being left as 0), Numtypes would be 1, and only one and only the first row of soilprops would be filled, the rest being zero. At the other extreme, if all nodes were to have a different substrate type, values for a given row of each Nodes column would be 1, 2, ..., 10 and a different value would be specified for each row of each soilprops column.

³Matrix of 10 rows (depths) and 6 columns

⁴at a matric potential of 0.1 bar

Soil moisture parameters

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
PE	J kg ⁻¹	0.7-3.7	air entry water potential; vector of 19 values ¹
KS	kg s m ⁻³	1 x 10 ⁻⁵	saturated conductivity; vector of 19 values ¹
BB	-	1.7-7.6	Campbell's 'b' parameter; vector of 19 values ¹
BD	Mg m ⁻³	>0	bulk density (should be matched to same variable in the soilprops matrix)
L	m ³ m ⁻³	0-90	root density; vector of 19 values ¹
LAI	-	0-90	leaf area index, used to partition transpiration/evaporation from PET
rainmult	-	>0	rainfall multiplier to impose a catchment
maxpool	mm	>0	max depth for surface water pooling ²
evenrain	-	1 or 2	even rainfall over 24hrs (1) or one event at midnight (2)

¹The 19 values represent the 10 depths as specified in the DEP vector, and an extra 9 depths in between.

²To account for runoff - can be used to simulate a wetland.

Snow model parameters

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
snowtemp	°C	-80 - 80	temperature at which precipitation falls as snow
snowdens	Mg m ⁻³	0.05-1	snow density
densfun	-	-	slope & intercept of linear model of snow density ¹
snowmelt	decimal %	0-1	proportion of calculated snowmelt that doesn't refreeze
undercatch	-	>=1	undercatch multiplier for converting rainfall to snow
rainmelt	-	0-2	parameter for rain melting snow ²

¹As a function of day of year (1,365), in units of Mg m⁻³ - if it is c(0,0) then fixed density used

²For equation from Anderson's SNOW-17 model that melts snow with rainfall as a function of air temp (typical value 1.25)

Intertidal simulation parameters (currently experimental and untested)

<i>Name</i>	<i>Units</i>	<i>Allowed Range</i>	<i>Description</i>
tides[1:julnum,1]	-	0 or 1	tide in (1) or out(0); vector of length julnum
tides[1:julnum,2]	°C	-80 - 80	sea water temperature; vector of length julnum
tides[1:julnum,3]	-	-	% surface wetness from wave splash; vector of length julnum

Appendix 3 Inputs and outputs of the NicheMapR microclimate model

Michael R. Kearney

2016-04-18

Contents

This vignette runs through the core inputs and outputs of the NicheMapR microclimate model, using monthly mean inputs for Madison Wisconsin, USA. It provides a basis for developing scripts and functions that are customised to particular databases of climate/weather, terrain, vegetation and soil properties. See also the [microclimate-hourly-input-example](#) vignette for an illustration of how to set the model to run from hourly input across all days of a year. It is Appendix 3 of the NicheMapR microclimate software note, Porter and Kearney (2016). NicheMapR - an R package for biophysical modelling: the microclimate model. *Ecography* x(x) p. x-x.

Model mode settings

```
library(NicheMapR) # load the NicheMapR package

writescsv<-0 # make Fortran program write output as csv files
microdaily<-0 # run microclimate model as normal, where each day is iterated
# 3 times starting with the initial condition of uniform soil
# temp at mean monthly temperature
runshade<-1 # run the model twice, once for each shade level (1) or just for
#the first shade level (0)?
runmoist<-1 # run soil moisture model (0=no, 1=yes)?
snowmodel<-1 # run the snow model (0=no, 1=yes)? - note that this runs slower
hourly<-0 # run from hourly weather input (0=no, 1=yes)?
```

Time and location parameters

```
julnum<-12 # number of time intervals to generate predictions for over
# a year (must be 12 <= x <=365)
julday<-c(15.,46.,74.,105.,135.,166.,196.,227.,258.,288.,319.,349.) # middle
# day of each month
idayst <- 1 # start month
ida<-12 # end month
HEMIS <- 1 # chose hemisphere
ALAT <- 43 # degrees latitude
AMINUT <- 4.383 # minutes latitude
ALONG <- 89 # degrees longitude
ALMINT <- 24.074 # minutes latitude
ALREF <- 89 # reference longitude for time zone
EC <- 0.0167238 # Eccentricity of the earth's orbit (current value 0.0167238, ranges
# between 0.0034 to 0.058)
```

Air and wind vertical profile parameters

```
RUF <- 0.004 # Roughness height (m), , e.g. sand is 0.05, grass
# may be 2.0, current allowed range: 0.001 (snow) - 2.0 cm.
Refhyt <- 2 # Reference height (m), reference height at which air
# temperature, wind speed and relative humidity input data
# are measured
Ushrht <- 0.01 # local height (m) at which air temperature, relative humidity
# and wind speed calculatinos will be made
# Next four parameters are segmented velocity profiles due to bushes, rocks etc.
# on the surface
#IF NO EXPERIMENTAL WIND PROFILE DATA SET ALL THESE TO ZERO! (then roughness
# height is based on the parameter RUF)
Z01 <- 0. # Top (1st) segment roughness height(m)
Z02 <- 0. # 2nd segment roughness height(m)
ZH1 <- 0. # Top of (1st) segment, height above surface(m)
ZH2 <- 0. # 2nd segment, height above surface(m)
```

Radiation-related parameters

```
SLE <- 0.96 # substrate longwave IR emissivity (decimal %), typically
# close to 1
REFL<-0.10 # substrate solar reflectivity (decimal %)
CMH2O <- 1. # precipitable cm H2O in air column, 0.1 = VERY DRY;
# 1.0 = MOIST AIR CONDITIONS; 2.0 = HUMID, TROPICAL
# CONDITIONS (note this is for the whole atmospheric profile,
#not just near the ground)
# Aerosol extinction coefficient profile
# the values extracted from GADS for Madison
TAI<-c(0.2699,0.2661,0.2624,0.2588,0.2552,0.2516,0.2481,0.2413,0.2346,0.2281,
0.2218,0.2157,0.2098,0.2040,0.1984,0.1877,0.1776,0.1681,0.1591,0.1507,0.1428,
0.1354,0.1285,0.1219,0.1158,0.1101,0.1048,0.0998,0.0951,0.0907,0.0866,0.0828,
0.0793,0.0760,0.0729,0.0700,0.0673,0.0649,0.0625,0.0604,0.0584,0.0565,0.0547,
0.0531,0.0516,0.0502,0.0464,0.0453,0.0443,0.0433,0.0423,0.0414,0.0406,0.0398,
0.0390,0.0382,0.0375,0.0368,0.0361,0.0341,0.0335,0.0328,0.0322,0.0316,0.0309,
0.0303,0.0291,0.0278,0.0272,0.0266,0.0260,0.0253,0.0241,0.0235,0.0217,0.0211,
0.0193,0.0181,0.0168,0.0155,0.0149,0.0142,0.0131,0.0127,0.0121,0.0112,0.0102,
0.0097,0.0095,0.0092,0.0090,0.0088,0.0087,0.0086,0.0085,0.0084,0.0085,0.0090,
0.0097,0.0106,0.0114,0.0119,0.0121,0.0117,0.0109,0.0096,0.0080,0.0064,0.0056,
0.0061,0.0093)
```

Terrain and shading parameters

```
ALTT<-226 # altitude (m)
slope<-0. # slope (degrees, range 0-90)
azimuth<-180. # aspect (degrees, 0 = North, range 0-360)
hori<-rep(0,24) # enter the horizon angles (degrees) so that they go from
# 0 degrees azimuth (north) clockwise in 15 degree intervals
VIEWF <- 1-sum(sin(hori*pi/180))/length(hori) # convert horizon angles to
```

```

# radians and calc view factor(s)
minshade<-0. # minimum available shade (%)
maxshade<-90. # maximum available shade (%)
PCTWET<-0 # percentage of surface area acting as a free water surface (%)

```

Soil profile settings

```

DEP <- c(0., 2.5, 5., 10., 15., 20., 30., 50., 100., 200.) # Soil nodes (cm)
# - keep spacing close near the surface, last value is where it is assumed
# that the soil temperature is at the annual mean air temperature
ERR <- 2.0 # Integrator error for soil temperature calculations

```

Time varying environmental data

```

TIMINS <- c(0, 0, 1, 1) # time of minima for air temp, wind, humidity and cloud
# cover (h), air & wind mins relative to sunrise, humidity and cloud cover mins
# relative to solar noon
TIMAXS <- c(1, 1, 0, 0) # time of maxima for air temp, wind, humidity and cloud cover
# (h), air temp & wind maxs relative to solar noon, humidity and cloud cover maxs
# relative to sunrise
TMINN<-c(-14.3,-12.1,-5.1,1.2,6.9,12.3,15.2,13.6,8.9,3,-3.2,-10.6) # minimum air
# temperatures (deg C)
TMAXX<-c(-3.2,0.1,6.8,14.6,21.3,26.4,29,27.7,23.3,16.6,7.8,-0.4) # maximum air
# temperatures (deg C)
RHMINN<-c(50.2,48.4,48.7,40.8,40,42.1,45.5,47.3,47.6,45,51.3,52.8) # min relative
# humidity (%)
RHMAXX<-c(100,100,100,100,100,100,100,100,100,100,100,100) # max relative humidity (%)
WNMINN<-c(4.9,4.8,5.2,5.3,4.6,4.3,3.8,3.7,4,4.6,4.9,4.8) # min wind speed (m/s)
WNMAXX<-c(4.9,4.8,5.2,5.3,4.6,4.3,3.8,3.7,4,4.6,4.9,4.8) # max wind speed (m/s)
CCMINN<-c(50.3,47,48.2,47.5,40.9,35.7,34.1,36.6,42.6,48.4,61.1,60.1) # min cloud cover (%)
CCMAXX<-c(50.3,47,48.2,47.5,40.9,35.7,34.1,36.6,42.6,48.4,61.1,60.1) # max cloud cover (%)
RAINFALL<-c(28,28.2,54.6,79.7,81.3,100.1,101.3,102.5,89.7,62.4,54.9,41.2) # monthly
# mean rainfall (mm)
TAIRhr=rep(0,24*julnum) # hourly air temperatures (deg C), not used unless 'hourly=1'
RHhr=rep(0,24*julnum) # hourly relative humidity (%), not used unless 'hourly=1'
WNhr=rep(0,24*julnum) # hourly wind speed (m/s), not used unless 'hourly=1'
CLDhr=rep(0,24*julnum) # hourly cloud cover (%), not used unless 'hourly=1'
SOLRhr=rep(0,24*julnum) # hourly solar radiation (W/m2, not used unless 'hourly=1'
RAINhr=rep(0,24*julnum) # hourly rainfall (mm), not used unless 'hourly=1'
tannul<-mean(c(TMAXX,TMINN)) # annual mean temperature for getting monthly deep soil
# temperature (deg C)
tannulrun<-rep(tannul,julnum) # monthly deep soil temperature (2m) (deg C)
SoilMoist<-c(0.42,0.42,0.42,0.43,0.44,0.44,0.43,0.42,0.41,0.42,0.42,0.43) # soil moisture
# (decimal %, 1 means saturated)
# creating the arrays of environmental variables that are assumed not to change with month
# for this simulation
MAXSHADES <- rep(maxshade,julnum) # daily max shade (%)
MINSHADES <- rep(minshade,julnum) # daily min shade (%)
SLES <- rep(SLE,julnum) # set up vector of ground emissivities for each day

```

```
REFLS<-rep(REFL,julnum) # set up vector of soil reflectances for each day
PCTWET<-rep(PCTWET,julnum) # set up vector of soil wetness for each day
```

Soil properties

```
# set up a profile of soil properties with depth for each day to be run
Numtyps <- 1 # number of soil types
Nodes <- matrix(data = 0, nrow = 10, ncol = julnum) # array of all
# possible soil nodes
Nodes[1,1:julnum]<-10 # deepest node for first substrate type

# soil thermal parameters
Thcond <- 1.25 # soil minerals thermal conductivity (W/mC)
Density <- 2560. # soil minerals density (kg/m3)
SpecHeat <- 870. # soil minerals specific heat (J/kg-K)
BulkDensity <- 2560. # soil bulk density (kg/m3)
SatWater <- 0.26 # volumetric water content at saturation (0.1 bar matric
# potential) (m3/m3)
Clay <- 20 # clay content for matric potential calculations (%)
#SoilMoist<-rep(SoilMoist,timeinterval) # soil moisture
Density<-Density/1000 # density of minerals - convert to Mg/m3
BulkDensity<-BulkDensity/1000 # density of minerals - convert to Mg/m3

# now make the depth-specific soil properties matrix
# columns are:
#1) bulk density (Mg/m3)
#2) volumetric water content at saturation (0.1 bar matric potential) (m3/m3)
#3) clay content (%)
#4) thermal conductivity (W/mK)
#5) specific heat capacity (J/kg-K)
#6) mineral density (Mg/m3)
soilprops<-matrix(data = 0, nrow = 10, ncol = 6) # create an empty soil
# properties matrix
soilprops[1,1]<-BulkDensity # insert soil bulk density to profile 1
soilprops[1,2]<-SatWater # insert saturated water content to profile 1
soilprops[1,3]<-Clay # insert percent clay to profile 1
soilprops[1,4]<-Thcond # insert thermal conductivity to profile 1
soilprops[1,5]<-SpecHeat # insert specific heat to profile 1
soilprops[1,6]<-Density # insert mineral density to profile 1
soilinit<-rep(tannul,20) # make initial soil temps equal to mean annual
```

Soil moisture parameters

```
# note that these are set for sand (Table 9.1 in Campbell and Norman, 1995)
PE<-rep(0.7,19) #air entry potential J/kg
KS<-rep(0.0058,19) #saturated conductivity, kg s/m3
BB<-rep(1.7,19) #soil 'b' parameter
BD<-rep(1.3,19) # soil bulk density, Mg/m3
L<-c(0,0,8.18990859,7.991299442,7.796891252,7.420411664,7.059944542,
```

```

6.385001059,5.768074989,4.816673431,4.0121088,1.833554792,0.946862989,
0.635260544,0.804575,0.43525621,0.366052856,0,0)*10000 # root density
# at each node, mm/m3 (from Campell 1985 Soil Physics with Basic, p. 131)
LAI<-0.1 # leaf area index, used to partition traspiration/evaporation
# from PET
rainmult<-1 # rainfall multiplier to impose catchment
maxpool<-10 # max depth for water pooling on the surface, mm
# (to account for runoff)
evenrain<-1 # spread daily rainfall evenly across 24hrs (1) or
# one event at midnight (2)
SoilMoist_Init<-rep(0.2,10) # initial soil water content for each
# node, m3/m3
moists<-matrix(nrow=10, ncol = julnum, data=0) # set up an empty vector
# for soil moisture values through time
moists[1:10,]<-SoilMoist_Init # insert inital soil moisture

```

Snow model parameters

```

snowtemp<-1.5 # temperature at which precipitation falls as snow
# (used for snow model)
snowdens<-0.375 # snow density (mg/m3)
densfun<-c(0,0) # slope and intercept of linear model of snow density
# as a function of day of year - if it is c(0,0) then fixed density used
snowmelt<-0.9 # proportion of calculated snowmelt that doesn't refreeze
undercatch<-1. # undercatch multiplier for converting rainfall to snow
rainmelt<-0.0125 # parameter in equation from Anderson's SNOW-17 model
# that melts snow with rainfall as a function of air temp

```

Intertidal simulation parameters (currently experimental and untested)

```

# intertidal simulation input vector (col 1 = tide in(1)/out(0),
# col 2 = sea water temperature in deg C, col 3 = % wet from wave splash)
tides<-matrix(data = 0., nrow = 24*julnum, ncol = 3) # matrix for tides

```

Collating all input parameters

```

# input parameter vector
microinput<-c(julnum,RUF,ERR,Usrhyt,Refhyt,Numtyps,Z01,Z02,ZH1,ZH2,idayst,
ida,HEMIS,ALAT,AMINUT,ALONG,ALMINT,ALREF,slope,azimuth,ALTT,CMH20,microdaily,
tannul,EC,VIEWF,snowtemp,snowdens,snowmelt,undercatch,rainmult,runshade,
runmoist,maxpool,evenrain,snowmodel,rainmelt,writescv,densfun,hourly)

# Final input list - all these variables are expected by the input argument of
# the Fortran microclimate subroutine
micro<-list(microinput=microinput,tides=tides,julday=julday,SLES=SLES,DEP=DEP,
Nodes=Nodes,MAXSHADES=MAXSHADES,MINSHADES=MINSHADES,TIMAXS=TIMAXS,TIMINS=TIMINS,
TMAXX=TMAXX,TMINN=TMINN,RHMAXX=RHMAXX,RHMINN=RHMINN,CCMAXX=CCMAXX,CCMINN=CCMINN,
WNMAXX=WNMAXX,WNMINN=WNMINN,TAIRhr=TAIRhr,RHhr=RHhr,WNhr=WNhr,CLDhr=CLDhr,

```

```
SOLRhr=SOLRhr,RAINhr=RAINhr,REFLS=REFLS,PCTWET=PCTWET,soilinit=soilinit,horihori,
TAI=TAI,soilprops=soilprops,moists=moists,RAINFALL=RAINFALL,tannulrun=tannulrun,
PE=PE,KS=KS,BB=BB,BD=BD,L=L,LAI=LAI)
```

Executing the microclimate model

```
microut<-microclimate(micro) # run the model in Fortran
```

Retrieving the data

```
metout<-as.data.frame(microut$metout[1:(julnum*24),]) # retrieve above ground
# microclimatic conditions, min shade
shadmet<-as.data.frame(microut$shadmet[1:(julnum*24),]) # retrieve above ground
# microclimatic conditions, max shade
soil<-as.data.frame(microut$soil[1:(julnum*24),]) # retrieve soil temperatures,
# minimum shade
shadsoil<-as.data.frame(microut$shadsoil[1:(julnum*24),]) # retrieve soil
# temperatures, maximum shade
soilmoist<-as.data.frame(microut$soilmoist[1:(julnum*24),]) # retrieve soil
# moisture, minimum shade
shadmoist<-as.data.frame(microut$shadmoist[1:(julnum*24),]) # retrieve soil
# moisture, maximum shade
humid<-as.data.frame(microut$humid[1:(julnum*24),]) # retrieve soil humidity,
# minimum shade
shadhumid<-as.data.frame(microut$shadhumid[1:(julnum*24),]) # retrieve soil
# humidity, maximum shade
soilpot<-as.data.frame(microut$soilpot[1:(julnum*24),]) # retrieve soil water
# potential, minimum shade
shadpot<-as.data.frame(microut$shadpot[1:(julnum*24),]) # retrieve soil water
# potential, maximum shade
```

Plotting the results

```
# append dates
days<-rep(seq(1,12),24)
days<-days[order(days)]
dates<-days+metout$TIME/60/24-1 # dates for hourly output
dates2<-seq(1,12,1) # dates for daily output

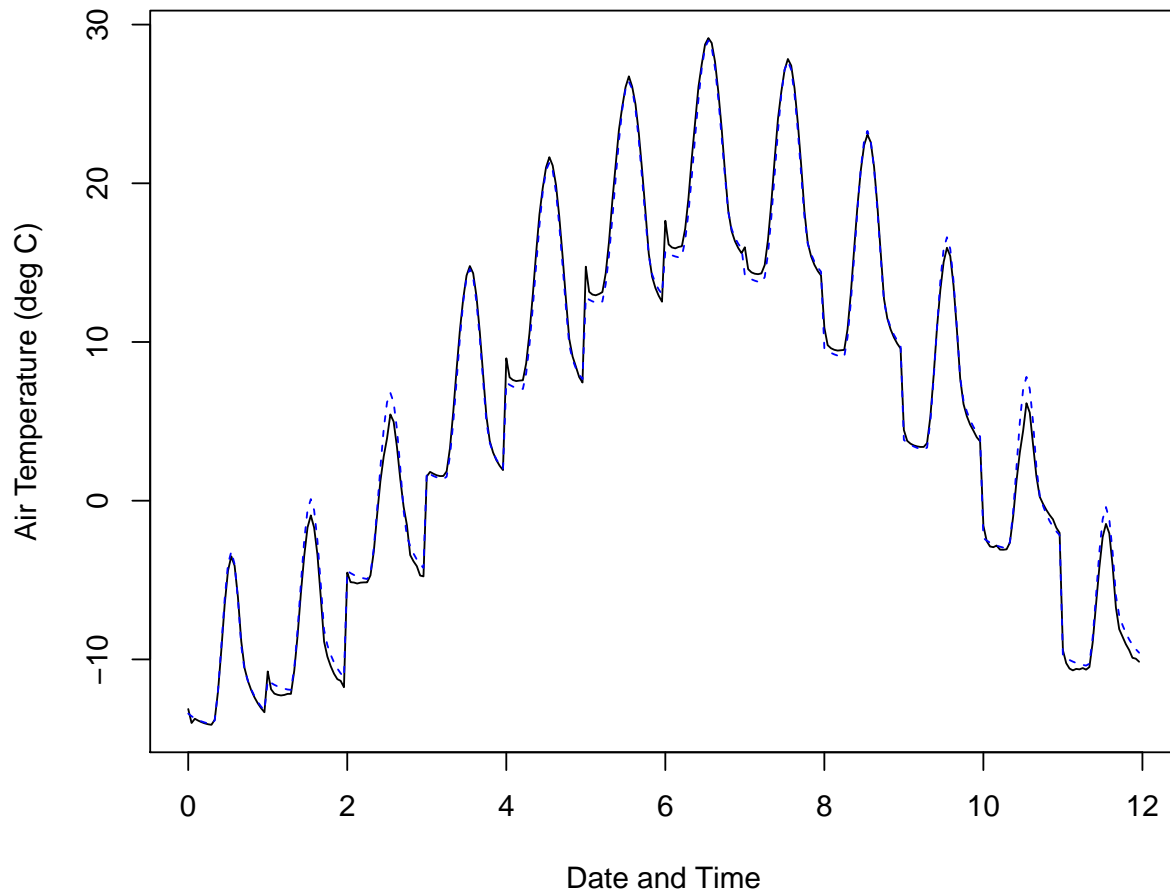
plotmetout<-cbind(dates,metout)
plotsoil<-cbind(dates,soil)
plotshadmet<-cbind(dates,shadmet)
plotshadsoil<-cbind(dates,shadsoil)

minshade<-micro$MINSHADES[1]
maxshade<-micro$MAXSHADES[1]

# plotting above-ground conditions in minimum shade
```

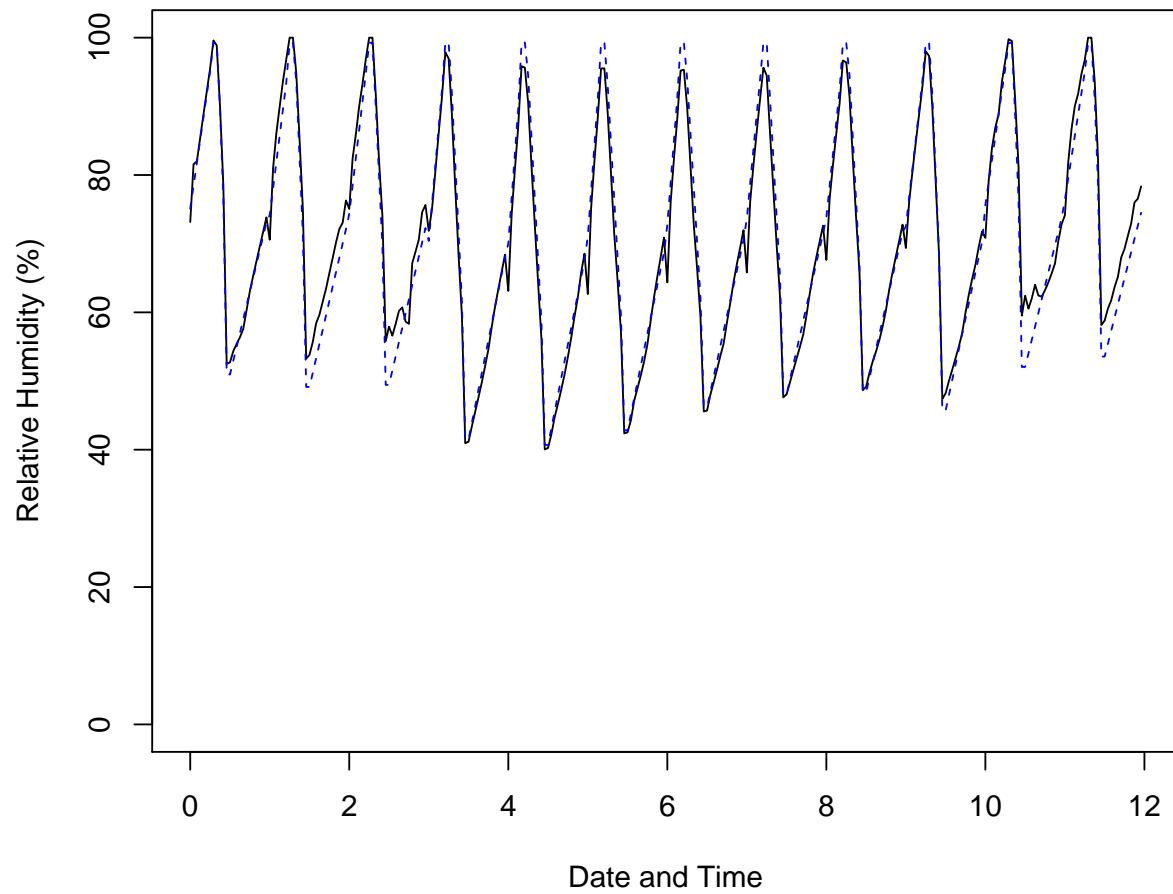
```
with(plotmetout,{plot(TALOC ~ dates,xlab = "Date and Time", ylab = "Air Temperature (deg C)"
, type = "l",main=paste("air temperature, ",minshade,"% shade",sep=""))})
with(plotmetout,{points(TAREF ~ dates,xlab = "Date and Time", ylab = "Air Temperature (deg C)"
, type = "l",lty=2,col='blue')}})
```

air temperature, 0% shade



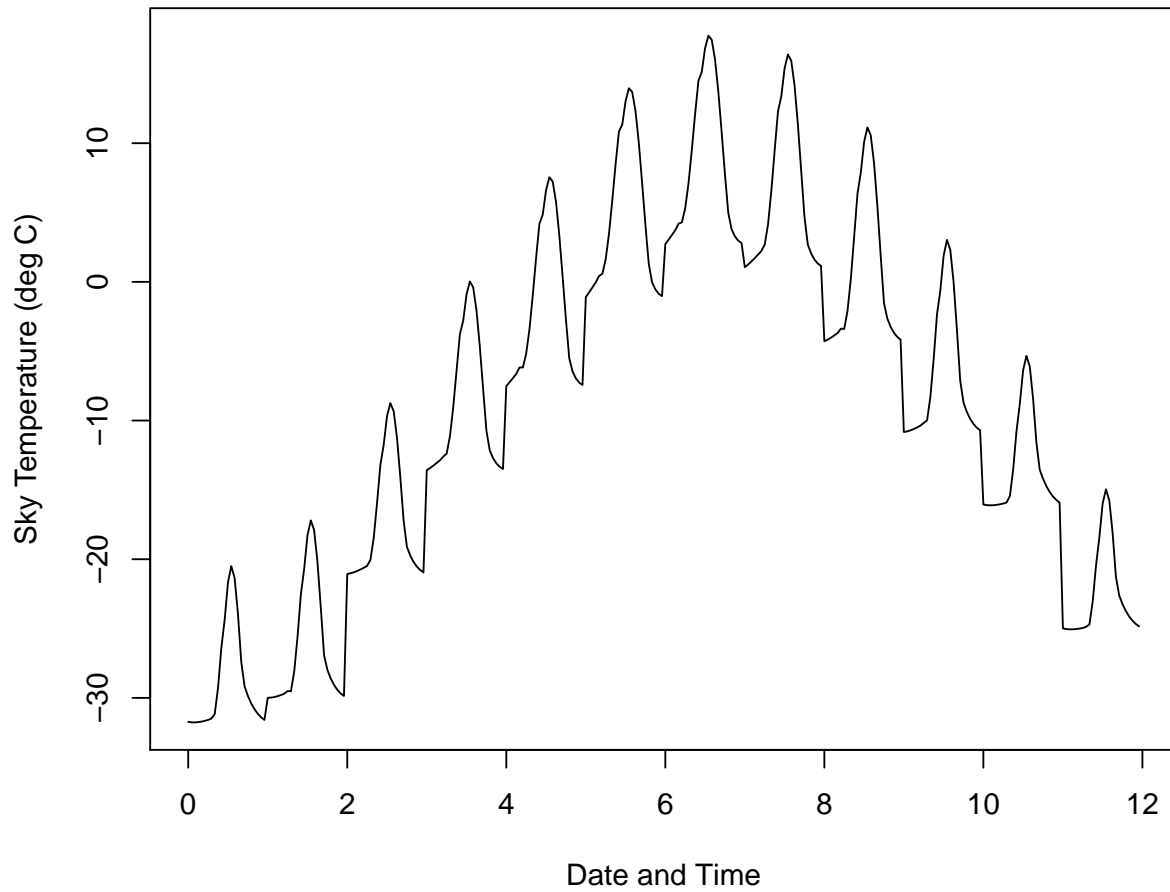
```
with(plotmetout,{plot(RHLOC ~ dates,xlab = "Date and Time", ylab = "Relative Humidity (%)"
, type = "l",ylim=c(0,100),main=paste("humidity, ",minshade,"% shade",sep=""))})
with(plotmetout,{points(RH ~ dates,xlab = "Date and Time", ylab = "Relative Humidity (%)"
, type = "l",col='blue',lty=2,ylim=c(0,100))})
```

humidity, 0% shade



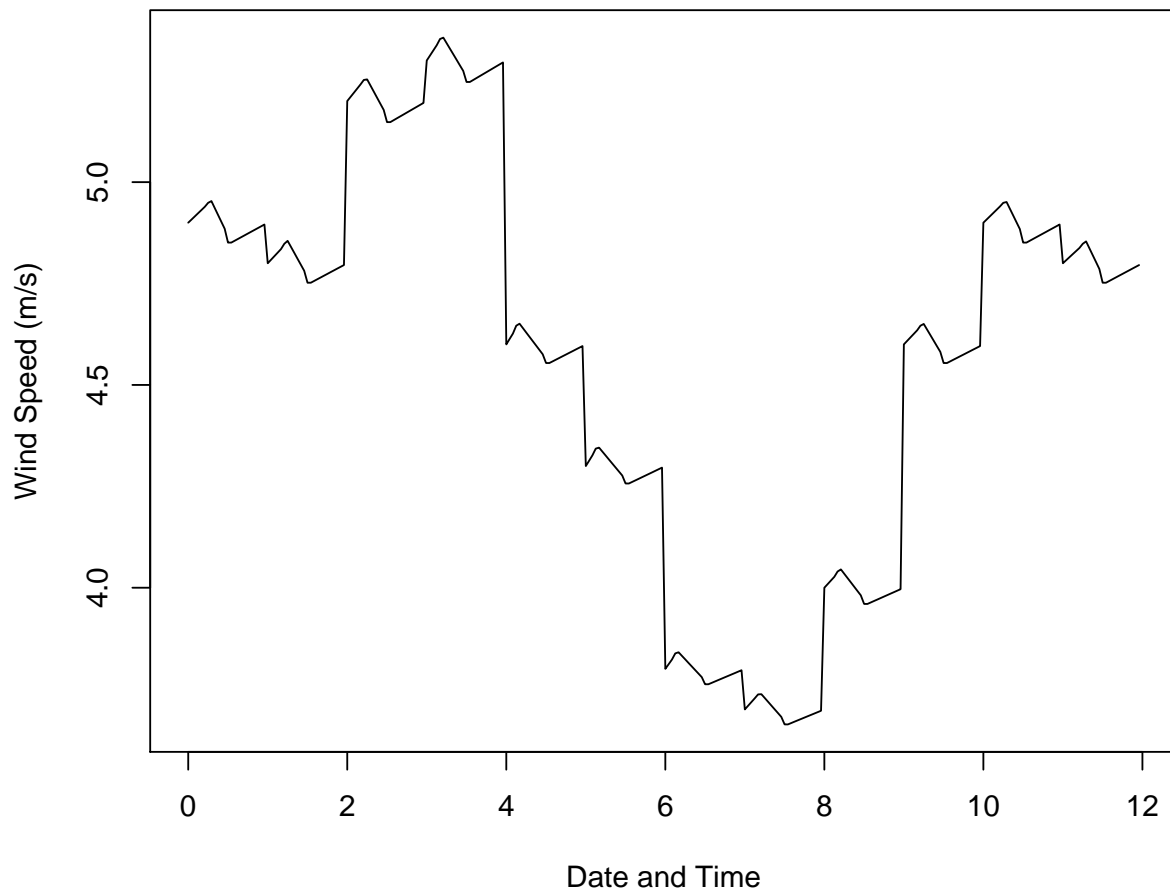
```
with(plotmetout,{plot(TSKYC ~ dates,xlab = "Date and Time", ylab = "Sky Temperature (deg C)"  
, type = "l",main=paste("sky temperature, ",minshade,"% shade",sep=""))})
```

sky temperature, 0% shade



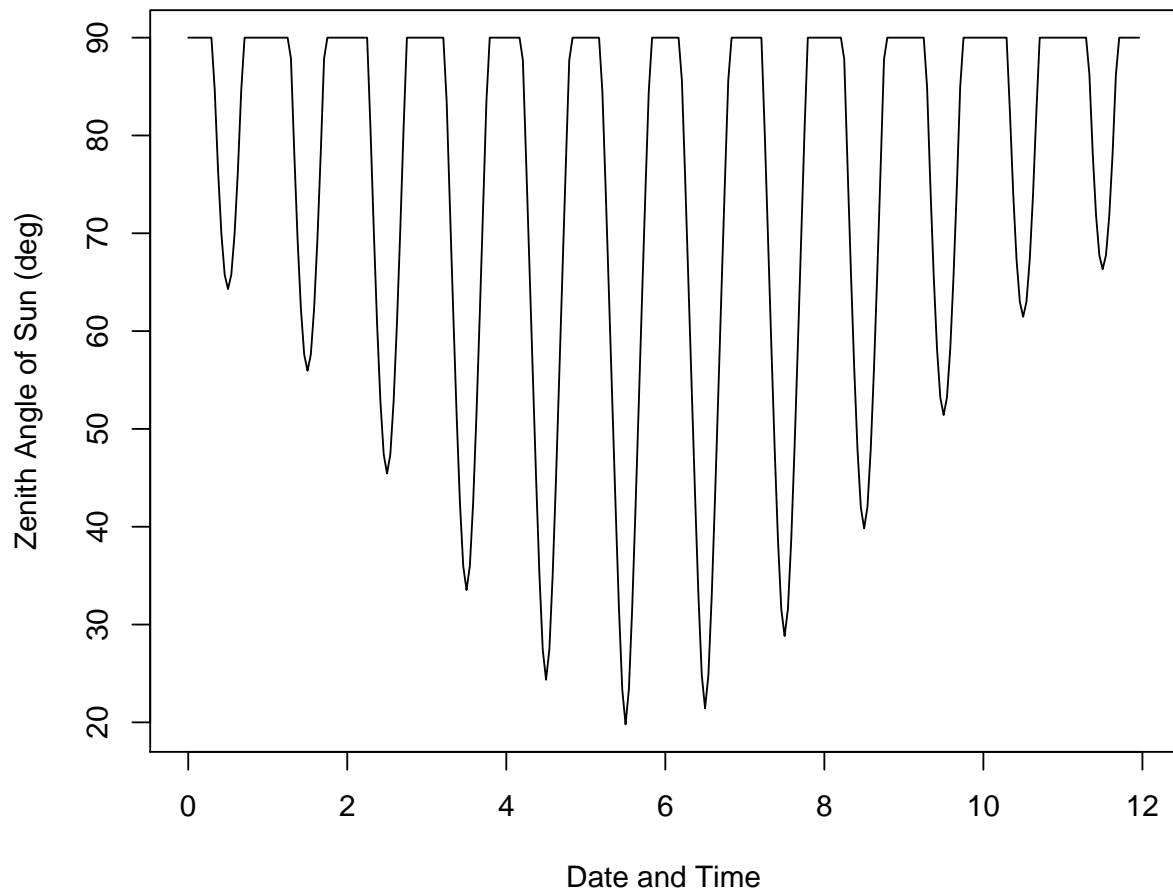
```
with(plotmetout,{plot(VREF ~ dates,xlab = "Date and Time", ylab = "Wind Speed (m/s)"  
, type = "l",main="wind speed")})  
with(plotmetout,{points(VLOC ~ dates,xlab = "Date and Time", ylab = "Wind Speed (m/s)"  
, type = "l",lty=2,col='blue')})
```

wind speed



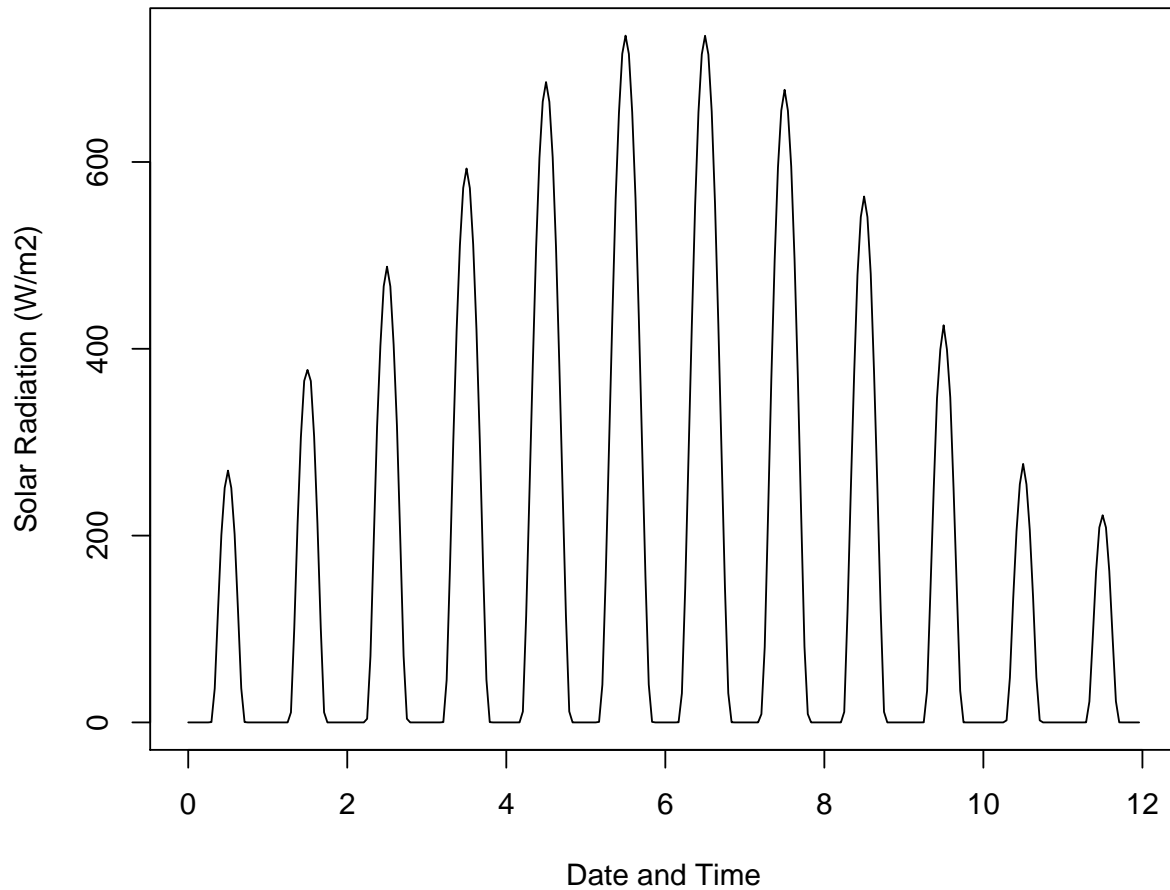
```
with(plotmetout,{plot(ZEN ~ dates,xlab = "Date and Time", ylab = "Zenith Angle of Sun (deg)"  
, type = "l",main="solar angle, sun")})
```

solar angle, sun



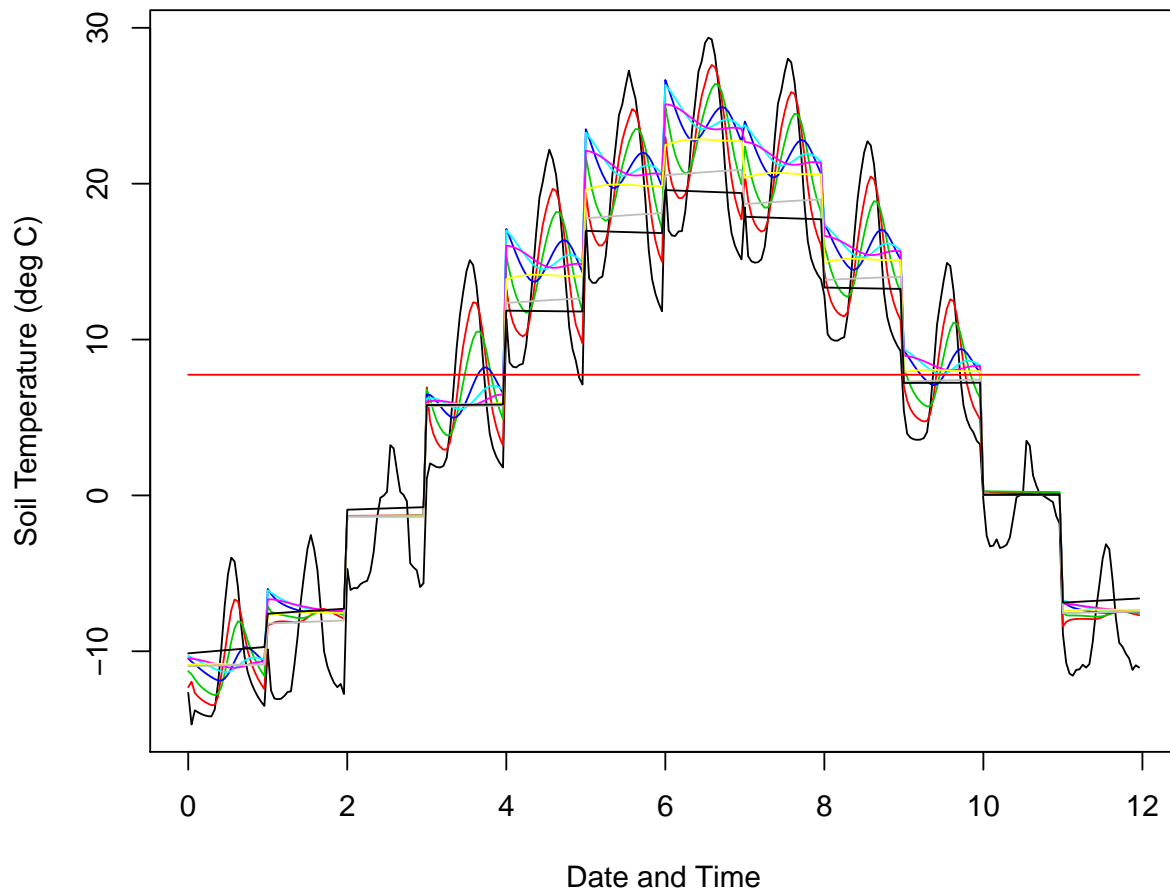
```
with(plotmetout,{plot(SOLR ~ dates,xlab = "Date and Time", ylab = "Solar Radiation (W/m2)"  
, type = "l",main="solar radiation"))})
```

solar radiation



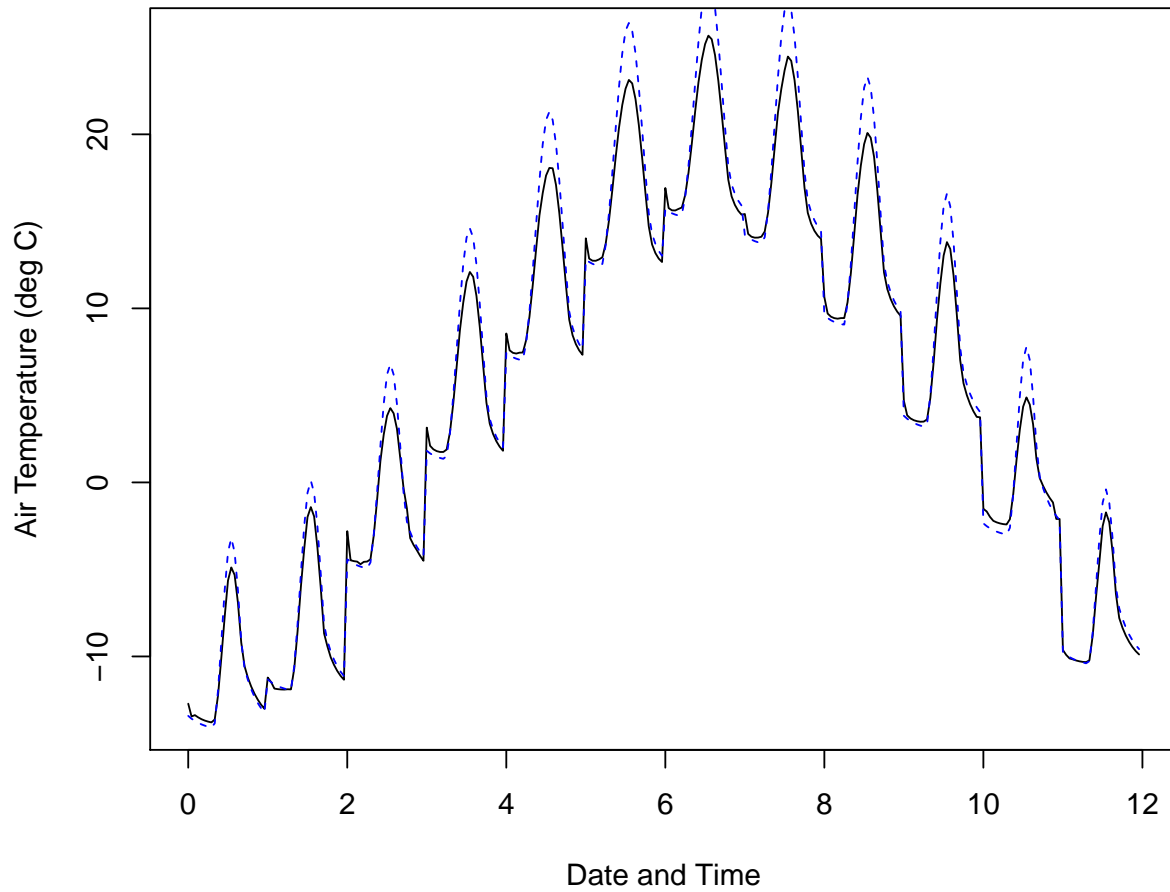
```
# plotting soil temperature for minimum shade
for(i in 1:10){
  if(i==1){
    plot(plotsoil[,i+3]~plotsoil[,1],xlab = "Date and Time", ylab = "Soil Temperature (deg C)"
    ,col=i,type = "l",main=paste("soil temperature ",minshade,"% shade",sep=""))
  }else{
    points(plotsoil[,i+3]~plotsoil[,1],xlab = "Date and Time", ylab = "Soil Temperature
    (deg C)",col=i,type = "l")
  }
}
```

soil temperature 0% shade



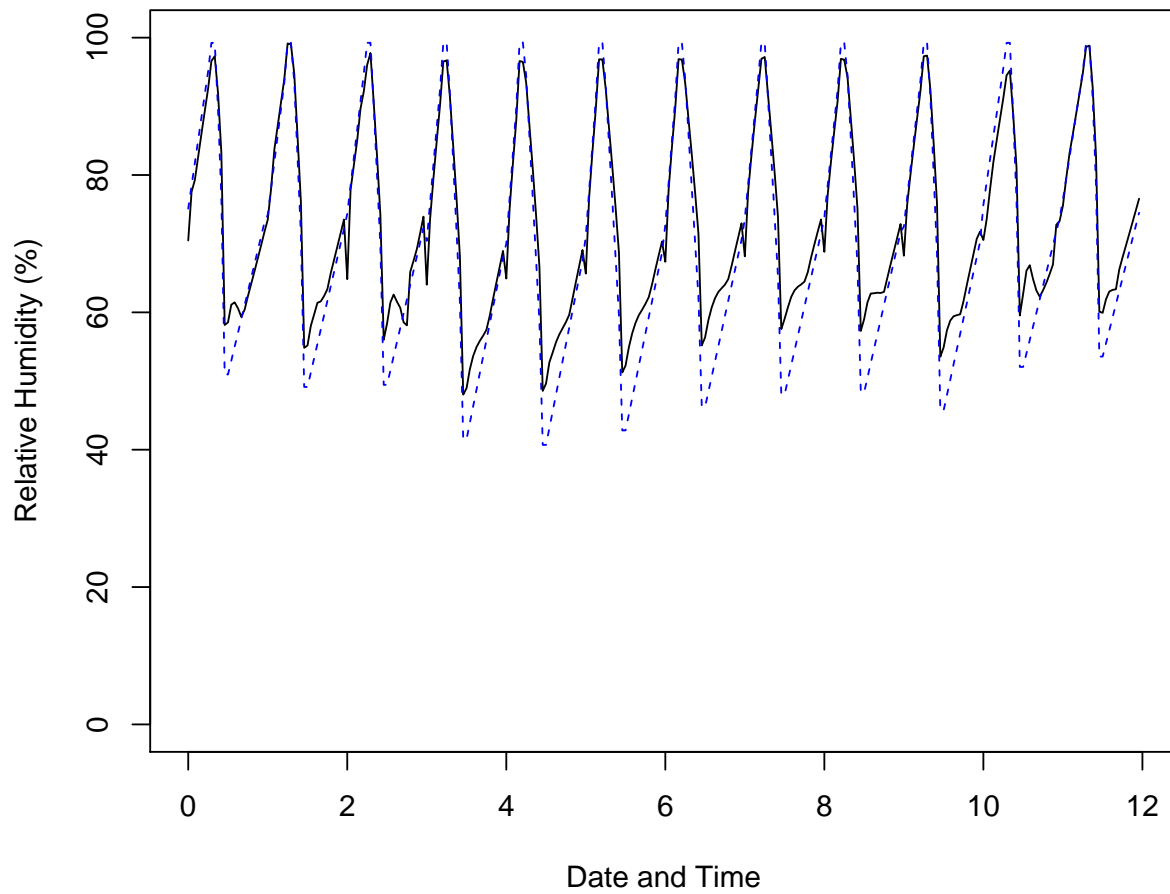
```
# plotting above-ground conditions in maximum shade  
with(plotshadmet,{plot(TALOC ~ dates,xlab = "Date and Time", ylab = "Air Temperature (deg C)"  
, type = "l",main=paste("air temperature, ",maxshade,"% shade",sep=""))})  
with(plotshadmet,{points(TAREF ~ dates,xlab = "Date and Time", ylab = "Air Temperature (deg C)"  
, type = "l",lty=2,col='blue')})
```

air temperature, 90% shade



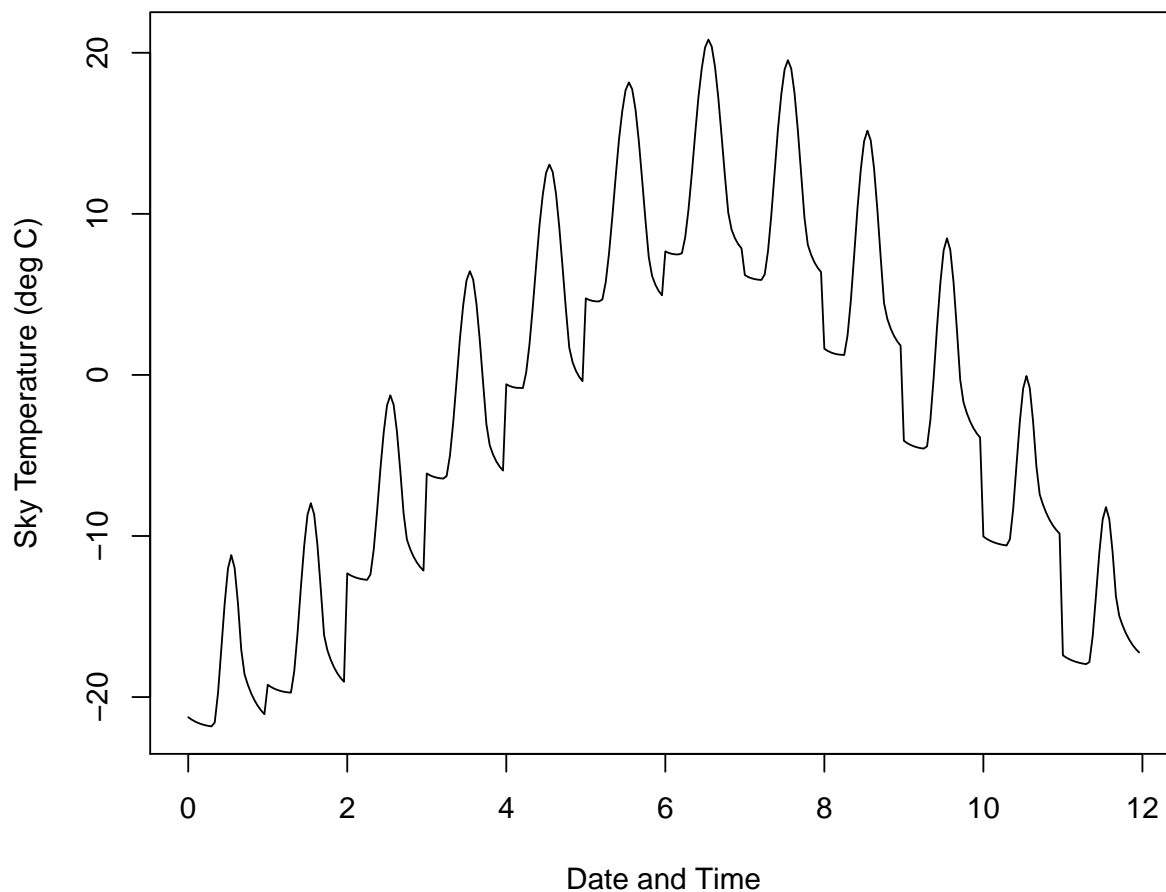
```
with(plotshadmet,{plot(RHLOC ~ dates,xlab = "Date and Time", ylab = "Relative Humidity (%)", type = "l",ylim=c(0,100),main=paste("humidity, ",maxshade,"% shade",sep=""))})  
with(plotshadmet,{points(RH ~ dates,xlab = "Date and Time", ylab = "Relative Humidity (%)", type = "l",col='blue',lty=2,ylim=c(0,100))})
```

humidity, 90% shade



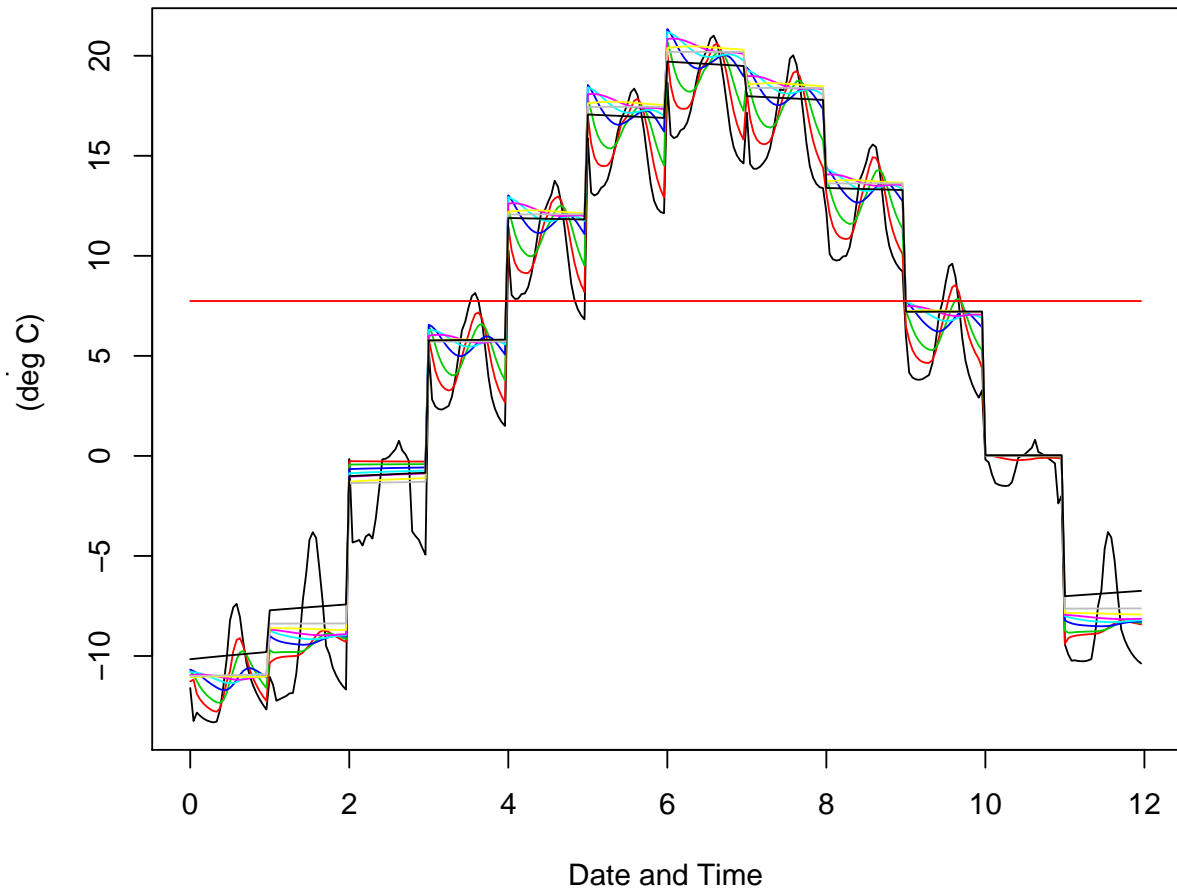
```
with(plotshadmet,{plot(TSKYC ~ dates,xlab = "Date and Time", ylab = "Sky Temperature (deg C)",  
type = "l",main=paste("sky temperature, ",maxshade,"% shade",sep=""))})
```

sky temperature, 90% shade



```
# plotting soil temperature for maximum shade
for(i in 1:10){
  if(i==1){
    plot(plotshadsoil[,i+3]~plotshadsoil[,1],xlab = "Date and Time", ylab = "Soil Temperature
(deg C)",col=i,type = "l",main=paste("soil temperature ",maxshade,"% shade",sep=""))
  }else{
    points(plotshadsoil[,i+3]~plotshadsoil[,1],xlab = "Date and Time", ylab = "Soil Temperature
(deg C)",col=i,type = "l")
  }
}
```

soil temperature 90% shade



Appendix 4 Microclimate model hourly input example

Michael R. Kearney

2016-04-18

Contents

Introduction

This vignette provides an example of how to run the model using hourly input weather data from the Soil and Climate Network (SCAN) for the USA, working with data for Ford Dry Lake in California for 2015. This dataset is provided in the package as `Scan_FordDryLake_2015` and has been pre-processed to a small degree as described in the help for this dataset. You can take the R code in this vignette and adapt it to work with other datasets you might have collected yourself or have obtained from elsewhere. Note that there is also an experimental function, `micro_SCAN`, to run the model from the SCAN data in a more automated fashion.

Getting the site data

First load the NicheMapR package and also the zoo package, from which we'll use the `na.approx` function to fill in missing data.

```
library(NicheMapR)
library(zoo)
```

Also included in this package is a table of summary information on all the SCAN sites, `SCANsites`. Have a look at the first 5 by using the 'head' function:

```
head(SCANsites)
```

```
##      network id state          name      lat      lon elev
## 1 SCAN      15  PR MARICAO FOREST  18.15000 -67.00468 2450
## 2 SCAN     581  MT LINDSAY          47.21415 -105.18702 2871
## 3 SCAN     674  ID ORCHARD RANGE SITE  43.32268 -115.99643 3200
## 4 SCAN     750  NV SHELDON          41.90450 -119.44360 5860
## 5 SCAN     808  MT TABLE MOUNTAIN  45.80272 -111.58655 4474
## 6 SCAN     965  AK IKALUKROK CREEK  68.08333 -163.00000  650
##           county NRCS Soil Survey Pedon Code GMT offset
## 1           MAYAGUEZ                28522      -4
## 2            DAWSON                33997      -7
## 3              ADA                 NULL      -8
## 4            WASHOE                16312      -8
## 5            GALLATIN               NULL      -7
## 6 NORTHWEST ARCTIC                <NA>      -9
##           start date
## 1 2001-05-08 00:00:00
## 2 2006-10-22 06:59:59
## 3 2004-03-19 12:59:59
## 4 1996-09-17 14:00:00
## 5 2006-10-25 15:00:00
## 6 2006-10-01 00:00:00
```

Now we'll choose the Ford Dry Lake site and put some of the summary data that we need into variables:

```
sitenum='2184' # Ford Dry Lake
site=subset(SCANsites,id==sitenum) # subset the SCANsites dataset for Ford Dry Lake
name=site$name # the name of the sites
Latitude<-site$lat # the latitude in decimal degrees
Longitude<-site$lon # the longitude in decimal degrees
Elevation<-site$elev/3.28084 # elevation, converted from feet to metres
TZoffset<-site$`GMT offset` # the offset from Greenwich Mean Time, in hours
ystart=2015 # start year
yfinish=2015 # end year
nyears=yfinish-ystart+1 # number of years to run
```

The SCAN data for Ford Dry Lake is included as a preloaded datatable too, `SCAN_FordDryLake_2015`. We will now create a new variable called `weather` based on this dataset:

```
weather<-SCAN_FordDryLake_2015 # make SCAN_FordDryLake_2015 supplied package data
# the weather input variable
```

Setting the model modes

These parameters control how the model runs. We will run the model for both sun and shade, with the soil moisture and snow options on, and also note that the hourly parameter is also on so that it uses the hourly SCAN weather data as input

```
writescv<-0 # make Fortran code write output as csv files
runshade<-1 # run the model twice, once for each shade level (1) or just for
# the first shade level (0)?
runmoist<-1 # run soil moisture model (0=no, 1=yes)?
snowmodel<-1 # run the snow model (0=no, 1=yes)? - note that this runs slower
hourly<-1 # run the model with hourly input data
microdaily<-1 # run microclimate model where one iteration of each day occurs
# and last day gives initial conditions for present day
```

Setting the times and location info

These parameters relate to the geographic location and the time period over which the model will run

```
longlat<-c(Longitude,Latitude) # decimal degrees longitude and latitude from
# the SCAN site data table
julnum<-floor(nrow(weather)/24) # number of days to run, determined by counting
# the number of rows in the weather dataset and dividing by 24 to get days,
# but keeping it as a whole number
idayst <- 1 # start month
ida<-julnum # end month
HEMIS <- ifelse(longlat[2]<0,2.,1.) # chose hemisphere based on latitude
ALAT <- abs(trunc(longlat[2])) # degrees latitude
AMINUT <- (abs(longlat[2])-ALAT)*60 # minutes latitude
ALONG <- abs(trunc(longlat[1])) # degrees longitude
ALMINT <- (abs(longlat[1])-ALONG)*60 # minutes longitude
ALREF <- ALONG # reference longitude for time zone
```

Time-independent microclimate model parameters

Now we set the non-temporal parameters including the depths we want to simulate (these have been matched in part to the depths at which the SCAN data are reported), terrain properties, and using the Global Aerosol Data Set (GADS) to get aerosol an aerosol attenuation profile. Note that the TIMAXS and TIMINS vectors which control how min/max weather data are interpolated to hourly profiles are specified but they will be redundant because the hourly option was set to 1 above.

```
EC <- 0.0167238 # Eccentricity of the earth's orbit (current value 0.0167238,
# ranges between 0.0034 to 0.058)
RUF <- 0.004 # Roughness height (m), , e.g. sand is 0.05, grass may be 2.0,
# current allowed range: 0.001 (snow) - 2.0 cm.
# Next four parameters are segmented velocity profiles due to bushes, rocks
# etc. on the surface
#IF NO EXPERIMENTAL WIND PROFILE DATA SET ALL THESE TO ZERO! (then roughness
# height is based on the parameter RUF)
Z01 <- 0. # Top (1st) segment roughness height(m)
Z02 <- 0. # 2nd segment roughness height(m)
ZH1 <- 0. # Top of (1st) segment, height above surface(m)
ZH2 <- 0. # 2nd segment, height above surface(m)
SLE <- 0.96 # Substrate longwave IR emissivity (decimal %), typically close to 1
ERR <- 1.5 # Integrator error for soil temperature calculations
Refhyt <- 2 # Reference height (m), reference height at which air temperature,
# wind speed and relative humidity input data are measured
DEP <- c(0., 2.5, 5., 10., 15., 20., 30., 50., 100., 200.) # Soil nodes (cm)
# - keep spacing close near the surface, last value is where it is assumed that
# the soil temperature is at the annual mean air temperature
Thcond <- 2.5 # soil minerals thermal conductivity (W/mC)
Density <- 2560. # soil minerals density (kg/m3)
SpecHeat <- 870. # soil minerals specific heat (J/kg-K)
BulkDensity <- 1300 # soil bulk density (kg/m3)
SatWater <- 0.26 # volumetric water content at saturation (0.1 bar matric
# potential) (m3/m3)
Clay <- 20 # clay content for matric potential calculations (%)
REFL<-0.20 # soil reflectance (decimal %)
ALTT<-Elevation # altitude (m)
slope<-0. # slope (degrees, range 0-90)
azmuth<-180. # aspect (degrees, 0 = North, range 0-360)
hori<-rep(0,24) # enter the horizon angles (degrees) so that they go
# from 0 degrees azimuth (north) clockwise in 15 degree intervals
VIEWF <- 1-sum(sin(hori*pi/180))/length(hori) # convert horizon angles
# to radians and calc view factor(s)
PCTWET<-0 # percentage of surface area acting as a free water surface (%)
CMH2O <- 1. # precipitable cm H2O in air column, 0.1 = VERY DRY; 1.0 = MOIST
# AIR CONDITIONS; 2.0 = HUMID, TROPICAL CONDITIONS (note this is for the whole
# atmospheric profile, not just near the ground)
TIMAXS <- c(1.0, 1.0, 0.0, 0.0) # Time of Maximums for Air Wind RelHum Cloud (h),
# air & Wind max's relative to solar noon, humidity and cloud cover max's
# relative to sunrise
TIMINS <- c(0.0, 0.0, 1.0, 1.0) # Time of Minimums for Air Wind RelHum Cloud (h),
# air & Wind min's relative to sunrise, humidity and cloud cover min's relative
# to solar noon
minshade<-0. # minimum available shade (%)
maxshade<-90. # maximum available shade (%)
```

```

Usrhyt <- 0.01# local height (m) at which air temperature, relative humidity
# and wind speed calculations will be made
# Aerosol profile using GADS
relhum<-1.
optdep.summer<-as.data.frame(rungads(longlat[2],longlat[1],relhum,0))
optdep.winter<-as.data.frame(rungads(longlat[2],longlat[1],relhum,1))
optdep<-cbind(optdep.winter[,1],rowMeans(cbind(optdep.summer[,2],optdep.winter[,2])))
optdep<-as.data.frame(optdep)
colnames(optdep)<-c("LAMBDA","OPTDEPTH")
a<-lm(OPTDEPTH~poly(LAMBDA, 6, raw=TRUE),data=optdep)
LAMBDA<-c(290,295,300,305,310,315,320,330,340,350,360,370,380,390,400,420,440,460
,480,500,520,540,560,580,600,620,640,660,680,700,720,740,760,780,800,820,840,860
,880,900,920,940,960,980,1000,1020,1080,1100,1120,1140,1160,1180,1200,1220,1240
,1260,1280,1300,1320,1380,1400,1420,1440,1460,1480,1500,1540,1580,1600,1620,1640
,1660,1700,1720,1780,1800,1860,1900,1950,2000,2020,2050,2100,2120,2150,2200,2260
,2300,2320,2350,2380,2400,2420,2450,2490,2500,2600,2700,2800,2900,3000,3100,3200
,3300,3400,3500,3600,3700,3800,3900,4000)
TAI<-predict(a,data.frame(LAMBDA))

```

Time-independent microclimate model variables

Now we need to specify hourly air temperature, wind speed, relative humidity, solar radiation, precipitation and cloud cover. The first 5 come directly from the SCAN dataset, but this dataset doesn't include cloud cover so we will have to estimate it as described below.

To start with, we need to use the `na.approx` function from the `zoo` package to interpolate NA values.

```

# check if first element is NA and, if so, use next non-NA value for
# na.approx function
if(is.na(weather$TAVG.H[1])==TRUE){ # mean hourly air temperature
  weather$TAVG.H[1]<-weather$TAVG.H[which(!is.na(weather$TAVG.H))[1]]
}

if(is.na(weather$WSPDV.H[1])==TRUE){ # mean hourly wind speed
  weather$WSPDV.H[1]<-weather$WSPDV.H[which(!is.na(weather$WSPDV.H))[1]]
}

if(is.na(weather$RHUM[1])==TRUE){ # mean hourly relative humidity
  weather$RHUM[1]<-weather$RHUM[which(!is.na(weather$RHUM))[1]]
}

if(is.na(weather$SRADV.H[1])==TRUE){ # mean hourly solar radiation
  weather$SRADV.H[1]<-weather$SRADV.H[which(!is.na(weather$SRADV.H))[1]]
}

if(is.na(weather$PRCP.H[1])==TRUE){ # hourly precipitation
  weather$PRCP.H[1]<-weather$PRCP.H[which(!is.na(weather$PRCP.H))[1]]
}

# use na.approx function from zoo package to fill in missing data
TAIRhr<-weather$TAVG.H<-na.approx(weather$TAVG.H)
RHhr<-weather$RHUM.I<-na.approx(weather$RHUM.I)
SOLRhr<-weather$SRADV.H<-na.approx(weather$SRADV.H)
RAINhr<-weather$PRCP.H<-na.approx(weather$PRCP.H*25.4) # convert rainfall
# from inches to mm
WNhr<-weather$WSPDV.H<-na.approx(weather$WSPDV.H*0.44704) # convert wind
# speed from miles/hour to m/s

```

Now we run the microclimate model using the `micro_global` function with the option `clearsky` set to 1 to obtain 365 days of clear sky solar radiation which we can then use in comparison to the observed solar radiation to infer cloud cover.

```
# run global microclimate model in clear sky mode to get clear sky radiation
micro<-micro_global(loc=c(Longitude,Latitude),timeinterval = 365, clearsky = 1)

## extracting climate data

## Loading required namespace: ncdf4

## extracting soil moisture data
## running microclimate model for 365 days by 1 years at site long -115.09763 lat 33.6547
## user system elapsed
## 2.12 0.00 2.12

# append dates
tzone=paste("Etc/GMT",TZoffset,sep="") # doing it this way ignores
# daylight savings!
dates=seq(ISOdate(ystart,1,1,tz=tzone)-3600*12,
ISOdate((ystart+nyears),1,1,tz=tzone)-3600*13, by="hours")
clear<-as.data.frame(cbind(dates,as.data.frame(rep(micro$metout[1:(365*24)],13),
nyears))),stringsAsFactors = FALSE)
julday<-rep(seq(1,365),nyears)[1:floor(nrow(weather)/24)] # julian days to run
clear=as.data.frame(clear,stringsAsFactors = FALSE)
colnames(clear)=c("datetime","sol")

# find the maximum observed solar and adjust the clear sky prediction to this value
maxsol=max(SOLRhr)
clear2<-clear[,2]*(maxsol/max(clear[,2])) # get ratio of max observed to predicted
# max clear sky solar

# compute cloud cover from ratio of max to observed solar
sol=SOLRhr
sol[sol<5]<-0 # remove very low values
clr<-clear2
clr[clr<5]<-0 # remove very low values
a=((clr-sol)/clr)*100 # get ratio of observed to predicted solar, convert to %
a[a>100]<-100 # cap max 100%
a[a<0]<-0 # cap min at 0%
a[is.na(a)]=0 # replace NA with zero
a[is.infinite(a)]=0 # replace infinity with zero
a[a==0]=NA # change all zeros to NA for na.approx
a=na.approx(a,na.rm = FALSE) # apply na.approx, but leave any trailing NAs
a[is.na(a)]=0 # make trailing NAs zero
CLDhr<-a # now we have hourly cloud cover
```

We still need to give the model vectors of daily minimum and maximum weather data, even though they are not used when `hourly` is set to 1, so the next code chunk summarises the minima and maxima from the hourly data. If you set `hourly` to zero, you can see the difference having hourly data makes to the fit of the model to the observed SCAN data on soil temperature and soil moisture.

```

# aggregate hourly data to daily min/max
CCMAXX<-aggregate(CLDhr,by=list(weather$Date), FUN=max)[,2] # max cloud cover (%)
CCMINN<-aggregate(CLDhr,by=list(weather$Date), FUN=min)[,2] #min cloud cover (%)
TMAXX<-aggregate(TAIRhr,by=list(weather$Date), FUN=max)[,2] # maximum air temperatures
# (deg C)
TMINN<-aggregate(TAIRhr,by=list(weather$Date), FUN=min)[,2] # minimum air temperatures
# (deg C)
RAINFALL<-aggregate(RAINhr,by=list(weather$Date), FUN=sum)[,2] # minimum air temperatures
# (deg C)
RHMAXX<-aggregate(RHhr,by=list(weather$Date), FUN=max)[,2] # max relative humidity (%)
RHMINN<-aggregate(RHhr,by=list(weather$Date), FUN=min)[,2] # min relative humidity (%)
WNMAXX<-aggregate(WNhr,by=list(weather$Date), FUN=max)[,2] # max wind speed (m/s)
WNMINN<-aggregate(WNhr,by=list(weather$Date), FUN=min)[,2] # min wind speed (m/s)

```

Finally, we need the annual mean temperature and the running mean annual temperature for use as a boundary deep soil condition, as well as daily values of maximum and minimum shade, substrate emissivity and solar reflectance, and surface wetness, which we will keep constant across all days in this simulation. Also, for the deep soil boundary condition, we only have one year of data so we cannot compute a running 365 day mean of the air temperature and instead we simply use a constant mean annual temperature.

```

tannul<-mean(c(TMAXX, TMINN)) # annual mean temperature for getting monthly deep soil
# temperature (deg C)
tannulrun<-rep(tannul,julnum) # monthly deep soil temperature (2m) (deg C)
# creating the arrays of environmental variables that are assumed not to change
# with month for this simulation
MAXSHADES <- rep(maxshade,julnum) # daily max shade (%)
MINSHADES <- rep(minshade,julnum) # daily min shade (%)
SLES <- rep(SLE,julnum) # set up vector of ground emissivities for each day
REFLS<-rep(REFL,julnum) # set up vector of soil reflectances for each day
PCTWET<-rep(PCTWET,julnum) # set up vector of soil wetness for each day

```

Soil thermal properties

Next we need to specify the soil properties. This code sets up for one soil type in terms of thermal properties, but below we will make the soil moisture-related properties transition at a certain depth.

```

# set up a profile of soil properties with depth for each day to be run
Numtyps <- 1 # number of soil types
Nodes <- matrix(data = 0, nrow = 10, ncol = julnum) # array of all possible
# soil nodes for max time span of 20 years
Nodes[1,1:julnum]<-10 # deepest node for first substrate type
Density<-Density/1000 # density of minerals - convert to Mg/m3
BulkDensity<-BulkDensity/1000 # density of minerals - convert to Mg/m3

# now make the soil properties matrix
# columns are:
#1) bulk density (Mg/m3)
#2) volumetric water content at saturation (0.1 bar matric potential) (m3/m3)
#3) clay content (%)
#4) thermal conductivity (W/mK)
#5) specific heat capacity (J/kg-K)
#6) mineral density (Mg/m3)

```

```

soilprops<-matrix(data = 0, nrow = 10, ncol = 6) # create an empty soil
# properties matrix
soilprops[1,1]<-BulkDensity # insert soil bulk density to profile 1
soilprops[1,2]<-SatWater # insert saturated water content to profile 1
soilprops[1,3]<-Clay # insert percent clay to profile 1
soilprops[1,4]<-Thcond # insert thermal conductivity to profile 1
soilprops[1,5]<-SpecHeat # insert specific heat to profile 1
soilprops[1,6]<-Density # insert mineral density to profile 1
soilinit<-rep(tannul,20) # make initial soil temps equal to mean annual

```

Soil moisture properties

Now we specify the soil moisture-related parameters using Table 9.1 out of Campbell and Norman's 1990 book 'Environmental Biophysics'. Note that there are 19 total nodes because an extra node has been inserted between each of the 10 depths specified in the DEP array to improve the accuracy of the soil moisture calculations. First all 19 nodes are given the values for soil type 3, a sandy loam, and then the deeper nodes (greater than 15 cm) are overwritten with soil type 5 which is a silt loam. Also specified is the root density profile, the leaf area index (both default values based on Campbell's 1985 book 'Soil Physics with Basic'), and some other parameters that control how the rainfall is applied together with the initial soil moisture values.

```

#use Campbell and Norman Table 9.1 soil moisture properties
soiltype=3 # 3 = sandy loam
PE<-rep(CampNormTb19_1[soiltype,4],19) #air entry potential J/kg
KS<-rep(CampNormTb19_1[soiltype,6],19) #saturated conductivity, kg s/m3
BB<-rep(CampNormTb19_1[soiltype,5],19) #soil 'b' parameter
BD<-rep(BulkDensity,19) # soil bulk density, Mg/m3
soiltype=5 # change deeper nodes to 5 = a silt loam
PE[10:19]<-CampNormTb19_1[soiltype,4] #air entry potential J/kg
KS[10:19]<-CampNormTb19_1[soiltype,6] #saturated conductivity, kg s/m3
BB[10:19]<-CampNormTb19_1[soiltype,5] #soil 'b' parameter

L<-c(0,0,8.18990859,7.991299442,7.796891252,7.420411664,7.059944542,6.385001059,
5.768074989,4.816673431,4.0121088,1.833554792,0.946862989,0.635260544,0.804575,
0.43525621,0.366052856,0,0)*10000 # root density at each node, mm/m3 (from Campbell
# 1985 Soil Physics with Basic, p. 131)
LAI<-0.1 # leaf area index, used to partition transpiration/evaporation from PET
rainmult<-1 # rainfall multiplier to impose catchment
maxpool<-10 # max depth for water pooling on the surface, mm (to account for runoff)
evenrain<-0 # spread daily rainfall evenly across 24hrs (1) or one event at
# midnight (2)
SoilMoist_Init<-rep(0.2,10) # initial soil water content for each node, m3/m3
moists<-matrix(nrow=10, ncol = julnum, data=0) # set up an empty vector for soil
# moisture values through time
moists[1:10,]<-SoilMoist_Init # insert initial soil moisture

```

Snow model parameters

We also need to specify the snow model parameters - these ones tend to work well in general and at the site being considered there is no snowfall so they will not be of consequence.

```

snowtemp<-1.5 # temperature at which precipitation falls as snow
# (used for snow model)
snowdens<-0.375 # snow density (mg/m3)
densfun<-c(0,0) # slope and intercept of linear model of snow density
# as a function of day of year - if it is c(0,0) then fixed density used
snowmelt<-0.9 # proportion of calculated snowmelt that doesn't refreeze
undercatch<-1. # undercatch multiplier for converting rainfall to snow
rainmelt<-0.0125 # parameter in equation from Anderson's SNOW-17 model
# that melts snow with rainfall as a function of air temp

```

Tide parameters

Finally, we need to give the model a vector of tide conditions although we are not running the model in intertidal mode so they also will be of no consequence.

```

# intertidal simulation input vector (col 1 = tide in(1)/out(0),
# col 2 = sea water temperature in deg C, col 3 = % wet from wave splash)
tides<-matrix(data = 0., nrow = 24*julnum, ncol = 3) # matrix for tides

```

Running the model

The data inputs are all ready now and they need to be sent to the Fortran program. The single-value inputs are collected in on long vector called `microinput` and then this, together with the longer inputs, are then sent put into a list called `microin` and passed to the function `microclimate` which runs the model. We will return the results to a list object called `micro`.

```

# microclimate input parameters list
microinput<-c(julnum,RUF,ERR,Usrhyt,Refhyt,Numtyps,Z01,Z02,ZH1,ZH2,idayst,ida,
HEMIS,ALAT,AMINUT,ALONG,ALMINT,ALREF,slope,azmuth,ALTT,CMH20,microdaily,tannul,
EC,VIEWF,snowtemp,snowdens,snowmelt,undercatch,rainmult,runshade,runmoist,
maxpool,evenrain,snowmodel,rainmelt,writescv,densfun,hourly)

# all microclimate data input list - all these variables are expected
# by the input argument of the fortran micro2014 subroutine
microin<-list(microinput=microinput,tides=tides,julday=julday,SLES=SLES,DEP=DEP,
Nodes=Nodes,MAXSHADES=MAXSHADES,MINSHADES=MINSHADES,TIMAXS=TIMAXS,TIMINS=TIMINS,
TMAXX=TMAXX,TMINN=TMINN,RHMAXX=RHMAXX,RHMINN=RHMINN,CCMAXX=CCMAXX,CCMINN=CCMINN,
WNMAXX=WNMAXX,WNMINN=WNMINN,TAIRhr=TAIRhr,RHhr=RHhr,WNhr=WNhr,CLDhr=CLDhr,
SOLRhr=SOLRhr,RAINhr=RAINhr,REFLS=REFLS,PCTWET=PCTWET,soilinit=soilinit,horihori,
TAI=TAI,soilprops=soilprops,moists=moists,RAINFALL=RAINFALL,tannulrun=tannulrun,
PE=PE,KS=KS,BB=BB,BD=BD,L=L,LAI=LAI)

micro<-microclimate(microin) # run the model in Fortran

```

Retrieving the output and plotting the results against observed values

Now the model has run and we need to retrieve the output from the `micro` list and add the date/time vector to them from the original SCAN dataset.

```

# retrieve ouput
dates=weather$datetime[1:nrow(micro$metout)]
metout<-as.data.frame(micro$metout[1:(julnum*24),]) # retrieve above ground
# microclimatic conditions, min shade
shadmet<-as.data.frame(micro$shadmet[1:(julnum*24),]) # retrieve above ground
# microclimatic conditions, max shade
soil<-as.data.frame(micro$soil[1:(julnum*24),]) # retrieve soil temperatures,
# minimum shade
shadsoil<-as.data.frame(micro$shadsoil[1:(julnum*24),]) # retrieve soil temperatures,
# maximum shade
soilmoist<-as.data.frame(micro$soilmoist[1:(julnum*24),]) # retrieve soil moisture,
# minimum shade
shadmoist<-as.data.frame(micro$shadmoist[1:(julnum*24),]) # retrieve soil moisture,
# maximum shade
humid<-as.data.frame(micro$humid[1:(julnum*24),]) # retrieve soil humidity, minimum
# shade
shadhumid<-as.data.frame(micro$shadhumid[1:(julnum*24),]) # retrieve soil humidity,
# maximum shade
soilpot<-as.data.frame(micro$soilpot[1:(julnum*24),]) # retrieve soil water potential,
# minimum shade
shadpot<-as.data.frame(micro$shadpot[1:(julnum*24),]) # retrieve soil water potential,
# maximum shade

# append dates
metout<-cbind(dates,metout)
shadmet<-cbind(dates,shadmet)
soil<-cbind(dates,soil)
shadsoil<-cbind(dates,shadsoil)
soilmoist<-cbind(dates,soilmoist)
shadmoist<-cbind(dates,shadmoist)
humid<-cbind(dates,humid)
shadhumid<-cbind(dates,shadhumid)
soilpot<-cbind(dates,soilpot)
shadpot<-cbind(dates,shadpot)

```

Finally, we can specify a time window by setting the variables `tstart` and `tfinish` and then make two composite plots each showing the predictions and observations for the 5 depths, for soil temperature and soil moisture, respectively.

```

# choose a time window to plot
tstart=as.POSIXct("2015-05-01",format="%Y-%m-%d")
tfinish=as.POSIXct("2015-07-31",format="%Y-%m-%d")

# set up plot parameters
par(mfrow = c(5,1)) # set up for 6 plots in 1 columns
par(oma = c(2,1,2,2) + 0.1) # margin spacing stuff
par(mar = c(3,3,1,1) + 0.1) # margin spacing stuff
par(mgp = c(2,1,0) ) # margin spacing stuff

# plot the soil temperatures
plot(dates,soil$D5cm,type='l',ylim=c(-10,70),xlim=c(tstart,tfinish),
     xaxt = "n",ylab=expression("soil temperature (" * degree * C *)" ),xlab="")
points(weather$datetime,weather$ST0.I_2,type='l',col="red")

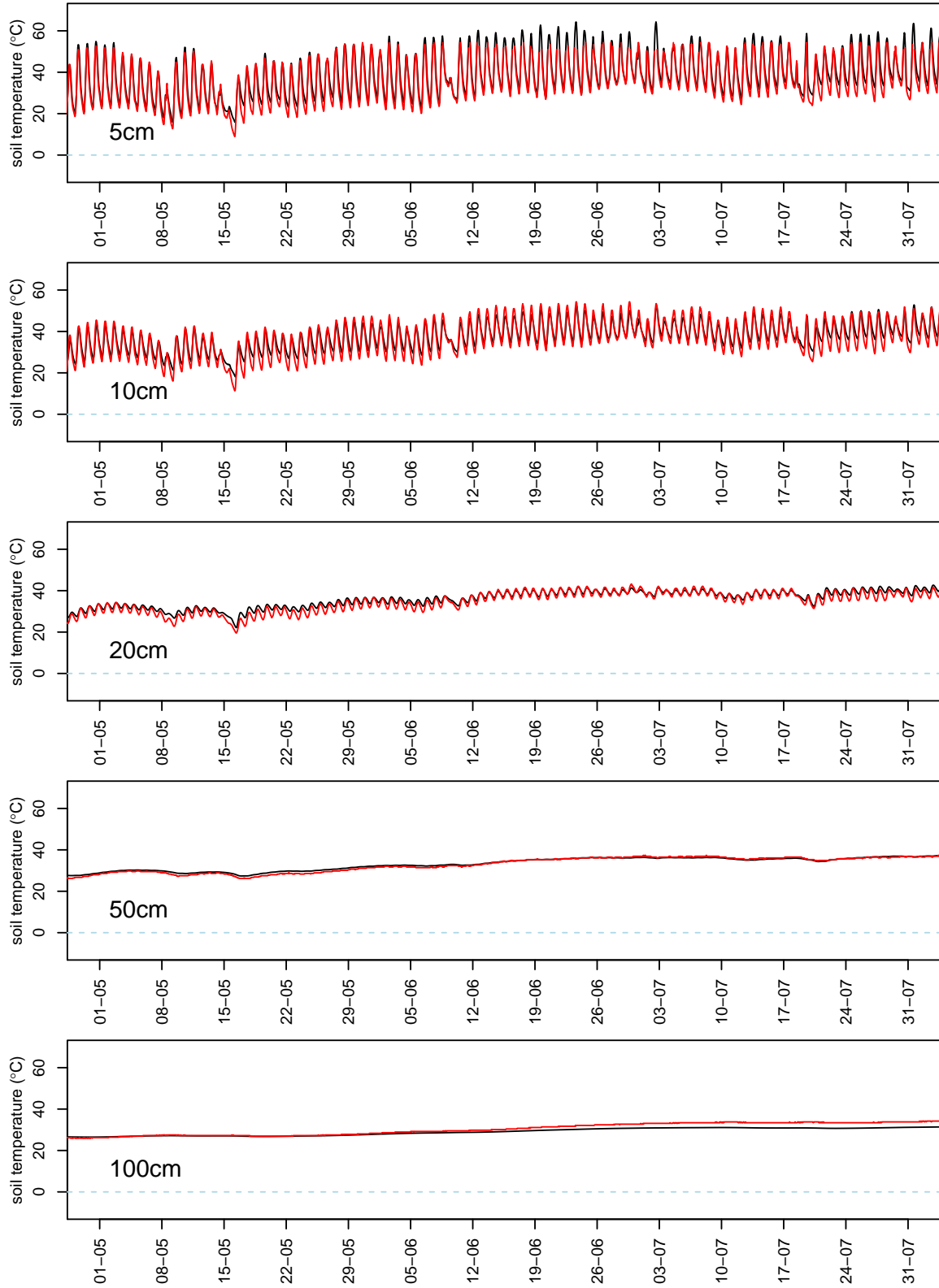
```

```

axis.POSIXct(side = 1, x = micro_shd$dates, at = seq(tstart,tfinish, "weeks"),
  format = "%d-%m", las = 2)
text(tstart,10,"5cm",col="black",pos=4,cex=1.5)
abline(0,0,lty=2,col='light blue')
#points(dates,metout$SNOWDEP,type='h',col='light blue')
plot(dates,soil$D10cm,type='l',ylim=c(-10,70),xlim=c(tstart,tfinish),xaxt = "n",
  ylab=expression("soil temperature (" * degree * C *)"),xlab="")
points(weather$datetime,weather$STO.I_4,type='l',col="red")
axis.POSIXct(side = 1, x = micro_shd$dates, at = seq(tstart,tfinish, "weeks"),
  format = "%d-%m", las = 2)
text(tstart,10,"10cm",col="black",pos=4,cex=1.5)
abline(0,0,lty=2,col='light blue')
plot(dates,soil$D20cm,type='l',ylim=c(-10,70),xlim=c(tstart,tfinish),xaxt = "n",
  ylab=expression("soil temperature (" * degree * C *)"),xlab="")
points(weather$datetime,weather$STO.I_8,type='l',col="red")
axis.POSIXct(side = 1, x = micro_shd$dates, at = seq(tstart,tfinish, "weeks"),
  format = "%d-%m", las = 2)
text(tstart,10,"20cm",col="black",pos=4,cex=1.5)
abline(0,0,lty=2,col='light blue')
plot(dates,soil$D50cm,type='l',ylim=c(-10,70),xlim=c(tstart,tfinish),xaxt = "n",
  ylab=expression("soil temperature (" * degree * C *)"),xlab="")
points(weather$datetime,weather$STO.I_20,type='l',col="red")
axis.POSIXct(side = 1, x = micro_shd$dates, at = seq(tstart,tfinish, "weeks"),
  format = "%d-%m", las = 2)
text(tstart,10,"50cm",col="black",pos=4,cex=1.5)
abline(0,0,lty=2,col='light blue')
plot(dates,soil$D100cm,type='l',ylim=c(-10,70),xlim=c(tstart,tfinish),xaxt = "n",
  ylab=expression("soil temperature (" * degree * C *)"),xlab="")
points(weather$datetime,weather$STO.I_40,type='l',col="red")
axis.POSIXct(side = 1, x = micro_shd$dates, at = seq(tstart,tfinish, "weeks"),
  format = "%d-%m", las = 2)
abline(0,0,lty=2,col='light blue')
text(tstart,10,"100cm",col="black",pos=4,cex=1.5)
mtext(site$name,outer = TRUE)

```

FORD DRY LAKE



```

# plot the soil moisture
plot(dates,soilmoist$WC5cm*100,type='l',ylim=c(0,60),xaxt = "n",
     xlim=c(tstart,tfinish),col="blue",ylab="soil moisture (%)",xlab="")
axis.POSIXct(side = 1, x = micro_shd$dates, at = seq(tstart,tfinish, "weeks"),
  format = "%d-%m", las = 2)
points(weather$datetime,weather$SMS.I_2,type='l',col="red")
text(tstart,40,"5cm",col="black",pos=4,cex=1.5)
plot(dates,soilmoist$WC10cm*100,type='l',ylim=c(0,60),xaxt = "n",xlim=c(tstart,
  tfinish),col="blue",ylab="soil moisture (%)",xlab="")
axis.POSIXct(side = 1, x = micro_shd$dates, at = seq(tstart,tfinish, "weeks"),
  format = "%d-%m", las = 2)
points(weather$datetime,weather$SMS.I_4,type='l',col="red")
text(tstart,40,"10cm",col="black",pos=4,cex=1.5)
plot(dates,soilmoist$WC20cm*100,type='l',ylim=c(0,60),xaxt = "n",xlim=c(tstart,
  tfinish),col="blue",ylab="soil moisture (%)",xlab="")
axis.POSIXct(side = 1, x = micro_shd$dates, at = seq(tstart,tfinish, "weeks"),
  format = "%d-%m", las = 2)
points(weather$datetime,weather$SMS.I_8,type='l',col="red")
text(tstart,40,"20cm",col="black",pos=4,cex=1.5)
plot(dates,soilmoist$WC50cm*100,type='l',ylim=c(0,60),xaxt = "n",xlim=c(tstart,
  tfinish),col="blue",ylab="soil moisture (%)",xlab="")
axis.POSIXct(side = 1, x = micro_shd$dates, at = seq(tstart,tfinish, "weeks"),
  format = "%d-%m", las = 2)
points(weather$datetime,weather$SMS.I_20,type='l',col="red")
text(tstart,40,"50cm",col="black",pos=4,cex=1.5)
plot(dates,soilmoist$WC100cm*100,type='l',ylim=c(0,100),xaxt = "n",xlim=c(tstart,tfinish),col="blue",yl
axis.POSIXct(side = 1, x = micro_shd$dates, at = seq(tstart,tfinish, "weeks"),
  format = "%d-%m", las = 2)
points(weather$datetime,weather$SMS.I_40,type='l',col="red")
text(tstart,40,"100cm",col="black",pos=4,cex=1.5)
mtext(site$name,outer = TRUE)

```

FORD DRY LAKE

