NMRF/TR/05/2016

TECHNICAL REPORT

Documentation of the Land Surface Scheme in NCMRWF Unified Model

C.K. Unnikrishnan, Saji Mohandas and E.N. Rajagopal

June 2016

National Centre for Medium Range Weather Forecasting
Ministry of Earth Sciences, Government of India
A-50, Sector-62, NOIDA-201309, INDIA

Documentation of the Land Surface Scheme in NCMRWF Unified Model

C. K. Unnikrishnan, Saji Mohandas and E. N. Rajagopal

June 2016

National Centre for Medium Range Weather Forecasting
Ministry of Earth Sciences
A-50, Sector 62, NOIDA-201 309, India

Document Control Data Sheet		
1	Name of the Institute	National Centre for Medium Range Weather Forecasting (NCMRWF)
2	Document Number	NMRF/TR/05/2016
3	Date of publication	June 2016
4	Title of the document	Documentation of the Land Surface Scheme in NCMRWF Unified Model
5	Type of Document	Technical Report
6	No. of Pages, Figures and Tables	69 Pages, 9 Figures and 10 Tables
7	Number of References	7
8	Author (S)	C.K. Unnikrishnan, Saji Mohandas and E.N. Rajagopal
9	Originating Unit	National Centre for Medium Range Weather Forecasting (NCMRWF), A-50, Sector-62, NOIDA, Uttar Pradesh
10	Abstract (100 words)	A Technical description of the land surface scheme (JULES version 3.3) used in operational NCMRWF Unified Model (NCUM) at NCMRWF is presented in this report. The coupling, input fields and formulation of the JULES model in the present operational forecast system are provided. The future scope and requirements are also discussed.
11	Security classification	Non-secure
12	Distribution	Unrestricted distribution
13	Key Words	Land surface, JULES, UM

Abstract

A technical description of the land surface scheme (JULES version 3.3) used in the NCMRWF Unified Model (NCUM) at NCMRWF is presented in this report. The coupling, input fields and formulation of the JULES model in the present operational forecast system are provided. The future scope and requirements are also discussed.

Contents

1. Land surface scheme in NCMRWF Unified Model

1.1 Introduction

Land surface is the lower boundary for the atmospheric models, which transfer mass, energy and momentum to the atmosphere. JULES (Joint UK Land Environment Simulator) land surface model (Best et al., 2011; Clark et al., 2011) simulates the land surface processes and exchanges the mass, energy and momentum to the atmospheric model (NCUM). The JULES model is forced by the atmospheric parameters like surface wind speed, specific humidity, surface air temperature, precipitation and short wave and long wave radiation from atmospheric model. This interactive coupling influences the evolution of state of the atmosphere and land in NCUM. The exchange from the surface can directly impact the boundary layer scheme and can impact other schemes like convection by influencing the stability of the lower atmosphere. Further, it impacts through the feedback mechanisms with the interaction of other schemes in the model, which may be more visible in extended and seasonal prediction. A proper understanding of the structure, formulation, input data set requirements and documentation of the land surface scheme is required for the fine tuning and developments towards the improvement of the land surface scheme coupling in model to achieve a better forecast and further helpful for the validation and scientific analysis.

1.2 Land surface schemes in Unified Model

Met Office Surface Exchange Scheme (MOSES) was developed during the 1990s for use in global and regional models at UK Met Office. Description MOSES-1 can be found in Cox et al. (1999), which calculate single energy balance for each grid box. However, the next version known as MOSES-2 (Essery, 2001) explicitly calculates the surface energy balance for different heterogeneous surface tiles in the grid box. The present land surface scheme JULES originated from the previous MOSES (Cox et al., 1999; Essery et al., 2003). Detailed scientific formulation of JULES on its energy and water fluxes are provided by Best et al. (2011) and the carbon and vegetation dynamics part is documented by Clark et al. (2011).

In this report, section 2 describes structure and data requirements of the land surface scheme in NCUM. Section 3 briefly explains the building processes of the JULES land surface model code and the calling sequence of its subroutines. Section 3 also describes the input and output variables of JULES. Section 4 summarizes the report and discusses future directions. It also describes the operational data requirements for a better land surface data assimilation and the formulation of a high resolution data assimilation system strategy.

2. Description of JULES land surface scheme

The formulation of the JULES land surface scheme used in UM is discussed here. The basic components of the model are described in the following subsections, which include radiation, surface energy budget, hydrology and thermodynamics.

2.1 Radiation

Tile method is used to represent the surface heterogeneity. Each grid box is divided into fractions of different surface types with independent solutions of the surface energy budget. An area weighted average of the energy fluxes is computed. JULES model accounts for the sub grid scale heterogeneity in each grid box as a mixture of the five vegetation types and four non vegetated surface types.

The radiation forcing from the atmospheric model is processed further to obtain the net radiation partitioned on tiles. Net shortwave radiation and downward long wave radiation are calculated by the radiation scheme of UM. Surface albedos are specified either as single values for all bands with diagnosed snow albedos. Albedo values are assigned to tiles. The snow-free and cold deep snow albedos for non-vegetated tiles are given in Table 1. Bare soil albedo depends on the type of soil and shows more spatial variations.

For vegetation with a leaf area index Λ, the snow-free (α_0) and cold deep snow (α_{cds}) albedos are calculated as follows

$$\alpha_0 = (1 - fr)\alpha_{soil} + fr\alpha_0 = (1 - fr)\,\alpha_{soil} + fr\,\alpha^{\infty}{}_{0} \tag{1}$$

$$\alpha_{cds} = (1 - fr)\alpha^{0}{}_{s} + fr\,\alpha^{\infty}{}_{s} \tag{2}$$

Where α_{soil} is soil albedo and *fr* is radiative fraction.

$$fr = 1 - e^{-\Lambda/2}$$

The vegetation dependent parameters of $\alpha^{\infty}{}_{0}$, $\alpha^{0}{}_{s}$ and $\alpha^{\infty}{}_{s}$ values are provided in Table 2. The aging of snow is included by reducing the snow albedo when surface temperature T exceeds a limit of -2° C and T_m is melting point then α_s .

$$\alpha_s = \begin{cases} \alpha_{cds} \text{ when } T < T_m - 2 \\ \alpha_{cds} + 0.3(\alpha_0 - \alpha_{cds})(T - T_m + 2) \text{ when } T_m - 2 < T < Tm \end{cases} \tag{3}$$

For a tile with snow mass *S* (kg m^{-2}), the albedo is a weighted average

$$\alpha = \alpha_0 + (\alpha_s - \alpha_0)(1 - e^{-0.2S}) \quad (4)$$

There is a spectral albedo scheme based on two stream canopy radiation model. Separate direct-beam and diffuse albedos in visible and near-infrared wave bands are calculated for each vegetation type as

$$\alpha_{dir} = h1 + h2 + h3$$
$$\alpha_{dif} = h7 + h8$$

Where,

$$h1 = -dp4 - cf$$

$$h_3 = \frac{-1}{D_1}\left[\left(d - \frac{p_3 h_1}{\sigma}\right)(u_1 + h)\frac{1}{S_1} - p_1 S_2 \left(d - c - \frac{h_1}{\sigma}(u_1 + K)\right)\right]$$

$$h_7 = \frac{c}{D_1 S_1}(u_1 - h)$$

$$h_8 = \frac{cS_1}{D_1}(u_1 + h)$$

$$\beta_0 = \frac{1+K}{wK} a_s$$

$$c = \frac{1}{3}(\alpha + \omega)$$

$$\beta = \frac{c}{\omega}$$

$$b = 1 - (1 - \beta)\, \omega,\ d = \omega K \beta_0 ,\ f = \omega K(1 - \beta_0)$$

$$h = (b2 - c2)1/2 ,\ \sigma = K\ 2 + c2 - b2$$

$$u_1 = b - \frac{c}{\alpha_{soil}}$$

$$S_1 = e^{-h\Lambda}$$

$$S_2 = e^{-K\Lambda}$$

$$p1 = b + h,\ p2 = b - h,\ p3 = b + K,\ p4 = b - K$$

$$D_1 = \frac{p_1}{S_1}(u_1 - h) - p_2 S_1 (u_1 + h)$$

The impact of the vegetation to radiation is included by using single scattering albedo and optical depth per unit leaf area, which are calculated based on the following equations

$$a_s = \frac{w}{2}\left[1 - \mu \ln\left(\frac{\mu+1}{\mu}\right)\right]$$

Where, μ is zenith angle cosine.

$$K = \frac{1}{2\mu}$$

Values of leaf reflection coefficient α and leaf scattering coefficient ω, depend on vegetation type and wave band, given in Table 3. Spectral snow albedo model available for aging of snow is characterized by introducing a prognostic grain size r(t), which is set to r0 = 50 μm for fresh snow and limited to a maximum value of 2000 μm. The change in r(t) over a timestep Δt is given by

$$r(t+\Delta t) = [r(t)^2 + G_r \frac{\Delta t}{\pi}]^{1/2} - [r(t) - r_0]\frac{S_f \Delta t}{d_o}$$

where, S_f is the snowfall rate during the timestep and do , the mass of fresh snow required to refresh the albedo, is set to 2.5 kg m^{-2}. The empirical grain area growth rate is

$$G_r = 0.6 \mu m^2 s^{-1} when T_0 = T_m (melting snow)$$

$$G_r = 0.06 \mu m^2 s^{-1} when T_0 < T_m ; r < 150 \mu m (cold fresh snow)$$

$$G_r = A \exp(E / RT_0) when T_0 > T_m ; r > 150 \mu m (cold aged snow)$$

where, A = 0.23 × 106 $\mu m^2 s^{-1}$, E = 37000 J mol^{-1} and R = 8.13451 J K^{-1} mol^{-1} .

Snow albedos (visible and near infrared) are calculated as

$$\alpha_{vis} = 0.98 - 0.002(r^{1/2} - r^{1/2}_0)$$

$$\alpha_{nir} = 0.7 - 0.07(r / r_0)$$

The zenith angle dependence is represented by using an effective grain size in place of r in calculations of direct-beam albedos. i.e.,

$$re = [1 + 0.77(\mu - 0.65)]^2 r$$

For a tile with snow-free albedo α_0 , snowdepth d and roughness length z_0 , the albedo in each band is

$$\alpha = f_{snow}\, \alpha_{snow} + (1 - f_{snow})\, \alpha_0$$

$$f_{snow} = \frac{d}{d + 10z_0}$$

Radiation diagnostics is done each timestep, for a grid box with tile fractions v_j, the grid box mean albedo for band *i* and the effective radiative surface temperature is given by

$$\alpha_i = \sum v_j \alpha_{ij}$$

$$T_{R0} = \left(\sum v_j T_{oj}^4\right)^{1/4}$$

The above are used in calculating downward shortwave and longwave radiation fluxes LW↓ and SW↓. Surface energy flux calculations require the net all-band shortwave radiation on each tile

$$SWN_j = i\,(1 - \alpha_{ij})SW{\downarrow}_i$$

$$\Delta OLR = OLR - \sigma T^4 R$$

which are used in diagnosing the adjustment in TOA outgoing longwave radiation OLR due to changes in surface temperature between radiation calls. SWN $_j$, LW↓ and ΔOLR are stored in the radiation increment array for use on timesteps between radiation calls.

	α_0	α_{cds}
Urban	0.18	0.8
Inland water	0.06	0.8
Soil	0.11-0.35	0.8
Ice	0.75	0.8

Table 1: Look-up table for Snow free and cold deep snow albedo for non vegetated surface types

	α_0^{∞}	α_s^{∞}	α_s^{0}
Broadleaf trees	0.1	0.15	0.3
Needleleaf trees	0.1	0.15	0.3
C3 grass	0.2	0.6	0.8
C4 grass	0.2	0.6	0.8
Shrubs	0.2	0.6	0.8

Table 2: Look-up table for albedo parameters for vegetation types

	α_{vis}	α_{nir}	ω_{vis}	ω_{nir}
Broad leaf trees	0.1	0.45	0.15	0.7
Needle leaf trees	0.07	0.35	0.15	0.45
C3 grass	0.1	0.58	0.15	0.83
C4 grass	0.1	0.58	0.17	0.83
Shrubs	0.1	0.58	0.15	0.83

Table 3: Look-up table for Spectral albedo parameters

2.2 Surface fluxes

Calculation of surface fluxes need surface exchange coefficients, which are further related to roughness length of the tile. Momentum roughness length z_0 is set to h/20 for trees of height h and h/10 for other vegetation types. Roughness lengths for non vegetated surface types are given in Table 4. The roughness length of a tile with snow mass S is reduced to max $[z_0 - 4 \times 10^{-4}$ S, $5 \times 10^{-4}]$. A surface exchange coefficient for sensible and latent heat fluxes between the surface and the lowest atmospheric level at height z_1 over each tile is calculated as CH = fh C_{Hn}, where C_{Hn} is neutral exchange coefficient.

$$C_{Hn} = k^2 \left[\ln \left(\frac{z_1 + z_0}{z_0} \right) n \left(\frac{z_1 + z_0}{z_{0h}} \right) \right]^{1}$$

(5)

fh = (1+ 10 R $_{iB}$/ Pr) -1 & R_{iB} >= 0 (stable)

fh = 1-10R_{iB}(1+10 Chn sqrt (-R_{iB}) / f_z) -1

Scalar roughness length is Z $_{oh}$ = Z$_0$/10

$$f_z = \frac{1}{4}\left(\frac{z_0}{z_1 + z_0}\right)^{1/2}$$

Prandtl number is calculated as following equation

$$Pr = \ln\left(\frac{z_0}{z_1 + z_0}\right)\left[\ln\left(\frac{z_0}{z_1 + z_{0h}}\right)\right]^{-1}$$

The bulk Richardson number for level-1 temperature T1, specific humidity q1 and wind speed U1, qsat (T , p) is the saturation humidity at the surface temperature and pressure, and the surface resistance factor ψ is given below

$$R_{iB} = \frac{g z_1}{U_1^2}\left(\frac{1}{T_1}\left[T_1 - T_0 + \frac{g}{c_p}\left(z_1 + z_{om} - z_{oh}\right)\right] + \Psi \frac{q_1 - q_{sat}\left(T_o, p_o\right)}{q_1 + \frac{\epsilon}{(1-\epsilon)}}\right) \tag{6}$$

The details of ψ are given in hydrology section. Since ψ depends on CH, it calculate ψ by assuming neutral conditions and also assumes that there is no level-1 cloud and does not include orographic roughness.

	Z$_0$ (m)
Urban	1.5
Water	3 x 10 -4
Soil	3 x 10 -4
Ice	1 x 10 -4

Table 4: Look-up table for roughness length for non vegetated surface types

Canopy heat capacity is calculated by a vegetation canopy model which also estimates the radiative coupling between the canopy and below ground. JULES also has a global dynamical vegetation model named ‘Top-down Representation of Interactive Foliage and Flora Including Dynamics’ (TRIFFID). TRIFFID is switched off in Unified model at NCMRWF. TRIFFID dynamic vegetation model gives the masses of carbon in leaves and stems per unit area of canopy as $\sigma_l \Lambda_b$ and $a_{wl} \Lambda_b^{5/3}$, where the balanced-growth leaf area index for vegetation (Λ_b) of height h is provided by

$$\Lambda_b = \left(\frac{a_{ws}\eta_{sl} h}{a_{wl}}\right)^{3/2} \tag{7}$$

with parameters given in Table 5. An areal canopy heat capacity, Cc , is calculated assuming specific heat capacities (in kJ K^{-1} per kg of carbon) of 570 for leaves and 110 for wood. Cc is set to zero for non-vegetated tiles and vegetated tiles if the canopy model is not selected.

	$\mathbf{a_{wl}}$	$\mathbf{a_{ws}}$	$\boldsymbol{\eta_{sl}}$	$\boldsymbol{\sigma_l}$
Broadleaf trees	0.65	10	0.01	0.0375
Needle leaf	0.65	10	0.01	0.1
C3 grass	0.005	1	0.01	0.025
C4 grass	0.005	1	0.01	0.05
Shurbs	0.1	10	0.01	0.05

Table 5: Look-up table for vegetation parameters

Surface evaporation is calculated from soil, canopy and snow moisture. Potential evaporation from saturated parts of the surfaces (lakes, wet vegetation canopies and snow) are calculated. Further it depends on aerodynamic resistance. Transpiration of vegetation is controlled by canopy conductance (*gc*), photosynthesis model is used to estimate its dependence on temperature, humidity deficit, incident radiation, soil moisture availability and vegetation type. Root density plays a major role in the calculation of moisture at each level in the soil, it is assumed to follow an exponential distribution with depth. The fraction of roots in soil layer k extending from depth z_{k-1} to z_k is given as

$$r_k = \frac{e^{-2z_{k-1}/d_r} - e^{-2z_k/d_r}}{1 - e^{-2z_t/d_r}} \tag{8}$$

where, d_r is the root depth for the vegetation type (Table 6) and z_t is the total depth of the soil model. For transpiration Et, the flux extracted from soil layer k is e_0E_t.

$$e_{k^0} = \frac{r_k \beta_k}{\sum r_k \beta_k}$$

$$\beta_k = \begin{cases} 1 & \theta_k >= \theta_w \\ (\theta k - \theta cw)/(\theta c - \theta w) & \theta_w < \theta_k < \theta_c \\ 0 & \theta_k >= \theta_w \end{cases}$$

β_k is a soil moisture availability factor for a soil layer with unfrozen volumetric soil moisture concentration θ_k , critical point θ_c and wilting point θ_w. Baresoil evaporation is calculated using a conductivity equation which extracts moisture from soil layers and the conductivity is calculated as below both bare-soil tiles and fraction (1−fr) of vegetated tiles.

Total conductance is the sum of the soil and canopy conductances i.e,

$$gs = gc + (1 - fr)\, gsoil$$

Evapotranspiration fraction extracted from each soil layer is calculated for the surface layer and or lower layers as below

$$e_1 = \frac{g_c e_1^0 + (1 - f_r) g_{soil}}{gs}$$

The total evaporation from a tile is $E = \psi E_0$, where E_0 is the potential evaporation

$$\Psi = f_a + (1 - f_a) \frac{g_s}{g_s + C_H U_1}$$

and fa is the fraction of the tile which is saturated and hence has aerodynamic resistance only; fa= 1 for lake, ice or snow-covered tiles, and fa = C/Cm for a vegetated tile with canopy moisture content C (kg m^{-2}) and canopy capacity Cm = 0.5 + 0.05 Λ. The urban tile is also given a small surface capacity of 0.5 kg m^{-2}.

	dr (m)
Broad leaf trees	3
Needle leaf trees	1
C3 grass	0.5
C4 grass	0.5
Shrubs	0.5

Table 6: Look-up table for root depths in JULES

2.3 Surface energy balance

Surface temperature T is interpreted as a surface skin temperature or canopy layer temperature from the canopy model. The surface energy balance without snowmelt can be written as

$$C_c \frac{dT}{dt} = R_N + Q_H - H - LE - G_0 \tag{9}$$

Where, the surface net radiation (RN) is

$$RN = SWN + LW\downarrow - e\,\sigma T^4 \tag{10}$$

where, H and E are fluxes of sensible heat and moisture, QH is the anthropogenic heat source and L is the latent heat of vaporization for snow-free tiles or sublimation for snow-covered or ice tiles. The heat flux into the ground (G_0) is estimated as a combination of radiative flux (assuming only one reflection) and turbulent fluxes below vegetation canopies and conductive fluxes for the non-vegetated fraction.

$$G_0 = f_r \left[\epsilon_c \epsilon_s \left(\sigma T^4 - \sigma T_s^4\right) + c_p RK_{Hcan} \left(T - T_s\right)\right] + \left(1 - f_r\right) \frac{2\lambda}{\Delta z_s} \left(T - T_s\right) \tag{11}$$

where, Δz_s and T_s are the thickness and temperature of the surface soil layer, RK_{Hcan} is the turbulent exchange coefficient between the canopy and the underlying soil and *s* is the emissivity of the soil. Canopy fraction f_r is provided, if the canopy model is selected however is set to zero otherwise. The thermal conductivity λ is equal to the soil conductivity λ soil for snow-free tiles, but is adjusted for insulation by snow of depth d according to

$$\lambda = \begin{cases} \lambda_{soil} \left[1 + \frac{2d}{\Delta z_s} \left(\frac{\lambda_{soil}}{\lambda_{snow}} - 1\right)\right]^{1} & d < \Delta z_z/2 \\ \lambda_{snow} & d >= \Delta z_z/2 \end{cases}$$

with $\lambda_{snow} = 0.265\ \mathrm{W\ m^{-1} K^{-1}}$.

Expressions for surface fluxes of sensible heat and moisture over each tile are derived from the bulk aerodynamic formula.

$$H = c_p RK_H (1) \left[T - T_1 - \frac{g}{c_p} \left(z_1 + z_0 - z_{oh}\right)\right] \tag{12}$$

and

$$E=\psi RK_H(1)\left[q_{sat}(T,p)-q_1\right] \tag{13}$$

where, RKH (1) = ρCH U1 ; ρ and cp are the density and heat capacity of air.

q_{sat} can be linearized to give the following equation

$$q_{sat}\left(T^{n+1},p\right)\approx q_{sat}\left(T^{n},p\right)+D\left(T^{n+1}-T^{n}\right)$$

where,

$$D=\frac{q_{sat}\left(T^{(n)},p\right)-q_{sat}\left(T_1^{(n)},p\right)}{T^{(n)}-T_1^{(n)}}$$

By writing T $_{n+1}$ = T $_n$ + ΔT and linearizing (T)^4 as

$$(T)^4 \approx (T)^4 + 4(T)^3\,(T-T*)$$

the equations for the turbulent heat and moisture fluxes and the soil heat flux can be written as

$$H = H_{ex} + cp\ RKH\ (1)\ \Delta T \tag{14}$$

$$E = E_{ex} + \psi RKH\ (1)\ \Delta T \tag{15}$$

$$G_0=G_{0ex}+\left[f_r\left(4\epsilon_c\epsilon_s\sigma\left(T^n\right)^3+c_pRK_{Hcan}\right)+\left(1-f_r\right)2\frac{\lambda}{\Delta}z_s\right]\Delta T \tag{16}$$

where, H_{ex} and E_{ex} are the explicit heat and moisture fluxes and G_{0ex} is the explicit soil heat flux.

Discretizing the time derivative of T between timesteps n and n + 1 as

$$\frac{dT}{dt}\approx\frac{T^{n+1}-T^{n}}{\Delta t}$$

and using equations to obtain an expression for T gives

$$\Delta T=\frac{R_N^n+Q_H-H_{ex}-E_{ex}-G_{0\mathrm{ex}}}{RK_H(1)\left(c_p+LD\psi\right)+A_s'} \tag{17}$$

where, RN is the explicit net radiation and

$$R_N^n=SW_N+LW\downarrow-\epsilon_c\sigma\left(T^n\right)^4$$

$$A_s'= 4\left(1+\epsilon_sf_r\right)\epsilon_c\sigma\left(T^n\right)^3+f_r\,dpRK_{Hcan}+\left(1-f_r\right)2\frac{\lambda}{\Delta z_s}+\frac{C_c}{\Delta t}$$

The value of the surface temperature from equation (17) can then be used in equations (14) to (16) to obtain fluxes that are consistent with the Penman-Monteith equations.

2.4 Implicit boundary layer fluxes

Increments in temperatures on boundary-layer levels k = 1, . . . , N are calculated as

$$\delta T_k = g\Delta t / \Delta p_k \left[F_T(k+1) - F_T(k)\right]$$

where the fluxes are

$$F_T(k) = -RK_H(k)\left[\frac{T_k - T_{k-1}}{\Delta z_{k-1/2}} + \frac{g}{c_p}\right]$$

for 1 < k ≤ N with boundary conditions FT (N + 1) = 0 and FT (1) = H/cp for grid box mean surface sensible heat flux is

$$\bar{H} = \sum_j v_j H_j$$

Implicit fluxes during timestep n are calculated using

$$T_k = (1-\gamma_k)T_k^{(n)} + \gamma_k T_k^{(n+1)}$$
$$T_k = T_k^{(n)} + \gamma_k \delta T_k$$

where, γ k is the forward timestep weighting factor for level k. This gives a tridiagonal system of equations

BT N δTN + CT N δTN −1 = (δTN)ex

AT k δTk+1 + BT k δTk + CT k δTk−1 = (δTk)ex k = 2, . . . , N − 1

AT 1 δT2 + BT 1 δT1 = (δT1)ex − (gΔt/Δp1)FT (1)

With the following matrix elements

$$A_{Tk} = \gamma_{k+1}\frac{g\Delta t}{\Delta p_k}\frac{RK_H(k+1)}{\Delta z_{k+1/2}}$$

$$B_{Tk} = \begin{cases} 1 - C_{TN} & k = N \\ 1 - A_{Tk} - C_{TN} & k = 2,\ldots,N-1 \\ 1 - A_{Tk} & k = 1 \end{cases}$$

and

$$C_{Tk} = \gamma_k \frac{g\Delta t}{\Delta p_k}\frac{RK_H(k)}{\Delta z_{k-1/2}} \quad k=2,\ldots,N$$

The explicit increments on the RHS of equation are

$$(\delta T_k)_{ex} = -(g\Delta t/\Delta p_N)F_T^{(n)}(N)$$
$$(\delta T_k)_{ex} = -(g\Delta t/\Delta p_N)\left[F_T^{(n)}(k+1) - F_T^{(n)}(k)\right] k = 2,\ldots,N$$
$$(\delta T_k)_{ex} = -(g\Delta t/\Delta p_1)F_T^{(n)}(2) \quad k=1$$

where, the explicit fluxes with temperatures at the beginning of the timestep. A downward sweep to eliminate the below-diagonal elements gives

Tk + CT k δTk−1 = δTk k = 2, . . . , N

δT1 = δT1 − βFT (1)

Where, CT k = CT k /BT k with

$$B'_{Tk} = 1 - C_{TN} \quad \text{k=N}$$

$$B'_{Tk} 1 - A_{Tk}\left(1 + C'_{Tk+1}\right) - C_{Tk} \quad \text{k=2,...,N-1}$$

$$B'_{Tk} = 1 - A_{T1}\left(1 + C'_{T2}\right) \quad \text{k=1}$$

$$\beta = \frac{g\Delta t}{\Delta p_k} \frac{1}{B'_{TK}}$$

and

$$\left(\delta T'_k\right) = \left(\delta T_N\right)_{ex} / B'_{TN}$$

After adjustment of the surface fluxes, an upward sweep through the matrix equation gives temperature increments

δT1 = δT1 − β H/cp

δTk = δTk − CT k δTk−1

k = 2, . . . , N

and humidity increments

δQ1 = δQ1 − βE,

δQk = δQk − CT k δQk−1

Interpolation method as currently used by the boundary-layer scheme is used for screen level exchange coefficients. Air temperature and humidity over tiles are calculated and averaged to give gridbox-mean values, which are converted from cloud-conserved forms to actual temperatures and humidity. This conversion is required, if level-1 cloud is present.

2.5 Hydrology

The partitioning of precipitation into interception, through-fall, runoff and infiltration is applied separately on each tile. For rainfall rate (*R)* covering fraction of a grid box (1 for large-scale rain or condensation and 0.3 for convective rain, the through-fall from the canopy on a vegetated tile is calculated as

$$T_F = R\left(1-\frac{C}{C_m}\right)\exp\left(\frac{-\epsilon C_m}{R\Delta t}\right)+R\frac{C}{C_m} \tag{18}$$

and the tile canopy water content is updated by C (n+1) = C (n) + (R − TF)Δt.
Surface run off is calculated as

$$Y = \begin{cases} R\frac{C}{C_m}\exp\left(\frac{-eKC_m}{RC}\right)+R\left(1-\frac{C}{C_m}\right)\exp\left(\frac{-\ C_m}{R\Delta\Delta}\right); K\Delta\Delta\ \ C \\ R\exp\left[\frac{-\ (K\Delta\Delta+C_m-C)}{R\Delta\Delta}\right]; K\Delta\Delta > C \end{cases} \tag{19}$$

where, the surface infiltration rate K is equal to β Ks ; Ks is the soil saturated hydrological conductivity and β is an enhancement factor, values of which are given in Table 7. Runoff of melt water is calculated using snowmelt rate Sm in place of R. The flux of water into the soil is given by the grid box average.

$$W_0=\sum_j v_j\left(T_{Fj}+S_{mj}-Y_j\right) \tag{20}$$

Vegetation/Soil Type	***B***
Broadleaf trees	4
Needle leaf trees	4
C3 grass	2
C4 grass	2
Shrubs	2
Urban	0.1
Water	0
Soil	0.5
Ice	0

Table 7: Look-up table for Infiltration enhancement factors

2.6 Soil Thermodynamics

Discretized form of the heat diffusion equation is used to predict the subsurface temperatures, which is coupled to the soil hydrology module through soil water phase changes and the associated latent heat and soil thermal characteristics which are dependent on soil moisture content, including both liquid water and ice.

The temperature of the nth soil layer, of thickness Δz n, is incremented by the diffusive heat fluxes into and out of the layer, G_{n-1} and G_n respectively, and the net heat flux, J_n, advected from the layer by the moisture flux:

$$C_A \Delta z_n \frac{dT_n}{dt} = G_{n-1} - G_n - J_n \Delta z_n \tag{21}$$

The diffusive and advective fluxes are given by:

$$G = \lambda \frac{\partial T}{\partial z} \tag{22}$$

$$J = c_w W \frac{\partial T}{\partial z} \tag{23}$$

where, z is the vertical coordinate, W is the vertical flux of soil moisture (calculated within the soil hydrology module), cw is the specific heat capacity of water, and λ is the local soil thermal conductivity, modified in the presence of lying snow. The "apparent" volumetric heat capacity of the layer is calculated by:

$$C_A = C_s + \rho_w c_w \Theta_u + \rho_i c_i \Theta_f + \rho_w \left[(c_w - c_i) T + L_f \right] \frac{\partial \Theta_u}{\partial T} \tag{24}$$

where, Θ_u and Θ_f are the volumetric concentrations of frozen and unfrozen soil moisture, and ρ_i and c_i are the density and specific heat capacity of ice. The first three terms on the right hand side of above equation represent contributions from dry soil, liquid water and ice, and the final term is the apparent heat capacity associated with phase changes. The relationship between unfrozen water concentration, Θu, and temperature, T , can be derived by minimizing the Gibbs free energy of the soil-water-ice system. This results in an equation relating the water suction, Ψ (m), to the temperature, T (K), when ice is present:

$$\psi = \left[-k\frac{(T - T_m)}{\psi_s} \right]^{-1/b}$$

(25)

where, T$_m$ (K) is the freezing point of pure water, *g* is the acceleration due to gravity and k is a dimensionless constant which depends on the nature of the soil. A value of k = 1.0 is assumed, which is consistent with a clay-rich soil for which absorption forces dominate over capillary forces. (k = 2.2 would be more appropriate for granular soils). Combining above equations with form equation for the suction as a function of liquid water yields

$$\frac{\Theta_u^{max}}{\Theta_s} = \left[\frac{-k(T - T_m)}{\psi_s} \right]^{-(1/b)}$$

(26)

where, Θmax is the maximum unfrozen water that can exist at temperature T , Θ s is the saturation u, soil moisture concentration, Ψs and b are other soil specific parameters and κ is a constant defined by:

$$k = k\frac{\rho_i}{\rho_w}\frac{L_f}{gT_m} \approx 114.3\, mK^{-1}$$

(27)

the actual value of Θ_u is limited by the total water content of the soil:

$$\Theta_u = \min\{\Theta_{max}, \Theta\}$$

where, Θ is the “liquid” total volumetric concentration, i.e. that which would arise if all the moisture was in liquid form:

$$\Theta = \Theta_u + \frac{\rho_i}{\rho_w}\Theta_f$$

The temperature, above which all soil moisture is unfrozen, T$_{max}$, can be derived by equating Θ to Θmax in Equation (25):

$$T_{max} = T_m - \frac{\psi_s}{k}\left[\frac{\Theta_s}{\Theta}\right]^b$$

(28)

The second term on the right hand side represents the suppression of the initial freezing point. It is useful to rewrite equation in terms of two distinct temperature regimes:

$$\Theta_u = \begin{cases} \Theta_{max} \text{ if } T < T_{max} \\ \Theta \text{ if } T \geq T_{max} \end{cases}$$

Then differentiation with respect to temperature yields:

$$\frac{\partial \Theta_u}{\partial T} = \frac{k\Theta_s}{b\psi_s} - k\frac{(T-T_m)^{(-1/b-1)}}{\psi_s} \tag{29}$$

if T < T_{max}

0 if T > T_{max}

Which is used in equation. The surface soil heat flux, G_0, is calculated in boundary layer as a residual in Equation (9). Heat advection by surface infiltration is currently neglected. The lower boundary condition corresponds to zero vertical gradients in soil temperature.

2.7 Soil Hydrology

Richards' equation (Richards, 1931) is implemented to calculate the soil hydrology components. Finite difference approximation form of Richards' equation is used with the same vertical discretization as the soil thermodynamics module. The prognostic variables of the model are the total soil moisture content within each layer:

$$M = \rho w \, \Delta z \, \Theta s \, \{S_u + S_f\} \tag{30}$$

Where, Δz is the thickness of the layer, and S_u and S_f are the mass of unfrozen and frozen water within the layer as a fraction of that of liquid water at saturation:

$$S_u = \frac{\Theta_u}{\Theta_s} \tag{31}$$

$$S_f = \frac{\rho_i}{\rho_w}\frac{\Theta_f}{\Theta_s} \tag{32}$$

The total soil moisture content within the nth soil layer is incremented by the diffusive water flux flowing in from the layer above, W_{n-1}, the diffusive flux flowing out to the layer below, W_n and the evapotranspiration extracted directly from the layer by plant roots and soil evaporation, E_n:

$$\frac{dM_n}{dt} = W_{n-1} - W_n - E_n \tag{33}$$

E_n is calculated from the total evapotranspiration, E_t, based on the profiles of soil moisture and root density, E_n = *en* E_t. *en* is the weighting factor. The water fluxes in layers are given by the Darcy equation:

$$W=K\left[\frac{\partial\psi}{\partial z}+1\right] \tag{34}$$

where, K is the hydraulic conductivity and Ψ is the soil water suction. To close the model it is necessary to assume forms for the hydraulic conductivity and the soil water suction as a function of the soil moisture concentration, the Clapp and Hornberger relations are currently used by default.

$$\psi=\psi_s S_u^{-b} \tag{35}$$

$$K=K_s S_u^{2b+3} \tag{36}$$

where, Ks , Ψs and b are empirical soil dependent constants. The interpretation of the Clapp-Hornberger relations in terms of unfrozen rather than total soil moisture is consistent with the observation that the freezing of soil moisture reduces hydraulic conductivity and produces a large suction by reducing the unfrozen water content. The top boundary condition for the soil hydrology module is given by Equation (20). “free drainage” is considered as default boundary condition:

$$WN = K\,N \tag{37}$$

where, WN is the drainage from the lowest deepest soil layer and KN is the hydraulic conductivity of this layer.

2.8 Soil numeric

The soil numeric in this scheme is different from the old version of MOSES for updating soil moisture and soil temperatures through Equations (21) and (33). MOSES -1 used a simple explicit scheme, in which the fluxes on the RHS of these equations are calculated from the beginning of timestep values of T and M. By contrast, this scheme includes an implicit scheme which remains numerically stable and accurate at much longer timesteps and higher vertical resolution. This has many advantages to the seamless prediction system. This scheme has a relatively small impact on the model performance at the standard soil model resolution (4 soil layers with thicknesses from the top of 0.1, 0.25, 0.65, 2.0 m), it does make it feasible for users to choose many more soil layers without incurring massive computational costs. The prognostic equations for the soil (Equations 21 and 33) take the form

$$\frac{dY_n}{dt} = F_{n-1} - F_n - s_n \tag{38}$$

where Yn = {Tn , Mn }, Fn = {Gn /CA , Wn } and sn = {Jn /CA , En }. The fluxes F_n are a function of the prognostic variables Y_n. In the explicit scheme used in previous MOSES the F_n were calculated using the t values of Y_n at the beginning of timestep t, denoted Y_n . However here it is calculated using a forward timestep weighting, γ, such that,

$$F_n = F_n^t + \gamma \frac{\partial F_n}{\partial Y_n} \Delta Y_n + \gamma \frac{\partial F_n}{\partial Y_{n+1}} \Delta Y_{n+1} \tag{39}$$

where, ΔY_n is the increment to Y_n during the timestep t to t + Δt. The derivatives of the fluxes with respect to the prognostic variables are calculated in subroutines DARCY, HYD CON and SOIL HTC. The equation (39) can be substituted into equation (38) to yield a series of n simultaneous equations for the n prognostic variables:

$$a_n \Delta Y_{n-1} + b_n \Delta Y_n + c_n \Delta Y_{n+1} = d_n \tag{40}$$

Where,

$$a_n = -\gamma \Delta t \frac{\partial F_{n-1}}{\partial Y_{n-1}}$$

$$b_n = \Delta z - \gamma \Delta t \left[\frac{\partial F_{n-1}}{\partial Y_n} - \frac{\partial F_n}{\partial Y_n} \right]$$

$$c_n = -\gamma \Delta t \frac{\partial F_n}{\partial Y_{n+1}}$$

$$d_n = \Delta t \left[F_{n-1}^t - F_n^t - s_n^t \right]$$

The explicit update to the variable Y_n as is represented in left hand side and no implicit correction is made to the sink term, *sn*, since this would require an unwieldy implicit update to the entire coupled soil hydrology, soil thermodynamics and boundary layer system. Explicitly treatment of this term decouples the updates to the soil temperatures and soil moisture, such that these variables can be incremented independently on each timestep. Tridiagonal set of equations (40) are solved routinely by Gaussian elimination. Another major numerical difference between MOSES-1 and this scheme involves the treatment of super-saturation in a soil layer.

2.9 Gaussian elimination

The set of equations represented by (40) is solved by a two step algorithm. The first step is an upward sweep, the ΔY_{n+1} terms are eliminated by transforming the nth equation, Eq (n), thus (40) becomes

$$Eq(n) \dashrightarrow Eq(n)' = b_{j+1}' \, Eq(n) - cn \, Eq(n+1)' \qquad (41)$$

Where ′ denotes a transformed equation or variable. Under this transformation the nth equation becomes

$$an \, \Delta Yn-1 + bn \, \Delta Yn = dn \qquad (42)$$

Where,

$$an = bn{+}1 \, an \qquad (43)$$

$$bn = bn{+}1 \, bn - an{+}1 \, cn$$

$$dn = bn{+}1 \, dn - dn{+}1 \, cn$$

In the upward sweep *an, bn* and *dn* are evaluated iteratively beginning at the lowest soil layer (N), where the lower boundary conditions of the soil model imply c N = 0 such that *aN = aN, bN = bN* , and *d'N = dN* .

In second step is a downward sweep, the increments to the prognostics variables ΔYn are derived iteratively from the top downwards using Equation (42).

2.10 Input and output variables

The Input to JULES contains both initial conditions and atmospheric forcing. The input and output variables are listed below.

Initial conditions:

1. Amount of intercepted water that is held on each tile.
2. Soil carbon (kg m^{-2}).
3. Stomatal conductance for water vapour (m s^{-1}).
4. Total snow on the tile
5. Soil wetness for each soil layer.
6. Temperature of each soil layer (K).
7. Temperature of each tile (K) (surface or skin temperature).

8. Leaf area index of each PFT.
9. Height (m) of each PFT.
10. The fraction of land area of each gridbox that is covered by each surface type.
11. Soil wetness in the deep (“water table”) layer beneath the standard soil column.
12. Depth from the surface to the water table.
13. Snow surface grain size (μm) on each tile.
14. Bulk density of lying snow (kg m^{-3}).
15. Depth of snow (kg m).
16. Amount of snow on the ground, beneath the canopy (kg m^{-2}), on each tile.
17. The number of snow layers on each tile.
18. Depth of snow in each layer (kg m).
19. Mass of frozen water in each snow layer (kg m^{-2}).
20. Mass of liquid water in each snow layer (kg m^{-2}).
21. Temperature of each snow layer (K).
22. Snow grain size (μm) on each tile in each snow layer.

List of JULES forcing variables:

1. Air pressure (Pa).
2. Specific humidity (kg kg^{-1}).
3. Air temperature (K).
4. Radiation
5. Net (all wavelength) downward radiation (W m^{-2}).
6. Net downward longwave radiation (W m^{-2}).
7. Net downward shortwave radiation (W m^{-2}).
8. Downward longwave radiation (W m^{-2}).
9. Downward shortwave radiation (W m^{-2}).
10. Diffuse radiation (W m^{-2}).
11. Precipitation rate (kg m^{-2} s^{-1}).
12. Rainfall rate (kg m^{-2} s^{-1}).
13. Snowfall rate (kg m^{-2} s^{-1}).
14. Large-scale rainfall rate (kg m^{-2} s^{-1}).
15. Convective rainfall rate (kg m^{-2} s^{-1}).
16. Large-scale snowfall rate (kg m^{-2} s^{-1}).
17. Convective snowfall rate (kg m^{-2} s^{-1}).
18. Total wind speed (m s^{-1}).
19. Zonal component of the wind (m s^{-1}).
20. Meridional component of the wind (m s^{-1}).

Output variables:

Variables that have a single value at each gridpoint (land and sea) are:

1. Gridbox convective rainfall (kg m^{-2} s^{-1}).
2. Gridbox convective snowfall (kg m^{-2} s^{-1}).

3. Cosine of the zenith angle.
4. Gridbox fraction of radiation that is diffuse.
5. Gridbox mean evaporation from canopy/surface store (kg m s).
6. Gridbox sublimation from lying snow or sea-ice (kg m^{-2} s^{-1}).
7. Gridbox surface evapotranspiration from soil moisture store (kg m^{-2} s^{-1})
8. Gridbox moisture flux from surface (kg m^{-2} s^{-1}).
9. Gridbox surface sensible heat flux (W m^{-2}).
10. Gridbox albedo for waveband 1 (direct beam visible).
11. Gridbox albedo for waveband 2 (diffuse visible).
12. Gridbox albedo for waveband 3 (direct beam NIR).
13. Gridbox albedo for waveband 4 (diffuse NIR).
14. Gridbox surface latent heat flux (W m^{-2}).
15. Gridbox large-scale rainfall (kg m^{-2} s^{-1}).
16. Gridbox large-scale snowfall (kg m^{-2} s^{-1}).
17. Gridbox surface downward LW radiation (W m^{-2}).
18. Gridbox precipitation rate (kg m^{-2} s^{-1}).
19. Gridbox surface pressure (Pa).
20. Gridbox specific humidity at 1.5m height (kg kg^{-1}).
21. Gridbox specific humidity (total water content) (kg kg^{-1}).
22. Gridbox rainfall rate (kg m^{-2} s^{-1}).
23. Gridbox heat flux used for surface melting of snow (W m-2).
24. Gridbox snowfall rate (kg m^{-2} s^{-1}).
25. Gridbox snowmass (kg m^{-2}).
26. Gridbox net downward heat flux at surface over land and sea-ice fraction of gridbox (Wm^{-2}).
27. Gridbox surface downward SW radiation (W m^{-2}).
28. Gridbox temperature at 1.5m height (K).
29. Gridbox westerly component of surface wind stress (N m^{-2}).
30. Gridbox southerly component of surface wind stress (N m^{-2}).
31. Gridbox ice/liquid water temperature (K).
32. Gridbox surface temperature (K).
33. Gridbox westerly wind component (m s^{-1}).
34. Gridbox westerly wind component at 10 m height (m s^{-1}).
35. Gridbox southerly wind component (m s^{-1}).
36. Gridbox southerly wind component at 10m height (m s^{-1}).
37. Gridbox wind speed (m s^{-1})

Variables that have a single value at each land grid point:

1. Gridbox albedo.
2. Gridbox canopy water content (kg m^{-2}).
3. Gridbox total soil carbon (kg C m^{-2}).
4. Gridbox mean vegetation carbon at end of model timestep (kg C m^{-2}).
5. Gridbox depth of frozen ground at surface (m).
6. Gridbox depth of unfrozen ground at surface (m).

7. Gridbox drainage at bottom of soil column (kg m^{-2} s^{-1}).
8. Gridbox mean evaporation from lakes (kg m^{-2} s^{-1}).
9. Gridbox emissivity.
10. Gridbox scaled methane flux from wetland fraction (10-9 kg C m^{-2} s^{-1}).
11. Gridbox surface saturated fraction .
12. Gridbox soil moisture availability factor (beta).
13. Gridbox wetland fraction at end of model timestep.
14. Gridbox gross primary productivity (kg C m^{-2} s^{-1}).
15. Gridbox surface conductance to evaporation (m s^{-1}).
16. Gridbox snowmelt heat flux (W m^{-2}).
17. Index (gridbox number) of land points.
18. Index (gridbox number) of land ice points.
19. Gridbox mean carbon litter (kg C m-2 (360days)-1).
20. Gridbox surface net LW radiation (W m^{-2}).
21. Gridbox surface upward LW radiation (W m^{-2}).
22. Gridbox net primary productivity (kg C m^{-2} s^{-1}).
23. Gridbox baseflow (lateral subsurface runoff) (kg m^{-2} s-1).
24. Gridbox baseflow (lateral subsurface runoff) from deep layer (kg m^{-2} s^{-1}).
25. Surface net radiation (W m^{-2}).
26. Gridbox plant respiration (kg C m^{-2} s^{-1}).
27. Gridbox total soil respiration (kg C m^{-2} s^{-1}).
28. Gridbox mean soil respiration for driving TRIFFID (kg C m^{-2} (360days)-1).
29. Gridbox runoff rate (kg m^{-2} s^{-1}).
30. Gridbox saturation excess runoff rate (kg m^{-2} s^{-1}).
31. Gridbox available moisture in surface layer of depth given by zsmc (kg m^{-2}).
32. Gridbox available moisture in soil column (kg m^{-2}).
33. Gridbox total soil moisture in column (kg m^{-2}).
34. Gridbox sub-canopy snowmelt heat flux (W m^{-2}).
35. Gridbox snow on canopy (kg m^{-2}).
36. Gridbox depth of snow (m).
37. Gridbox snow-covered fraction of land points.
38. Gridbox average snow beneath canopy (snow_grnd) (kg m^{-2}).
39. Gridbox frozen water in snowpack (kg m^{-2}). Only available if nsmax > 0.
40. Gridbox liquid water in snowpack (kg m^{-2}). Only available if nsmax > 0.
41. Gridbox rate of snowmelt (kg m^{-2} s^{-1}).
42. Index (gridbox number) of soil points.
43. Soil wetness in the deep (water table) layer .
44. Gridbox sub-surface runoff (kg m^{-2} s^{-1}).
45. Gridbox surface runoff (kg m^{-2} s^{-1}).
46. Gridbox unfrozen soil moisture as fraction of saturation.
47. Gridbox soil moisture as fraction of saturation .
48. Gribox net shortwave radiation at the surface (W m^{-2}).
49. Gridbox throughfall (kg m^{-2} s^{-1}).
50. Gridbox effective radiative temperature (K).

51. Gridbox mean depth to water table (m).
52. Gridbox mean isoprene emission flux (kg m^{-2} s^{-1}).
53. Gridbox mean monoterpene emission flux (kg m^{-2} s^{-1}).
54. Gridbox mean methanol emission flux (kg m^{-2} s^{-1}).
55. Gridbox mean monoterpene emission flux (kg m^{-2} s^{-1}).

Variables that have a value for each plant functional types (PFT) at each land gridpoint:

1. PFT total carbon content of the vegetation at end of model timestep (kg C m^{-2}).
2. PFT canopy height (m).
3. PFT flux of O_3 to stomata (mol m^{-2} s^{-1}).
4. PFT soil moisture availability factor.
5. PFT leaf turnover rate (360days $^{-1}$).
6. PFT mean leaf turnover rate for input to PHENOL (360days^{-1}).
7. PFT mean leaf turnover rate for driving TRIFFID (360days^{-1}).
8. PFT mean leaf turnover rate over phenology period (360days^{-1}).
9. PFT gross primary productivity (kg C m^{-2} s^{-1}).
10. PFT leaf area index.
11. PFT leaf area index after phenology.
12. PFT carbon litter (kg C m-2 (360days) $^{-1}$).
13. PFT mean NPP for driving TRIFFID (kg C m^{-2} (360days) $^{-1}$).
14. PFT net primary productivity (kg C m^{-2} s^{-1}).
15. PFT ozone exposure factor.
16. PFT plant respiration (kg C m^{-2} s^{-1}).
17. PFT mean wood respiration for driving TRIFFID (kg C m^{-2} (360days)$^{-1}$).
18. PFT wood respiration (kg C m^{-2} s^{-1}).
19. PFT isoprene emission flux (kg m^{-2} s^{-1}).
20. PFT monoterpene emission flux (kg m^{-2} s^{-1}).
21. PFT methanol emission flux (kg m^{-2} s^{-1}).
22. PFT monoterpene emission flux (kg m^{-2} s^{-1}).

Variables that have a value for each tile at each land gridpoint:

1. Tile land albedo, waveband 1 (direct beam visible).
2. Tile land albedo, waveband 2 (diffuse visible).
3. Tile land albedo, waveband 3 (direct beam NIR).
4. Tile land albedo, waveband 4 (diffuse NIR).
5. Anthropogenic heat flux for each tile (W m^{-2}).
6. Tile surface/canopy water for snow-free land tiles (kg m^{-2}).
7. Tile surface/canopy water capacity of snow-free land tiles (kg m^{-2}).
8. Tile evaporation from canopy/surface store for snow-free land tiles (kg m^{-2} s^{-1}).
9. Tile sublimation from lying snow for land tiles (kg m^{-2} s^{-1}).
10. Tile emissivity.
11. Tile surface evapotranspiration from soil moisture store for snow-free land (kg $m^{-2}s^{-1}$).

12. Tile surface moisture flux for land tiles (kg m^{-2} s^{-1}).
13. Tile surface sensible heat flux for land tiles (W m^{-2}).
14. Tile surface conductance to evaporation for land tiles (m s^{-1}).
15. Tile surface latent heat flux for land tiles (W m^{-2}).
16. Tile number of snow layers .
17. Tile specific humidity at 1.5m over land tiles (kg kg^{-1}).
18. Tile surface net radiation (W m^{-2}).
19. Tile snow surface grain size (μm).
20. Tile melt of snow on canopy (kg m^{-2} s^{-1}).
21. Tile snow on canopy (kg m^{-2}).
22. Tile snow depth (m).
23. Tile bulk density of snow on ground (kg m^{-3}).
24. Tile snow on ground below canopy (kg m^{-2}).
25. Tile snow on ground (snow_tile or snow_grnd depending on configuration) (kg m^{-2}).
26. Tile total frozen mass in snow on ground (kg m^{-2}). Only available if nsmax > 0.
27. Tile total liquid mass in snow on ground (kg m^{-2}).
28. Only available if nsmax > 0.
29. Tile lying snow (total) (kg m^{-2}).
30. Tile snow melt rate (melt_tile) (kg m^{-2} s^{-1}).
31. Downward heat flux for each tile (W m^{-2}).
32. C*(dT/dt) for each tile (W m^{-2}).
33. Tile temperature at 1.5m over land tiles (K).
34. Tile surface temperature (K).
35. Tile surface roughness (m).

Variables that have a value for each surface type at each land grid point:

1. Fractional cover of each surface type.
2. Index (gridbox number) of land points with each surface type

Variables that have a value for each soil level at each land grid point:

1. Brooks-Corey exponent for each soil layer.
2. Extraction of water from each soil layer (kg m^{-2} s^{-1}).
3. Dry soil heat capacity (J K^{-1} m^{-3}) for each soil layer.
4. Dry soil thermal conductivity (W m^{-1} K^{-1}) for each soil layer.
5. Saturated hydraulic conductivity (kg m^{-2} s^{-1}) for each soil layer.
6. Saturated soil water pressure (m) for each soil layer.
7. Moisture content of each soil layer (kg m^{-2}).
8. Total moisture content of each soil layer, as fraction of saturation .
9. Frozen moisture content of each soil layer as a fraction of saturation .
10. Unfrozen moisture content of each soil layer as a fraction of saturation .
11. Sub-surface temperature of each layer (K).
12. Volumetric moisture content at critical point for each soil layer .

13. Volumetric moisture content at saturation for each soil layer.
14. Volumetric moisture content at wilting point for each soil layer .

Variables that have a value for each soil carbon pool at each land grid point:

1. Carbon in each soil pool (kg C m^{-2}).
2. Respiration rate from each soil carbon pool (kg C m^{-2} s^{-1})

Variables that have a value for each snow layer on each tile at each land grid point:

1. Grain size in snow layers for each tile (µm).
2. Depth of each snow layer for each tile (m).
3. Mass of ice in each snow layer for each tile (kg m^{-2}).
4. Mass of liquid water in each snow layer for each tile (kg mP).
5. Temperature of each snow layer (K).

2.11 Update and configure

The surface variables in UM can be updated using updating and reconfiguring options available in UM User Interface (UMUI). UMUI can be used to set the path and variables. The UMUI selection options are shown below

Model Selection

--Atmosphere

--Ancillary and input data files

-- Climatologies and potential climatologies

-- Soil moisture and snow depth

-- Deep soil temperature

-- Soil VSMC, soil hydraulic and thermal conductivity

-- Effective vegetation parameters

-- Vegetation distribution

-- SST/Skin temperature

-- Sea ice

-- River routing

-- Large scale hydrology

The above options make it possible to be configured or updated from the climatology or from Startdump file. Configure option updates the start file of the model, however the updating option directly change the state variables during the model simulation. The following soil moisture fields can be updated or configured using UMUI: The soil moisture, snow depth, layer soil moisture content, deep soil temperature ancillary files volumetric soil moisture concentration at wilting point, volumetric soil moisture content at critical point and saturation and saturation hydraulic conductivity.

Similarly following vegetation ancillary files can be updated: vegetation fraction of plant function types, root depth, snow free surface albedo, deep free albedo, surface resistance to evaporation, canopy capacity, leaf area index of plant function, canopy height of plant function, canopy conductance and fraction of surface types required for surface and vegetation characterization. Optionally river routing needs ancillary direction and sequence of the river.

2.12 Static land surface data

The model start file contains many static variables as an input to JULES. The important static variables required by JULES are land sea mask, deep soil temperature ancillary files volumetric soil moisture concentration at wilting point, volumetric soil moisture content at critical point, saturation and saturation hydraulic conductivity, thermal conductivity, thermal capacity, snow free albedo, canopy conductance, soil carbon content, fraction of surface types. Land sea mask is derived from the IGBP surface types. The master data set and their sources for various land surface parameters used at NCMRWF are shown in Table 8. Table 8 also shows the list of data sets used to derive the static land surface parameters to input JULES. The start file variables are derived from the master data sets shown in Table 8. ANCIL/CAP utilities were used to process the master data. Details of the ANCIL/CAP Utility can be found at https://code.metoffice.gov.uk/doc/ancil/

2.13. Incorporating ISRO land use/land cover

The global IGBP data is converted to a netCDF format first and then ovet the Indian region IGBP data is replaced with regional NRSC land cover. The merged netcdf file was again converted back to original binary IGBP master file format. FORTRAN programs were used to merge these data sets. Technical details of merging are provided in this section.

The master global data is in direct access binary file format. The 18 land cover classes are stored as a single byte characters. Table 9 shows the land cover class and corresponding alphabet representation in the file. This is a global file of 30 arc second resolution. The header of file has six parameter informations including the x and y dimension size, increment and starting points. The binary single byte character data is converted to integer type data file for processing. The program for reading and converting are given at Appendix-1 to Appendix-5.

The data is converted to netCDF integer format for processing and visualization. The FORTRAN program is used for this purpose is given at Appendix-2. The land cover type and corresponding integer is provided in Table 10.

The NRSC ISRO regional land cover data is provided in netCDF format. The ISRO regional land cover data is then merged with the global IGBP master data in netcdf format. The FORTRAN program used for this step is provided in Appendix-3.

The merged netCDF file is then converted to original master data format. The FORTRAN program used for this step is given in Appendix-4. Central Ancillary Program (CAP) is used to create the surface type fraction from master Lu/Lc data.

Data	**Source**
Land Sea mask	Derived from *IGBP* classification
Orography	*GLOBE*
Vegetation : Vegetation fraction, Vegetation functional types & LAI	IGBP / MODIS
Surface cover type (9 Class) : Soil – vegetation classification	IGBP (18 class IGBP data)
Soil data	*HWSD and IGBP* Percentage of Clay, sand silt from ISRIC STASGO USDA National soil survey center
Albedo	*WHS /*MODIS
Soil vegetation carbon	*HWSD and IGBP*
Soil dust	*HWSD and IGBP*

Table 8: The master land surface data and sources used at NCMRWF

evergreen needle leaf forest	A
evergreen broad leaf forest	B
deciduous needle leaf forest	C
deciduous broadleaf forest	D
mixed forest	E
closed shrub	F
open shrub	G
woody savannah	H
savannah	I
grassland	J
wetland	K
cropland	L
urban	M
mosaic	N
snow and ice	O
barren	P
water	Q
open sea	R
missing data	Z

Table 9: Land cover and character representation in binary master file.

evergreen needle leaf forest	1
evergreen broad leaf forest	2
deciduous needle leaf forest	3
deciduous broadleaf forest	4
mixed forest	5
closed shrub	6
open shrub	7
woody savannah	8
savannah	9
grassland	10
wetland	11
cropland	12
urban	13
mosaic	14
snow and ice	15
barren	16
water	17
open sea	18
missing data	19

Table 10: The land cover and integer representation in the converted netCDF file.

3. Structure of land surface scheme in Unified Model

3.1 Building and making

Flexible Configuration Management (FCM) tool developed by the UK Met Office is used to build the UM. The compiling system consists of two separate steps of `extract' and `build', both are independently controlled by FCM. Extract/Build system variables can be set using Unified Model User Interface (UMUI). During the first UM model run, the following sequence of two actions are performed, namely, Extraction and Build. The JULES code is extracted from UM repository along with the UMATMOS code and both are compiled into a single model executable. It contains two steps: Base Repository Extraction (Local machine extract directory is created). Atmospheric model and JULES codes (UMATMOS, JULES) are extracted from the UM repository at NCMRWF (svn://ncmutil4/UM_repos/) to the local directory *$UM_OUTDIR/baserepos*. Then source code is mirrored to the remote machine directory *$UM_ROUTDIR*.

Separate extracts are performed for each of the required model code: including the UM scripts, the atmosphere main model and the reconfiguration. Similarly the build contains two parts: one UM script build and another model build.

3.2 Calling sequence of land surface scheme within UM

The main program is of UM is UM_MAIN, which calls the top level subroutine UM_SHELL in the atmospheric model. Again, the main call to the model itself is in routine U_MODEL. The main routine for the time integration is *"incrtime"* and dynamics and physics routines of the atmosphere model are called from ATM_STEP. Figure 1-3 shows the program flow chart and structure. The following dynamical and physical calculations are done as following order in ATM_STEP subroutine. Initialization of idealized cases, updating of lower boundary conditions and polar filtering are performed as a first step. First set of atmospheric physics including Cloud/Precip/Tracer, SW & LW radiation, Gravity Wave Drag are called. Semi-Lagrangian Advection scheme is called next. The second set of atmospheric physics including Boundary Layer, Convection, and Hydrology (JULES) are executed next. After this, there is an optional AC Assimilation. Next the Helmholtz solver is called. The variables are updated to new time-level. Further, there is an optional Aerosol Modelling and IAU Assimilation and corrections are performed at end of each timestep.

3.3 The inputs to the land surface in NCUM

The UM input file (Startdump) contains many land surface initial conditions and surface variables like: soil moisture content, canopy water content, silhouette orography roughness, deep soil temperature, canopy water, snow amount, roughness length, land mask, orography, standard deviation and gradient, volumetric soil moisture at wilting point, critical point and saturation, saturated soil conductivity, soil thermal capacity and conductivity, saturated soil water suction, Clapp - Hornberger B Coefficient, etc. Spatial patterns of orography, roughness length, volumetric soil moisture (saturation, critical and wilting point), canopy height, fraction of surface types and LAI in the UM over Indian region are shown in Figures 4 to 9. The soil and surface static parameters including orography are taken from the UM's Startdump file. The land surface type data used are based on 9 surface types derived from the International Geosphere Biosphere Programme (IGBP).

4. Summary and Discussions

JULES model coupled to UM is one of the boundary forcing component of the seamless prediction system (NCUM) operational at NCMRWF. There is scope to improve the regional land surface features in land surface scheme by using the real time satellite data and a better land surface assimilation system. The available land surface datasets listed in Table 8 can be replaced with best regional data sets. ISRO (NRSC & SAC) satellite datasets available over Indian region can be utilized for this purpose.

There is also scope for the evaluation of land surface state variables (soil moisture, temperature, surface fluxes, etc.) of this land surface scheme using offline simulations which may bring the biases and problems in land surface model in a better diagnostic way, which can be used to improve the land surface scheme performance over Indian region. There is scope to improve the land surface regional scale data sets by using real time data sets which can be updated in model. Furthermore, there is a need for better regional representation of agriculture and other local processes in land surface scheme for high resolution requirements of the model. Land surface test beds can be implemented to improve the model in a long run.

Better initial conditions to the land surface scheme is an important factor in the evolution of the future state of atmosphere in seamless prediction systems, which highlight need for a better regional and global land surface data assimilation systems. This system can make use of all regional in-situ observations, multi-satellite soil moisture observations and multi-model offline land surface models along with present surface analysis system at NCMRWF.

Appendix-1

```
c Read IGBP 30 arc sec. data (17 class) and write in ASCII format
cccccccccccccccccccccccccccccccccccccccccccccccccccc
        CHARACTER(LEN=1), ALLOCATABLE :: CIGBP_CLASS(:,:)
   CHARACTER(LEN=1) EG, test
        integer REC, IR, IC, ic_2, ir_2
   INTEGER:: a(19)
c   ccc
        write(*,*)'=== ========================== ==='
   write(*,*)'Start reading IGBP data :::::'
        write(*,*)'Check input file : "lc" '
        write(*,*)'You should delete previous .out, .bin files :'
c      readfile : 'lc'
        OPEN(10,FILE='lc',FORM='FORMATTED',ACCESS='DIRECT',RECL=43200)
c      out files : ascii lulc, latitude, longitude
        OPEN(19,FILE='header.out',FORM='FORMATTED',status='new')
        OPEN(20,FILE='lc_ascii.out',FORM='FORMATTED',status='new')
        OPEN(21,FILE='lat_ascii.out',FORM='FORMATTED',status='new')
        OPEN(22,FILE='lon_ascii.out',FORM='FORMATTED',status='new')
        OPEN(23,FILE='lc_int_ascii.out',FORM='FORMATTED',status='new')
     a(1:16) = (/ 18,15,17,01,02,03,04,05,06,07,08,09,10,11,12,13 /)
     a(17)=14
     a(18)=16
     a(19)=19
c      read input file
        read(10,15,REC=1)NCOL,NROW,START_LAT,START_LON,D_LAT,D_LON
        write(19,15)NCOL,NROW,START_LAT,START_LON,D_LAT,D_LON
c    display header :
c      no. of columns & rows, starting latitude & longitude,
c      increment of latitude and longitude
        write(*,*)'FILE HEADER  ::: '
        write(*,*)'COL=',NCOL,',            ROWS=',NROW
   write(*,*)'ST_LAT=',START_LAT, ',      ST_LON=',START_LON
        write(*,*)'Delta_LAT=',D_LAT,', Delta_LON=',D_LON
c      format of input file header
15     FORMAT(I5,1X,I5,F6.3,1X,F6.3,1X,F10.8,1X,F10.8)
        write(*,*)'HEADER reading completed'
        write(*,*)'=== ========================== ==='
296
c      ccc
        ALLOCATE (CIGBP_CLASS(NCOL,NROW))
        write(*,*)'ALLOCATE'
        write(*,*)'READING FULL DATA ...'
```

```
c      reading remaining data : lulc/surface types
       read(10,50,REC=2) ((CIGBP_CLASS(IC,IR),IC=1,NCOL),IR=1,NROW)
          write(*,*)'=== ========================== ==='
c      test and display some information’s, while reading...
c         write(*,*)'REC=',REC
          ic_2=IC-2
          ir_2=IR-2
c         test
    write(*,*)'IC=',IC,', IR=',IR,', data=', CIGBP_CLASS(ic_2,ir_2)
c   format of lulc file after header part
50        FORMAT(2((216(100A1))))
c    other test formats
60   FORMAT(A1,1X)
70   format(A1)
80   FORMAT(i2,1X)
c   test/display read data
          write(*,*)'=== ========================== ==='
          write(*,*)'Array read completed'
          write(*,*)'NROW = ',NROW,', NCOL = ',NCOL
          write(*,*)'Test = data(1,  1)      : ',CIGBP_CLASS(1, 1)
          write(*,*)'Test = data(1,NROW)     : ',CIGBP_CLASS(1,NROW)
          write(*,*)'Test = data(NCOL,NROW)  : ',CIGBP_CLASS(NCOL,NROW)
          write(*,*)'Test = data(NCOL,1)     : ',CIGBP_CLASS(NCOL,1)
          write(*,*)'=== ========================== ==='
          write(*,*)'Display array : testing (3x3) ...'
          write(*,*)'=== ========================== ==='
C
    do i=1,3
          do j=1,3
          write(*,*)CIGBP_CLASS(i,j)
       end do
          write(*,*)'Next column :'
       end do
          write(*,*)'=== ========================== ==='
          write(*,*)' Writing to ASCII file (LULC):lc_ascii.out '
ccccccccccccccccccccccccccccc write to an ascii file
ccc     output variable : CIGBP_CLASS(NCOL,NROW)
          do jrow=1,NROW
           write(20,*)(CIGBP_CLASS(icol,jrow),icol=1,NCOL)
       end do
cccccccccccccccc write lat. & lon.
          write(*,*)'===== ========================= ======='
          write(*,*)'Output lat_ascii.out file (Latitude)...'
ccc     latitude
```

```
           a_lat=START_LAT
           do jrow=1,NROW
     write(21,*)a_lat
           a_lat=a_lat-D_LAT
           end do
           write(*,*)'===== ========================== ======='
           write(*,*)'Output lon_ascii.out file (Longitude)...'
ccc     longitude
           a_lon=START_LON
           do icol=1,NCOL
     write(22,*)a_lon
           a_lon=a_lon+D_LON
           end do
cccccccccccccccccccccccccccccc change to integer
           write(*,*)'Convert to corresponding integer...'
cccccccccccccccccccccccccccccccccccc print new way
       write(*,*)'Write Integer ascii file ...'
cccccccccccccccccccccccccccccccccccccccccccccccccccccc
            do jrow=1,NROW
cccccccccccccccccccccccccccccccccc
        do icol=1,NCOL-1
ccccccccccccccc
c       CIGBP_CLASS(icol,jrow)
        test=CIGBP_CLASS(icol,jrow)
        if (test.eq.'R') then
        write(23,80,advance="no")a(1)
        end if
        if(test.eq.'O') then
        write(23,80,advance="no")a(2)
        end if
            if(test.eq.'Q') then
        write(23,80,advance="no")a(3)
        end if
            if(test.eq.'A') then
        write(23,80,advance="no")a(4)
        end if
            if(test.eq.'B') then
        write(23,80,advance="no")a(5)
        end if
            if(test.eq.'C') then
        write(23,80,advance="no")a(6)
        end if
            if(test.eq.'D') then
        write(23,80,advance="no")a(7)
        end if
           if(test.eq.'E') then
        write(23,80,advance="no")a(8)
        end if
           if(test.eq.'F') then
        write(23,80,advance="no")a(9)
        end if
           if(test.eq.'G') then
        write(23,80,advance="no")a(10)
        end if
           if(test.eq.'H') then
        write(23,80,advance="no")a(11)
        end if
           if(test.eq.'I') then
        write(23,80,advance="no")a(12)
        end if
           if(test.eq.'J') then
        write(23,80,advance="no")a(13)
        end if
           if(test.eq.'K') then
        write(23,80,advance="no")a(14)
        end if
           if(test.eq.'L') then
        write(23,80,advance="no")a(15)
        end if
           if(test.eq.'M') then
        write(23,80,advance="no")a(16)
        end if
           if(test.eq.'N') then
        write(23,80,advance="no")a(17)
        end if
           if(test.eq.'P') then
        write(23,80,advance="no")a(18)
        end if
           if(test.eq.'Z') then
        write(23,80,advance="no")a(19)
        end if
cccccccccccccccccc
        do icol2=NCOL,NCOL
cccccccccccccccccccccccccccccccccc
        test=CIGBP_CLASS(icol2,jrow)
        if (test.eq.'R') then
        write(23,80)a(1)
        end if
```

```
if(test.eq.'O') then
write(23,80)a(2)
end if
    if(test.eq.'Q') then
write(23,80)a(3)
end if
    if(test.eq.'A') then
write(23,80)a(4)
end if
    if(test.eq.'B') then
write(23,80)a(5)
end if
    if(test.eq.'C') then
write(23,80)a(6)
end if
    if(test.eq.'D') then
write(23,80)a(7)
end if
    if(test.eq.'E') then
write(23,80)a(8)
end if
    if(test.eq.'F') then
write(23,80)a(9)
end if
    if(test.eq.'G') then
write(23,80)a(10)
end if
    if(test.eq.'H') then
write(23,80)a(11)
end if
    if(test.eq.'I') then
write(23,80)a(12)
end if
    if(test.eq.'J') then
write(23,80)a(13)
end if
    if(test.eq.'K') then
write(23,80)a(14)
end if
    if(test.eq.'L') then
write(23,80)a(15)
end if
    if(test.eq.'M') then
write(23,80)a(16)
```

```
      end if
         if(test.eq.'N') then
      write(23,80)a(17)
      end if
         if(test.eq.'P') then
      write(23,80)a(18)
      end if
         if(test.eq.'Z') then
      write(23,80,advance="no")a(19)
      end if
ccccccccccccccccccccccccccccc
      end do
         end do
ccccccccccccccccccccccccccccccccccccccccccccccccccccc
        write (*,*)'COL=',icol2,'  ROWS=',jrow
cccccccccccccccccccc CLOSE ALL FILES HERE
   close(10)
        close(20)
        close(21)
        close(22)
        close(23)
ccccccccccccccccccccc
        write(*,*)'===== ============   END  ============= ====='
        stop
        end
```

Appendix-2

1. Conversion to binary file

```
C   read intger ASCII file and then out grd(binary)
          CHARACTER(LEN=1), ALLOCATABLE :: CIGBP_CLASS(:,:)
          INTEGER :: int_out(43200,21600)
     CHARACTER(LEN=1) EG
          write(*,*)' ::: Prog. to out grd file :::: '
          OPEN(23,FILE='lc_int_ascii.out',FORM='FORMATTED',status='old')
          NROW=21600
          NCOL=43200
     WRITE(*,*)'TRY TO OPEN GRD/bin FILE...'
          OPEN(24,FILE='lc_int.bin',ACCESS='sequential',
   1 FORM='UNFORMATTED')
c   read from int file
          write(*,*)'Reading input file ...'
c   do icol=1,NCOL
          do jrow=1,NROW
          read(23,*)(int_out(icol,jrow),icol=1,NCOL)
          end do
          write(*,*)'int_out(1,1)=',int_out(1,1)
          write(*,*)'int_out(1000,1000)=',int_out(1000,1000)
          write(*,*)'int_out(43200,21600)=',int_out(43200,21600)
c      test
          write(*,*)'Writing grd/bin file ...'
          WRITE(24)((int_out(icol,jrow),icol=1,NCOL),jrow=1,NROW)
c   close all files
          close(23)
          close(24)
          write(*,*)'********** End of prog. **********'
          stop
          end
```

2. Convert binary to netCDF

```
C    read integer binary file and then output netCDF file.
c    implicit none
     include 'netcdf.inc'
C    This is the name of the data file we will create.
     character*(*) FILE_NAME
     parameter (FILE_NAME='igbp_lulc.nc')
C    We are writing 2D data, a 6 x 12 grid.
     integer NDIMS
     parameter (NDIMS=2)
```

```
      integer NX, NY
      parameter (NX = 21600, NY = 43200)
      integer ncid, varid, dimids(NDIMS),varid2,varid3
      integer x_dimid, y_dimid
C     This is the data array we will write.
      integer x, y, retval
cccccccc from previous file
          CHARACTER(LEN=1), ALLOCATABLE :: CIGBP_CLASS(:,:)
          INTEGER :: int_out(43200,21600)
      CHARACTER(LEN=1) EG
      real, ALLOCATABLE :: a_loni(:),a_lati(:)
      real :: a_lat,a_lon
      write(*,*)'Reading ascii file...'
          OPEN(23,FILE='lc_int_ascii.out',FORM='FORMATTED',status='old')
          NROW=21600
          NCOL=43200
          do jrow=1,NROW
          read(23,*)(int_out(icol,jrow),icol=1,NCOL)
          end do
c80      FORMAT(i2,1X)
     write(*,*)'Reading ascii file completed: Check values ...'
          write(*,*)'int_out(1,1)=',int_out(1,1)
          write(*,*)'int_out(1000,1000)=',int_out(1000,1000)
          write(*,*)'int_out(43200,21600)=',int_out(43200,21600)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
c       next write out to nc file
        write(*,*)'define nc file details...'
CCCCCCCCCCCC next netcdf  CCCCCCCC
c        NX=21600
c        NY=43200
C      overwrite this file, if it already exists.
       retval = nf_create(FILE_NAME, NF_CLOBBER, ncid)
       if (retval .ne. nf_noerr) call handle_err(retval)
C      Define the dimensions. netCDF will hand back an ID for each.
       retval = nf_def_dim(ncid, "long", NX, x_dimid)
       if (retval .ne. nf_noerr) call handle_err(retval)
       retval = nf_def_dim(ncid, "lati", NY, y_dimid)
       if (retval .ne. nf_noerr) call handle_err(retval)
C      The dimids array is used to pass the IDs of the dimensions of
C      the variables.
       dimids(2) = x_dimid
       dimids(1) = y_dimid
C      Define the variable. The type of the variable in this case is
C      NF_INT (4-byte integer).
```

```
      retval = nf_def_var(ncid, "lulc", NF_INT, NDIMS, dimids, varid)
c       retval = nf_def_var(ncid, "long", NF_REAL, 1, x_dimid, varid2)
c       retval = nf_def_var(ncid, "lat", NF_REAL, 1, y_dimid, varid3)
      if (retval .ne. nf_noerr) call handle_err(retval)
C      End define mode. This tells netCDF we are done defining metadata.
      retval = nf_enddef(ncid)
      if (retval .ne. nf_noerr) call handle_err(retval)
C      Write the pretend data to the file. Although netCDF supports
C      reading and writing subsets of data
      write(*,*)'writing nc file ...'
      retval = nf_put_var_int(ncid, varid, int_out)
      if (retval .ne. nf_noerr) call handle_err(retval)
C      Close the file.
      retval = nf_close(ncid)
      if (retval .ne. nf_noerr) call handle_err(retval)
      print *,'*** SUCCESS writing example file igbp_lulc.nc !!!'
      write(*,*)'open this file then add x (long) & y ...'
      end
      subroutine handle_err(errcode)
      implicit none
      include 'netcdf.inc'
      integer errcode
      print *, 'Error: ', nf_strerror(errcode)
      close(23)
      stop 2
      end
```

3. Add latitude, longitude variables to netCDF file

```
C     Adding latitude/longitude variables in to netcdf file.
c     implicit none
      include 'netcdf.inc'
C     This is the name of the data file we will create.
      character*(*) FILE_NAME
      parameter (FILE_NAME='igbp_lulc_latlon.nc')
C     writing 2D data
      integer NDIMS
      parameter (NDIMS=2)
      integer NX, NY
      parameter (NX = 21600, NY = 43200)
C     When we create netCDF files, variables and dimensions,
      integer ncid, varid, dimids(NDIMS),varid2,varid3
      integer x_dimid, y_dimid
C     This is the data array we will write.
      integer x, y, retval
```

```
cccccccc
     real *2, ALLOCATABLE :: a_loni(:),a_lati(:)
     real :: a_lat,a_lon
            NROW=21600
           NCOL=43200
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
ccc    latitude
           a_lat=90.000000
           sum1=0
           do jrow=1,NROW
    a_lati(jrow)=a_lat
           a_lat=a_lat-0.00833333
           end do
           write(*,*)'Lat array ready now ...'
           write(*,*)'Output lon_ascii.out file (Longitude)...'
ccc  longitude
           a_lon=0.0000000
           do icol=1,NCOL
    a_loni(icol)=a_lon
           a_lon=a_lon+0.00833333
      end do
           write(*,*)'Longitude array ready now ...'
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
c      next write out to nc file
       write(*,*)'define nc file details...'
CCCCCCCCCCCC
c       NX=21600
c       NY=43200
C     overwrite this file, if it already exists.
      retval = nf_open(FILE_NAME, NF_WRITE, ncid)
      if (retval .ne. nf_noerr) call handle_err(retval)
C     Define the dimensions.
      retval = nf_def_dim(ncid, "long", NX, x_dimid)
      if (retval .ne. nf_noerr) call handle_err(retval)
      retval = nf_def_dim(ncid, "lati", NY, y_dimid)
      if (retval .ne. nf_noerr) call handle_err(retval)
      dimids(2) = x_dimid
      dimids(1) = y_dimid
C     Define the variable.
cc     retval = nf_def_var(ncid, "lulc", NF_INT, NDIMS, dimids, varid)
      retval = nf_def_var(ncid, "long", NF_REAL, 1, x_dimid, varid2)
      retval = nf_def_var(ncid, "lati", NF_REAL, 1, y_dimid, varid3)
      if (retval .ne. nf_noerr) call handle_err(retval)
C     End define mode.
```

```
      write(*,*)'writing nc file ...'
      retval = nf_put_var_real(ncid, varid2, a_loni)
      if (retval .ne. nf_noerr) call handle_err(retval)
      write(*,*)'Processing of x over.. next y'
CCCCCCCCCCCCCCC added new
      retval = nf_put_var_real(ncid, varid3, a_lati)
      if (retval .ne. nf_noerr) call handle_err(retval)
CCCCCCCCCCCCCCCCCCCCCCCCCC
      retval = nf_close(ncid)
      if (retval .ne. nf_noerr) call handle_err(retval)
      print *,'*** SUCCESS adding of x & y in igbp_lulc.nc !!!'
      write(*,*)'Check it...'
      end
      subroutine handle_err(errcode)
      implicit none
      include 'netcdf.inc'
      integer errcode
      print *, 'Error: ', nf_strerror(errcode)
      stop 2
      end
```

Appendix -3

```
c     sample program to merge the data sets
      implicit none
      include 'netcdf.inc'
C     This is the name of the data file we will read.
      character*(*) FILE_NAME
      parameter (FILE_NAME='nrsc.nc')
C     We are reading 2D data
      integer :: NX, NY
      parameter (NX = 4800, NY = 4800)
      integer :: data_in(NY, NX)
C     This will be the netCDF ID for the file and data variable.
      integer :: ncid, varid
C     Loop indexes, and error handling.
      integer :: x, y, retval
c     define
       integer :: N,istart,iend,jstart,jend,i,j,ic
       integer :: ic1,jc1,is,js
       integer :: A(36)
       integer :: data_in2(43200,21600)  !Table to be sorted
       integer :: MAX_VALUE,TMODE1
      integer :: NDIMS, NX1, NY1
      parameter (NDIMS=2)
      integer x_dimid, y_dimid, dimids(NDIMS)
C     Open the file. NF_NOWRITE tells netCDF we want read-only access to
C     the file.
      retval = nf_open(FILE_NAME, NF_NOWRITE, ncid)
      if (retval .ne. nf_noerr) call handle_err(retval)
C     Get the varid of the data variable.
      retval = nf_inq_varid(ncid, 'india_lulc', varid)
      if (retval .ne. nf_noerr) call handle_err(retval)
C     Read the data.
      retval = nf_get_var_int(ncid, varid, data_in)
      if (retval .ne. nf_noerr) call handle_err(retval)
C     Close the file, freeing all resources.
      retval = nf_close(ncid)
      if (retval .ne. nf_noerr) call handle_err(retval)
      write(*,*)'data_in(1,1)=',data_in(1,1)
      write(*,*)'data_in(4800,1)=',data_in(4800,1)
      write(*,*)'data_in(1,4800)=',data_in(1,4800)
      write(*,*)'data_in(4800,4800)=',data_in(4800,4800)
      print *,'*** SUCCESS reading example file ', FILE_NAME, '!'
```

```
CCCCCCCCCC add main prog here  cccccccccccccccccccccccccccccc
      retval = nf_open("igbp_lulc.nc", NF_WRITE, ncid)
      if (retval .ne. nf_noerr) call handle_err(retval)
C     Get the varid of the data variable, based on its name.
      retval = nf_inq_varid(ncid, 'lulc', varid)
      if (retval .ne. nf_noerr) call handle_err(retval)
C     Read the data.
      retval = nf_get_var_int(ncid, varid, data_in2)
      if (retval .ne. nf_noerr) call handle_err(retval)
ccccccc check…
      write(*,*)'data_in2(1,1)=',data_in2(1,1)
      write(*,*)'data_in2(43200,1)=',data_in2(43200,1)
      write(*,*)'data_in2(1,21600)=',data_in2(1,21600)
      write(*,*)'data_in2(43200,21600)=',data_in2(43200,21600)
      print *,'*** SUCCESS reading example file :2 ', FILE_NAME, '!'
C     Close the file, freeing all resources.
      retval = nf_close(ncid)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCC merge it now
      is=7201
      do i=1,4800
      js=6001
      do j=1,4800
      data_in2(is,js)=data_in(j,i)
      js=js+1
      end do
      is=is+1
      end do
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC write
CCCCCCCCCCCCCCCCCCC
              NX1=21600
       NY1=43200
       retval = nf_create('abc.nc', NF_CLOBBER, ncid)
       if (retval .ne. nf_noerr) call handle_err(retval)
       retval = nf_def_dim(ncid, "long", NX1, x_dimid)
       if (retval .ne. nf_noerr) call handle_err(retval)
       retval = nf_def_dim(ncid, "lati", NY1, y_dimid)
       if (retval .ne. nf_noerr) call handle_err(retval)
       dimids(2) = x_dimid
       dimids(1) = y_dimid
       retval = nf_def_var(ncid, "lc", NF_INT, NDIMS, dimids, varid)
       if (retval .ne. nf_noerr) call handle_err(retval)
       retval = nf_enddef(ncid)
       if (retval .ne. nf_noerr) call handle_err(retval)
       retval = nf_put_var_int(ncid, varid, data_in2)
       if (retval .ne. nf_noerr) call handle_err(retval)
       retval = nf_close(ncid)
       if (retval .ne. nf_noerr) call handle_err(retval)
       print *,'*** SUCCESS writing : merged data !!! now test ..'
      retval = nf_close(ncid)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
cccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      stop
      end
CCCCCCCCCCCCCCCCCCC sub
      subroutine handle_err(errcode)
      implicit none
      include 'netcdf.inc'
      integer errcode
      print *, 'Error: ', nf_strerror(errcode)
      stop 2
      end
```

Appendix-4

```
c    program to convert integer back to character ASCII file
     implicit none
     include 'netcdf.inc'
C    This is the name of the data file we will read.
     character*(*) FILE_NAME
     parameter (FILE_NAME='abc.nc')
C    We are reading 2D data, a 6 x 12 grid.
     integer NX, NY
     parameter (NX = 21600, NY = 43200)
     integer :: data_in(NY, NX)
C    This will be the netCDF ID for the file and data variable.
     integer :: ncid, varid
C    Loop indexes, and error handling.
     integer :: x, y, retval,IR
c    define
      integer :: N,istart,iend,jstart,jend,i,j,ic
      integer :: ic1,jc1,is,js,test,icol,jrow,NCOL,NROW,icol2
      integer :: A(36)
c     integer :: data_in2(43200,21600)  !Table to be sorted
      integer :: MAX_VALUE,TMODE1
     integer :: NDIMS, NX1, NY1
     parameter (NDIMS=2)
     real :: START_LAT,START_LON,D_LAT,D_LON
     integer x_dimid, y_dimid, dimids(NDIMS)
          CHARACTER(LEN=1), ALLOCATABLE :: CIGBP_CLASS(:,:)
C    Open the file. NF_NOWRITE tells netCDF we want read-only access to
C    the file.
     retval = nf_open(FILE_NAME, NF_NOWRITE, ncid)
     if (retval .ne. nf_noerr) call handle_err(retval)
C    Get the varid of the data variable, based on its name.
     retval = nf_inq_varid(ncid, 'lc', varid)
     if (retval .ne. nf_noerr) call handle_err(retval)
C    Read the data.
     retval = nf_get_var_int(ncid, varid, data_in)
     if (retval .ne. nf_noerr) call handle_err(retval)
C    Close the file, freeing all resources.
     retval = nf_close(ncid)
     if (retval .ne. nf_noerr) call handle_err(retval)
     write(*,*)'data_in(1,1)=',data_in(1,1)
     write(*,*)'data_in(43200,1)=',data_in(43200,1)
     write(*,*)'data_in(1,21600)=',data_in(1,21600)
```

```
      write(*,*)'data_in(43200,21600)=',data_in(43200,21600)
      print *,'*** SUCCESS reading example file ', FILE_NAME, '!'
CCCCCC add ::: next part here  cccccccccccccccccccccccccccccc
c     readfile : 'lc'
           OPEN(10,FILE='lc',FORM='FORMATTED',ACCESS='DIRECT',RECL=43200)
c     read input file
           read(10,15,REC=1)NCOL,NROW,START_LAT,START_LON,D_LAT,D_LON
           write(*,15)NCOL,NROW,START_LAT,START_LON,D_LAT,D_LON
15   FORMAT(I5,1X,I5,F6.3,1X,F6.3,1X,F10.8,1X,F10.8)
           write(*,*)'HEADER reading completed'
ccccccccccccc next convert integer to character & write to File
           OPEN(24,FILE='lc_nrsc2.out',FORM='FORMATTED',status='new')
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccc
80      format(A1)
            do jrow=1,NROW
cccccccccccccccccccccccccccccccccc
        do icol=1,NCOL-1
ccccccccccccc
        test=data_in(icol,jrow)
        if (test.eq.1) then
        write(24,80)'A'
        end if
        if(test.eq.2) then
        write(24,80)'B'
        end if
            if(test.eq.3) then
        write(24,80)'C'
        end if
            if(test.eq.4) then
        write(24,80)'D'
        end if
            if(test.eq.5) then
        write(24,80)'E'
        end if
            if(test.eq.6) then
        write(24,80)'F'
        end if
            if(test.eq.7) then
        write(24,80)'G'
        end if
            if(test.eq.8) then
        write(24,80)'H'
        end if
```

```
            if(test.eq.9) then
        write(24,80)'I'
        end if
            if(test.eq.10) then
        write(24,80)'J'
        end if
            if(test.eq.11) then
        write(24,80)'K'
        end if
            if(test.eq.12) then
        write(24,80)'L'
        end if
            if(test.eq.13) then
        write(24,80)'M'
        end if
            if(test.eq.14) then
        write(24,80)'N'
        end if
            if(test.eq.15) then
        write(24,80)'O'
        end if
            if(test.eq.16) then
        write(24,80)'P'
        end if
            if(test.eq.17) then
        write(24,80)'Q'
        end if
            if(test.eq.18) then
        write(24,80)'R'
        end if
            if(test.eq.19) then
        write(24,80)'Z'
        end if
            end do
c
cccccccccccccccccccccccccccccccc
        do icol2=NCOL,NCOL
        test=data_in(icol2,jrow)
        if (test.eq.1) then
        write(24,80)'A'
        end if
        if(test.eq.2) then
        write(24,80)'B'
        end if
```

```
    if(test.eq.3) then
write(24,80)'C'
end if
    if(test.eq.4) then
write(24,80)'D'
end if
    if(test.eq.5) then
write(24,80)'E'
end if
    if(test.eq.6) then
write(24,80)'F'
end if
    if(test.eq.7) then
write(24,80)'G'
end if
    if(test.eq.8) then
write(24,80)'H'
end if
    if(test.eq.9) then
write(24,80)'I'
end if
    if(test.eq.10) then
write(24,80)'J'
end if
    if(test.eq.11) then
write(24,80)'K'
end if
    if(test.eq.12) then
write(24,80)'L'
end if
    if(test.eq.13) then
write(24,80)'M'
end if
    if(test.eq.14) then
write(24,80)'N'
end if
    if(test.eq.15) then
write(24,80)'O'
end if
    if(test.eq.16) then
write(24,80)'P'
end if
    if(test.eq.17) then
write(24,80)'Q'
```

```
        end if
            if(test.eq.18) then
        write(24,80)'R'
        end if
            if(test.eq.19) then
        write(24,80)'Z'
        end if
c
       end do
       end do
cccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
CCC  ccccccccccccc
          write(*,*)'=== ========================== ==='
          write(*,*)'Character array process completed ...'
          write(*,*)'NROW = ',NROW,', NCOL = ',NCOL
c         write(*,*)'Test = data(1,  1)      : ',CIGBP_CLASS(1, 1)
c         write(*,*)'Test = data(1,NROW)     : ',CIGBP_CLASS(1,NROW)
c         write(*,*)'Test = data(NCOL,NROW)  : ',CIGBP_CLASS(NCOL,NROW)
c         write(*,*)'Test = data(NCOL,1)     : ',CIGBP_CLASS(NCOL,1)
          write(*,*)'=== ========================== ==='
CCCCCCCCCCCCCCCCCCCCCCCC next write on the master file
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          close(10)
           close(24)
      stop
      end
CCCCCC  CCCCCCCCCCCC sub
      subroutine handle_err(errcode)
      implicit none
      include 'netcdf.inc'
      integer errcode
      print *, 'Error: ', nf_strerror(errcode)
      stop 2
      end
```

Appendix-5

```
cccccccc Convert the merged character ASCII file to original master file format
          CHARACTER(LEN=1), ALLOCATABLE :: CIGBP_CLASS(:,:)
          CHARACTER(LEN=1) EG
    write(*,*)'Reading ascii file...'
          OPEN(23,FILE='lc_nrsc2.out',FORM='FORMATTED',status='old')
          NROW=21600
          NCOL=43200
          ALLOCATE (CIGBP_CLASS(NCOL,NROW))
          do jrow=1,NROW
          read(23,80)(CIGBP_CLASS(icol,jrow),icol=1,NCOL)
    end do
    write(*,*)'Reading completed .... !!!'
80   format(A1)
    write(*,*)'Now, Check values ...'
          write(*,*)'CIGBP_CLASS(1,1)=',CIGBP_CLASS(1,1)
          write(*,*)'CIGBP_CLASS(1,21600)=',CIGBP_CLASS(1,21600)
          write(*,*)'CIGBP_CLASS(43200,1)=',CIGBP_CLASS(43200,1)
          write(*,*)'CIGBP_CLASS(43200,21600)=',CIGBP_CLASS(43200,21600)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      close(23)
ccccccccccccccccccccccccccccc
c      readfile : 'lc'
          OPEN(10,FILE='lc',FORM='FORMATTED',ACCESS='DIRECT',RECL=43200)
c      read input file
          read(10,15,REC=1)NCOL,NROW,START_LAT,START_LON,D_LAT,D_LON
          write(*,15)NCOL,NROW,START_LAT,START_LON,D_LAT,D_LON
15      FORMAT(I5,1X,I5,F6.3,1X,F6.3,1X,F10.8,1X,F10.8)
          write(*,*)'HEADER reading completed'
      close(10)
c      write data
c     declare file
          OPEN(11,FILE='ln',FORM='FORMATTED',ACCESS='DIRECT',RECL=43200)
      write(11,15,REC=1)NCOL,NROW,START_LAT,START_LON,D_LAT,D_LON
c      write remaining data : lulc/surface types
      write(11,50,REC=2)((CIGBP_CLASS(IC,IR),IC=1,NCOL),IR=1,NROW)
c      format of lulc file after header part
50        FORMAT(2((216(100A1))))
      close(11)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCC
ccccccccccccccccccccccccccccc
    stop
                                              end
```

References

Cox, P.M., Betts, R.A., Bunton, C.B., Essery, R.L.H., Rowntree, P.R., Smith, J., 1999: The impact of new land surface physics on the GCM simulation of climate and climate sensitivity. Clim. Dyn. 15, 183–203.

Essery, R., Best, M., and Cox, P., 2001: MOSES 2.2 Technical Documentation, Hadley Centre Technical Note 30, Hadley Centre, MetOffice, Bracknell, UK.

Essery, R. and Clark, D., 2003: Developments in the MOSES 2 land surface model for PILPS 2e, Glob. Planet. Change, 38, 161–164.

Best et al., 2011: The Joint UK Land Environment Simulator (JULES), model description – Part 1: Energy and water fluxes, Geosci. Model Dev., 4, 677–699

Clark et al., 2011: The Joint UK Land Environment Simulator (JULES), model description – Part 2: Carbon fluxes and vegetation dynamics, Geosci. Model Dev., 4, 701–722

Rajagopal E.N. , G.R. Iyengar, John P. George, Munmun Das Gupta, Saji Mohandas, Renu Siddharth, Anjari Gupta, Manjusha Chourasia, V.S. Prasad, Aditi, Kuldeep Sharma and Amit Ashish, 2012: Implementation of Unified Model based Analysis-Forecast System at NCMRWF, Technical Report, NMRF/TR/2/2012, 46p**.**

Richards, 1931: Capillary conduction of liquids through porous mediums. *Physics* 1 (5): 318–333.

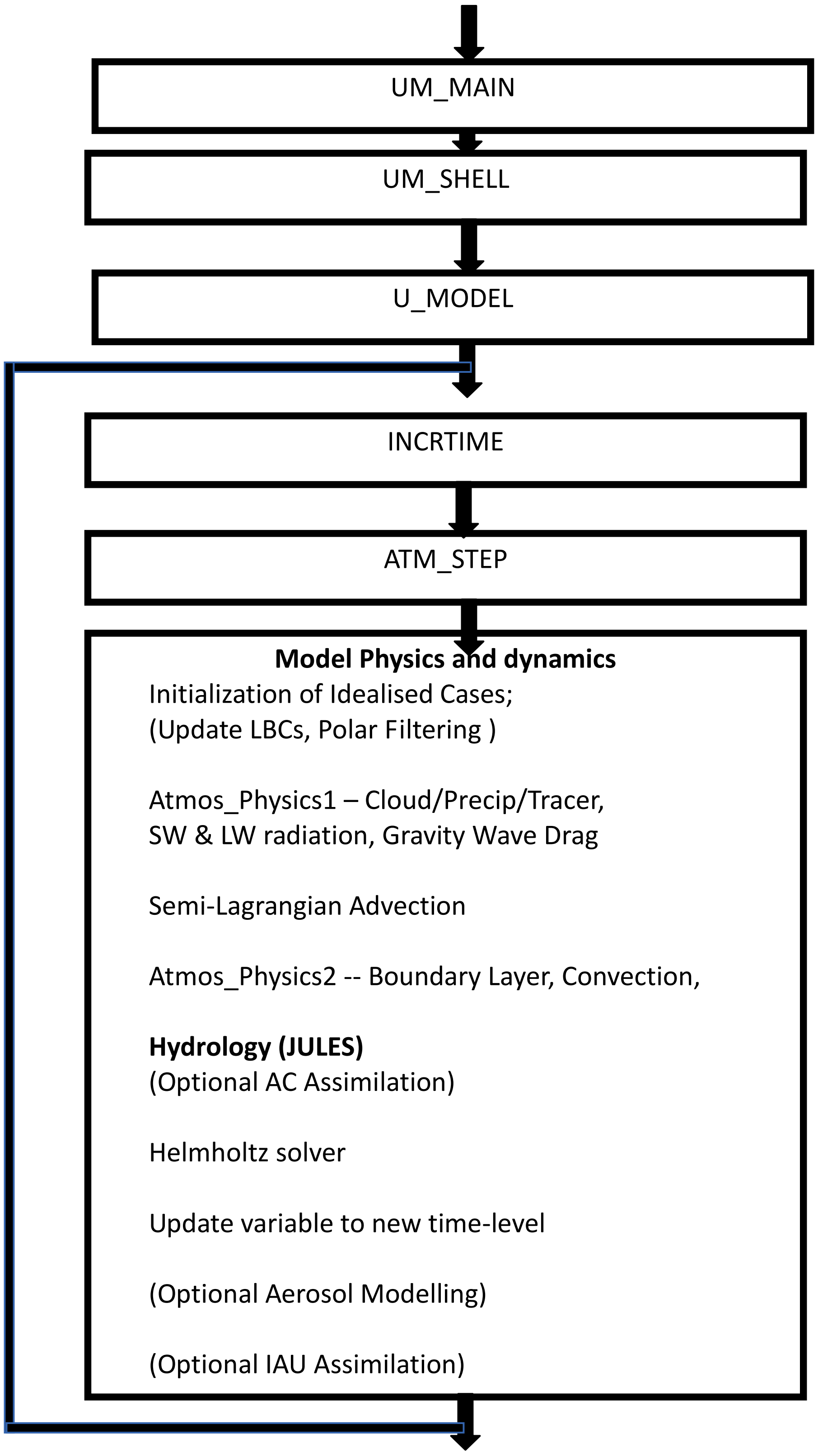

UM_MAIN
UM_SHELL
U_MODEL
INCRTIME
ATM_STEP
Model Physics and dynamics
Initialization of Idealised Cases;
(Update LBCs, Polar Filtering)
Atmos_Physics1 – Cloud/Precip/Tracer,
SW & LW radiation, Gravity Wave Drag
Semi-Lagrangian Advection
Atmos_Physics2 -- Boundary Layer, Convection,
Hydrology (JULES)
(Optional AC Assimilation)
Helmholtz solver
Update variable to new time-level
(Optional Aerosol Modelling)
(Optional IAU Assimilation)

Figure 1: Sequence of processes in the UM.

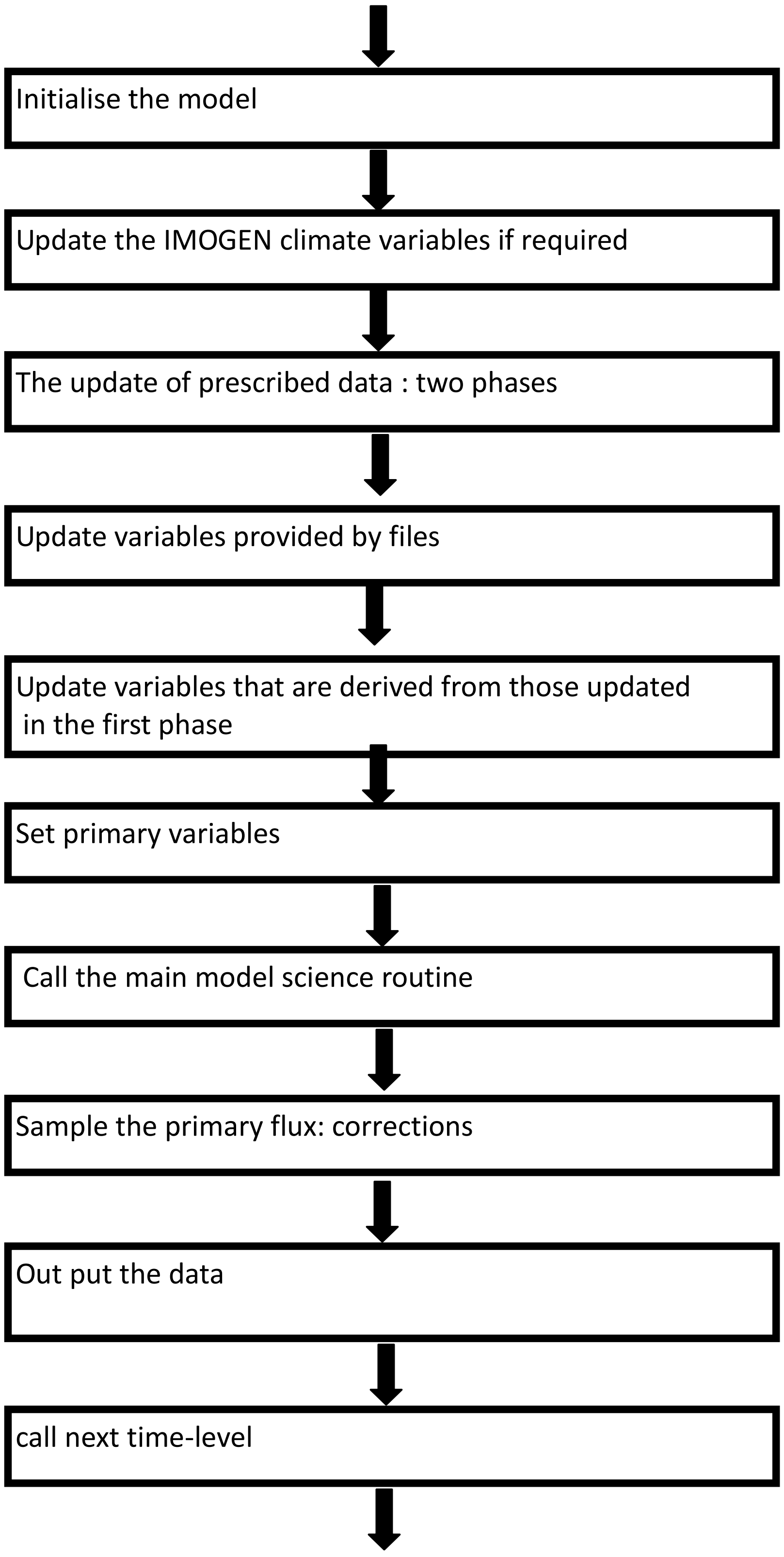
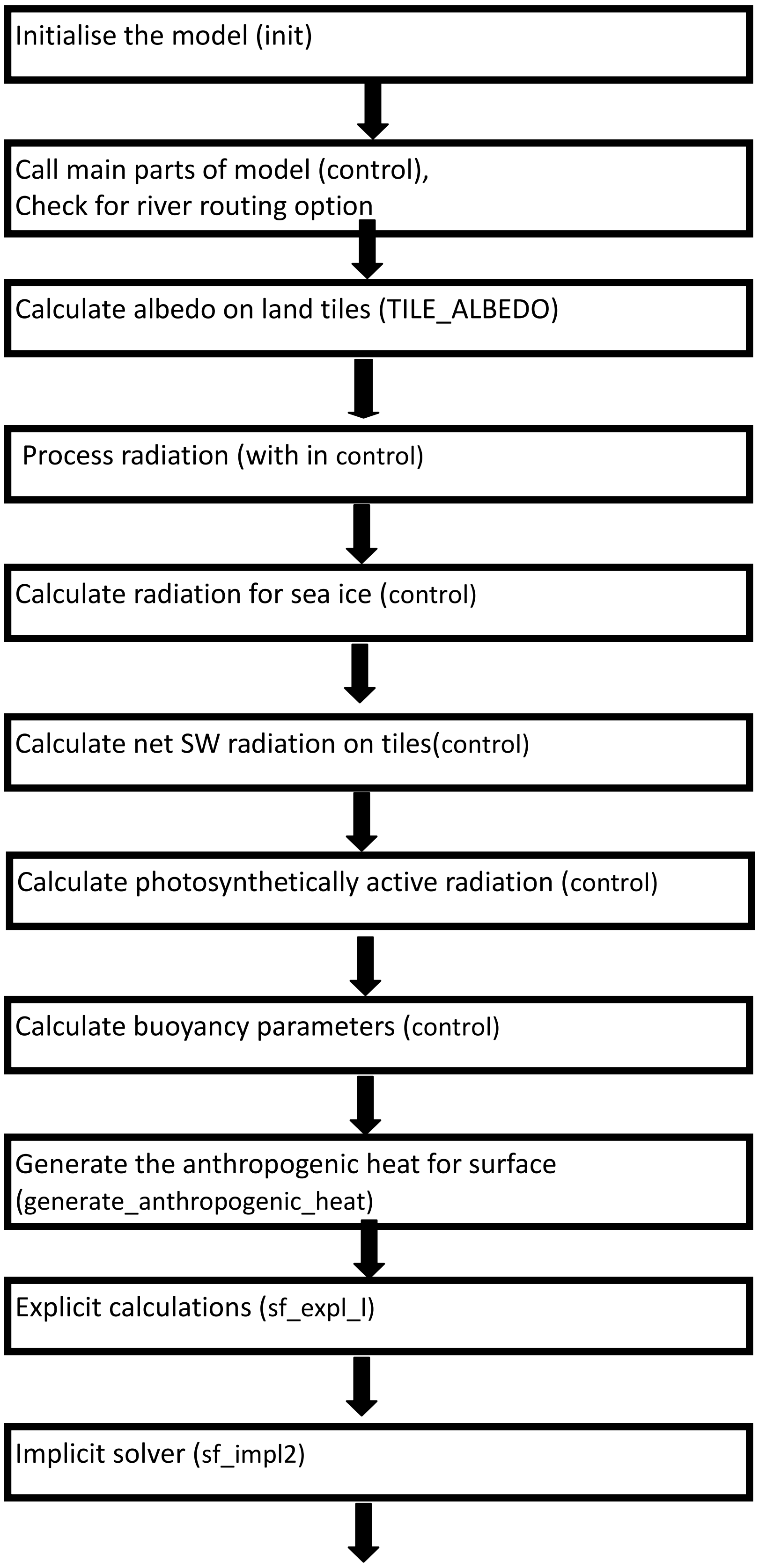

Initialise the model
Update the IMOGEN climate variables if required
The update of prescribed data : two phases
Update variables provided by files
Update variables that are derived from those updated in the first phase
Set primary variables
Call the main model science routine
Sample the primary flux: corrections
Out put the data
call next time-level

Figure 2*: Sequence of processes in JULES land surface scheme.*


Initialise the model (init)
Call main parts of model (control),
Check for river routing option
Calculate albedo on land tiles (TILE_ALBEDO)
Process radiation (with in control)
Calculate radiation for sea ice (control)
Calculate net SW radiation on tiles(control)
Calculate photosynthetically active radiation (control)
Calculate buoyancy parameters (control)
Generate the anthropogenic heat for surface
(generate_anthropogenic_heat)
Explicit calculations (sf_expl_l)
Implicit solver (sf_impl2)

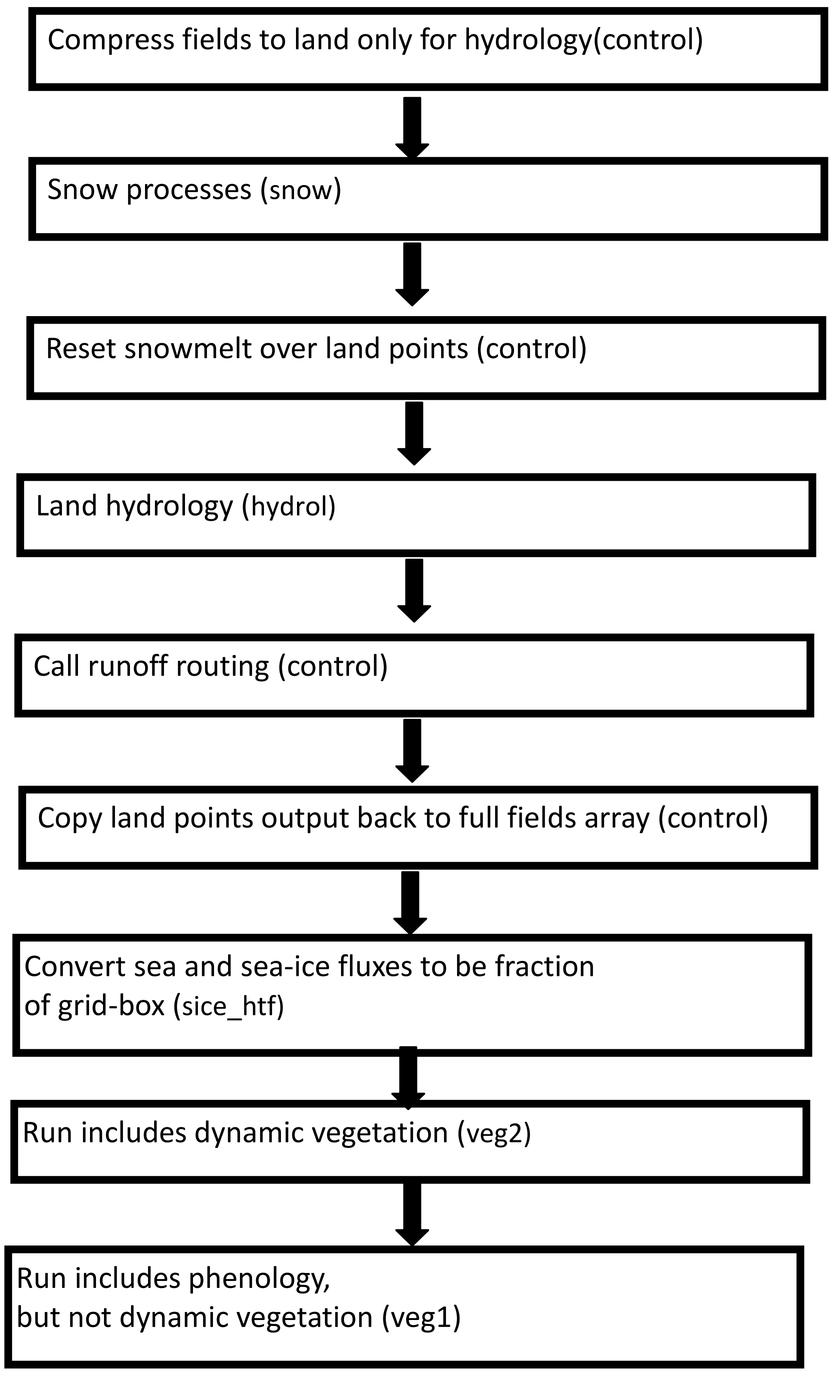

Compress fields to land only for hydrology(control)
Snow processes (snow)
Reset snowmelt over land points (control)
Land hydrology (hydrol)
Call runoff routing (control)
Copy land points output back to full fields array (control)
Convert sea and sea-ice fluxes to be fraction of grid-box (sice_htf)
Run includes dynamic vegetation (veg2)
Run includes phenology, but not dynamic vegetation (veg1)

Figure 3: Sequence of process in JULES control subroutine (module/subroutine name)

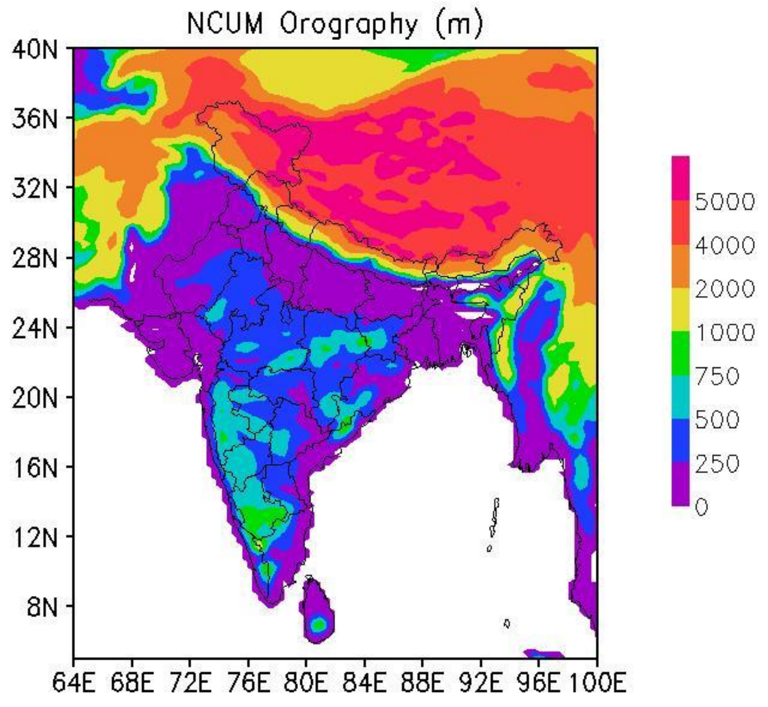


Figure 4: Mean orographic height (m) in NCUM over Indian region

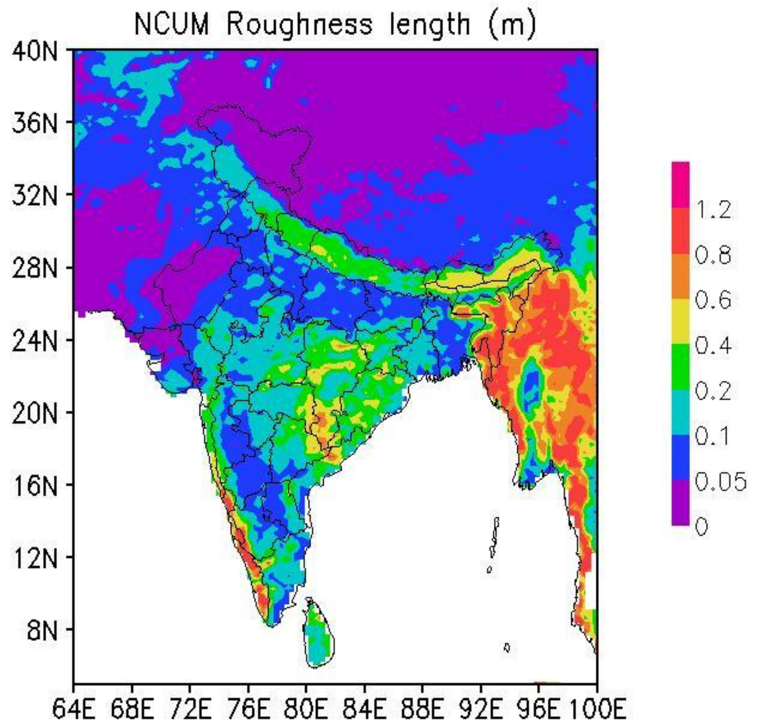

NCUM Roughness length (m)
40N
36N
32N
28N
24N
20N
16N
12N
8N
64E 68E 72E 76E 80E 84E 88E 92E 96E 100E
1.2
0.8
0.6
0.4
0.2
0.1
0.05
0

Figure 5: Roughness length (m) in NCUM

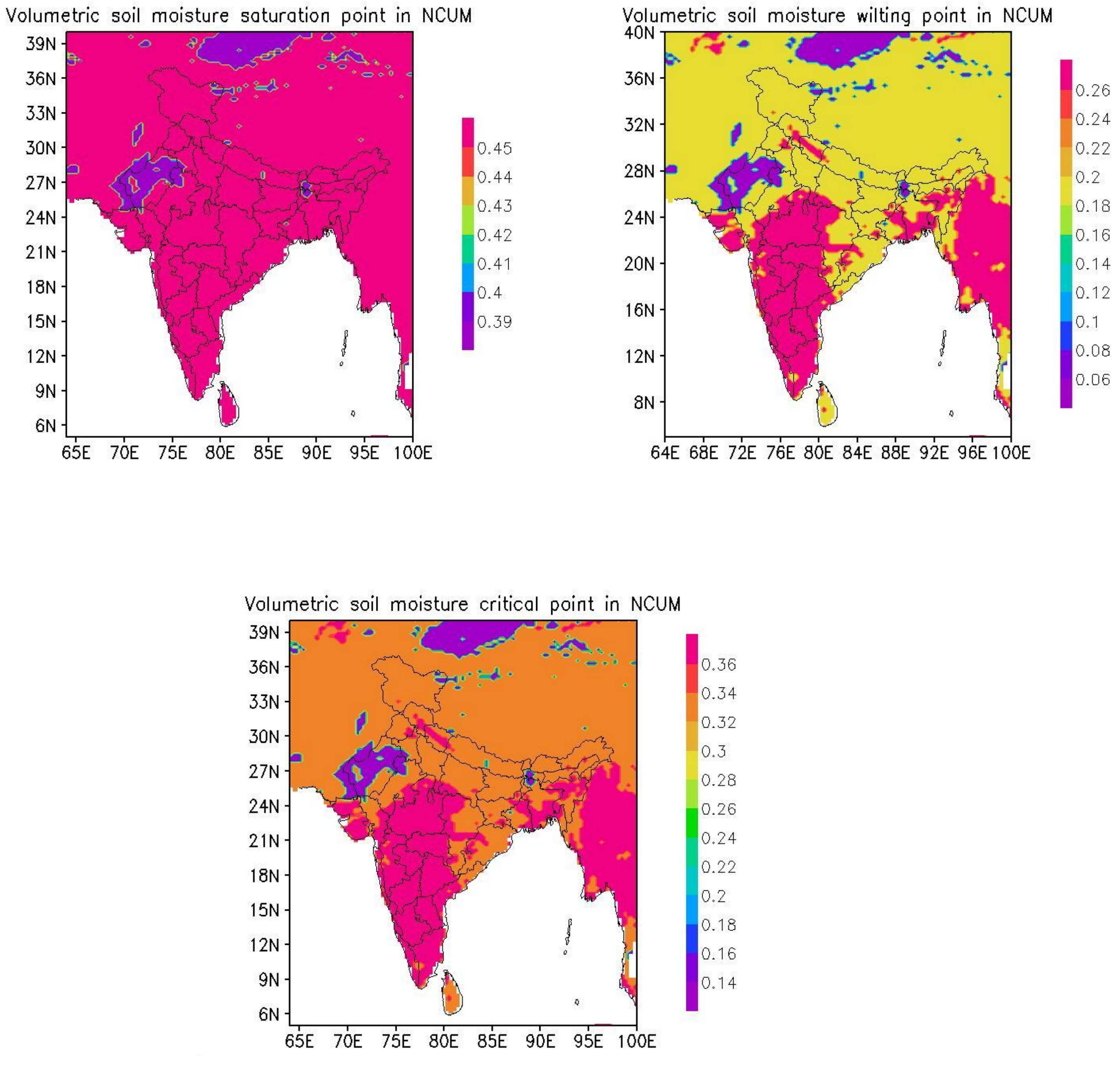

Volumetric soil moisture saturation point in NCUM
39N
36N
33N
30N
27N
24N
21N
18N
15N
12N
9N
6N
65E 70E 75E 80E 85E 90E 95E 100E
0.45
0.44
0.43
0.42
0.41
0.4
0.39
Volumetric soil moisture wilting point in NCUM
40N
36N
32N
28N
24N
20N
16N
12N
8N
64E 68E 72E 76E 80E 84E 88E 92E 96E 100E
0.26
0.24
0.22
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
Volumetric soil moisture critical point in NCUM
39N
36N
33N
30N
27N
24N
21N
18N
15N
12N
9N
6N
65E 70E 75E 80E 85E 90E 95E 100E
0.36
0.34
0.32
0.3
0.28
0.26
0.24
0.22
0.2
0.18
0.16
0.14

Figure 6: Volumetric soil moisture (m^3/m^3) (a) saturation point, (b) critical point and (c) wilting point in NCUM over Indian region

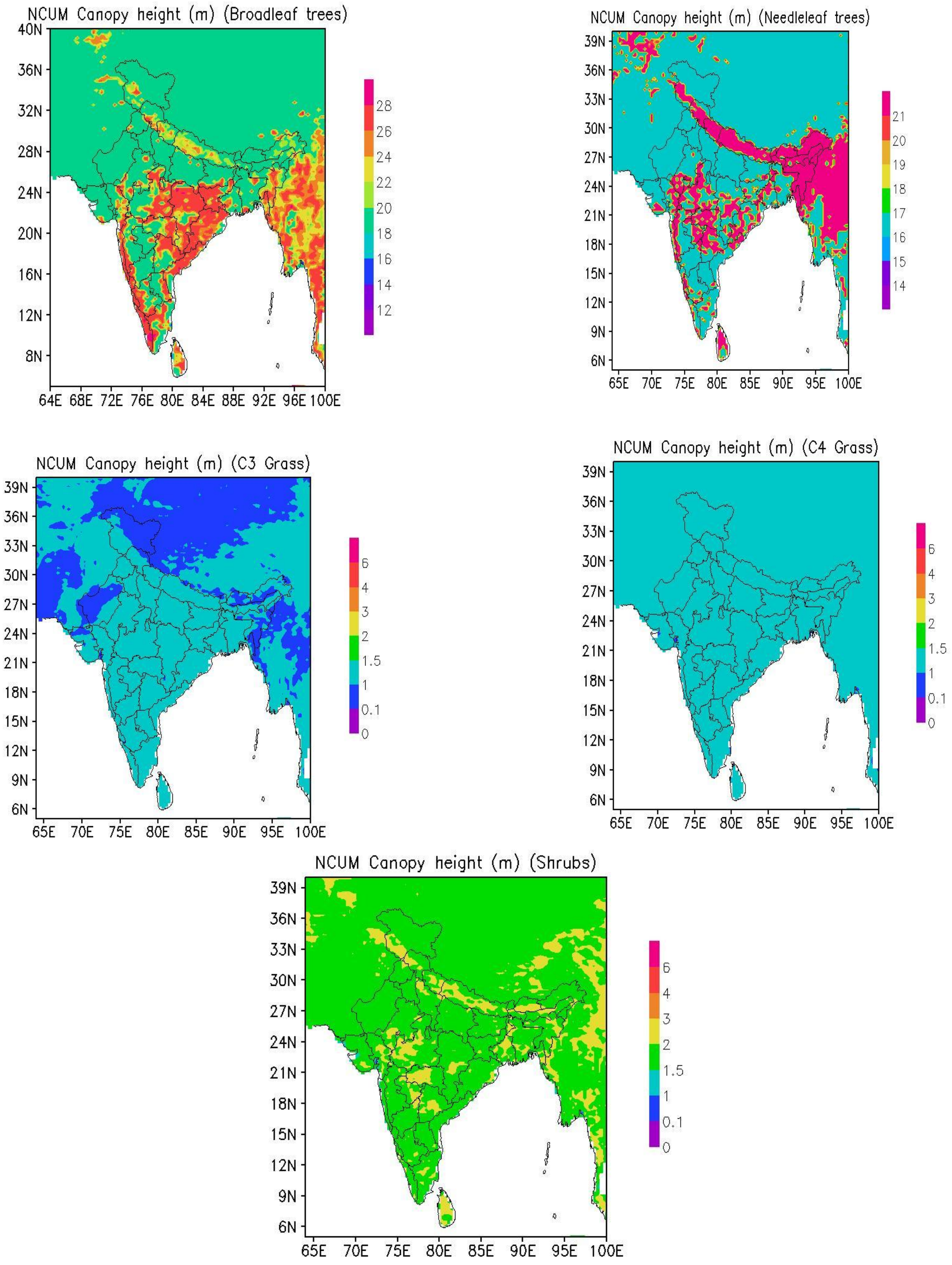

NCUM Canopy height (m) (Broadleaf trees)
12 14 16 18 20 22 24 26 28
NCUM Canopy height (m) (Needleleaf trees)
14 15 16 17 18 19 20 21
NCUM Canopy height (m) (C3 Grass)
0 0.1 1 1.5 2 3 4 6
NCUM Canopy height (m) (C4 Grass)
0 0.1 1 1.5 2 3 4 6
NCUM Canopy height (m) (Shrubs)
0 0.1 1 1.5 2 3 4 6
40N 36N 32N 28N 24N 20N 16N 12N 8N
64E 68E 72E 76E 80E 84E 88E 92E 96E 100E
39N 36N 33N 30N 27N 24N 21N 18N 15N 12N 9N 6N
65E 70E 75E 80E 85E 90E 95E 100E

Figure 7: Canopy height (m) used in NCUM over Indian region for different plant functions

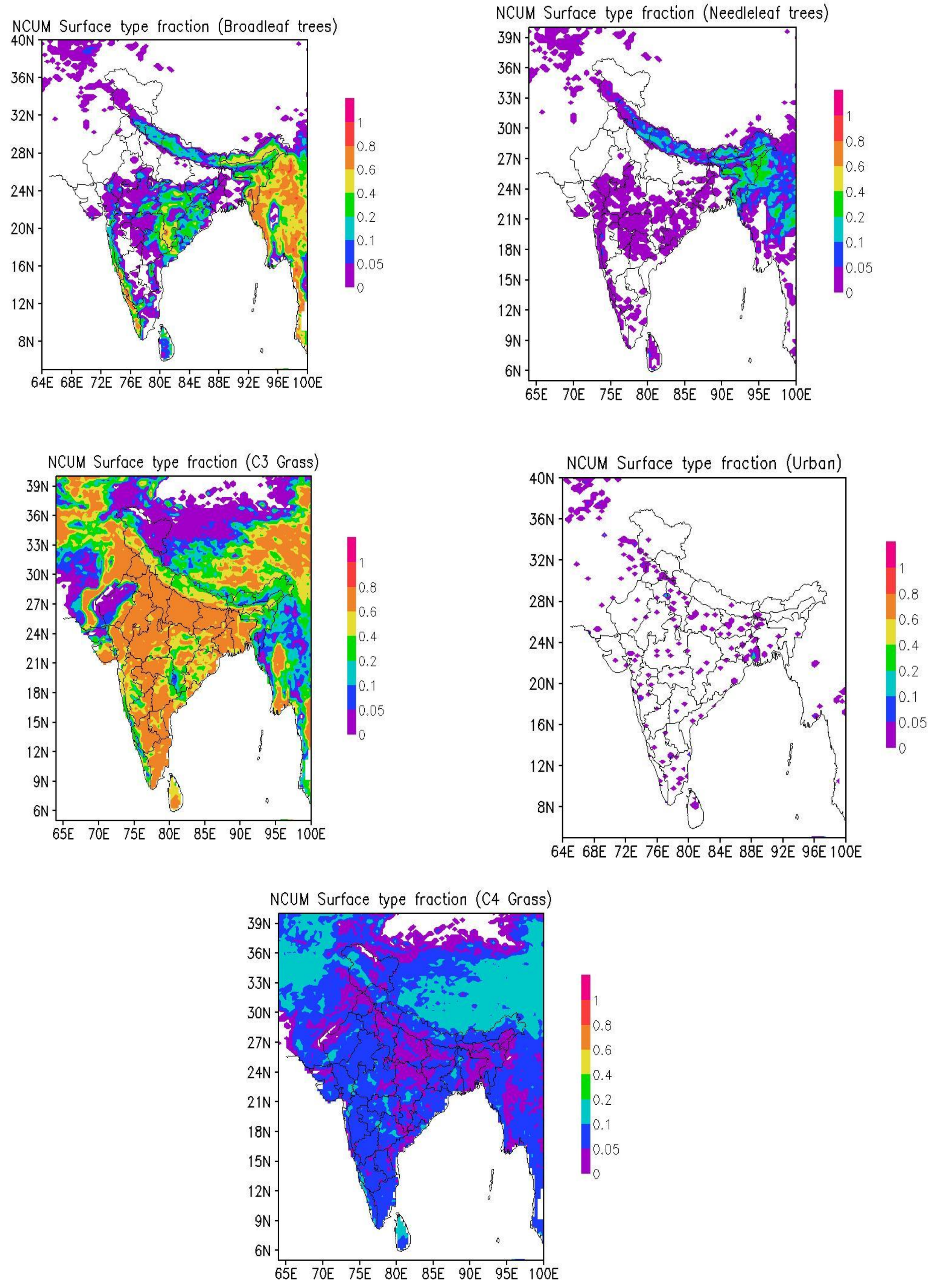
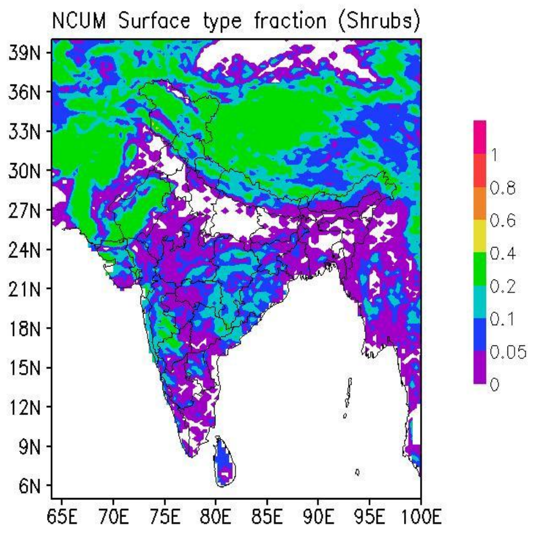
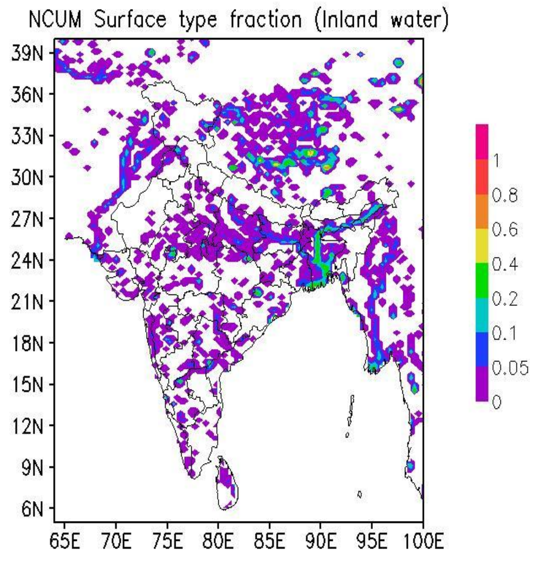
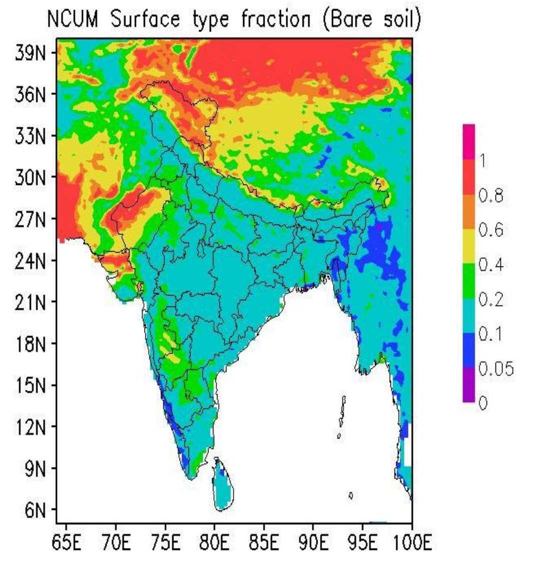
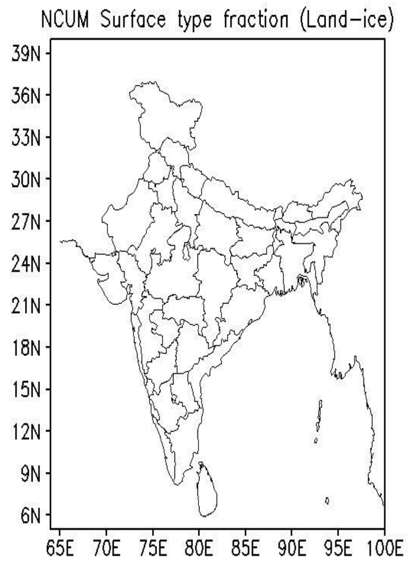

NCUM Surface type fraction (Broadleaf trees)
NCUM Surface type fraction (Needleleaf trees)
NCUM Surface type fraction (C3 Grass)
NCUM Surface type fraction (Urban)
NCUM Surface type fraction (C4 Grass)
0
0.05
0.1
0.2
0.4
0.6
0.8
1
6N
9N
12N
15N
18N
21N
24N
27N
30N
33N
36N
39N
40N
65E 70E 75E 80E 85E 90E 95E 100E
64E 68E 72E 76E 80E 84E 88E 92E 96E 100E

Figure 8: Surface type fraction in NCUM (Continued in next page)


NCUM Surface type fraction (Shrubs)
39N
36N
33N
30N
27N
24N
21N
18N
15N
12N
9N
6N
65E 70E 75E 80E 85E 90E 95E 100E
1
0.8
0.6
0.4
0.2
0.1
0.05
0


NCUM Surface type fraction (Inland water)
39N
36N
33N
30N
27N
24N
21N
18N
15N
12N
9N
6N
65E 70E 75E 80E 85E 90E 95E 100E
1
0.8
0.6
0.4
0.2
0.1
0.05
0


NCUM Surface type fraction (Bare soil)
39N
36N
33N
30N
27N
24N
21N
18N
15N
12N
9N
6N
65E 70E 75E 80E 85E 90E 95E 100E
1
0.8
0.6
0.4
0.2
0.1
0.05
0


NCUM Surface type fraction (Land–ice)
39N
36N
33N
30N
27N
24N
21N
18N
15N
12N
9N
6N
65E 70E 75E 80E 85E 90E 95E 100E

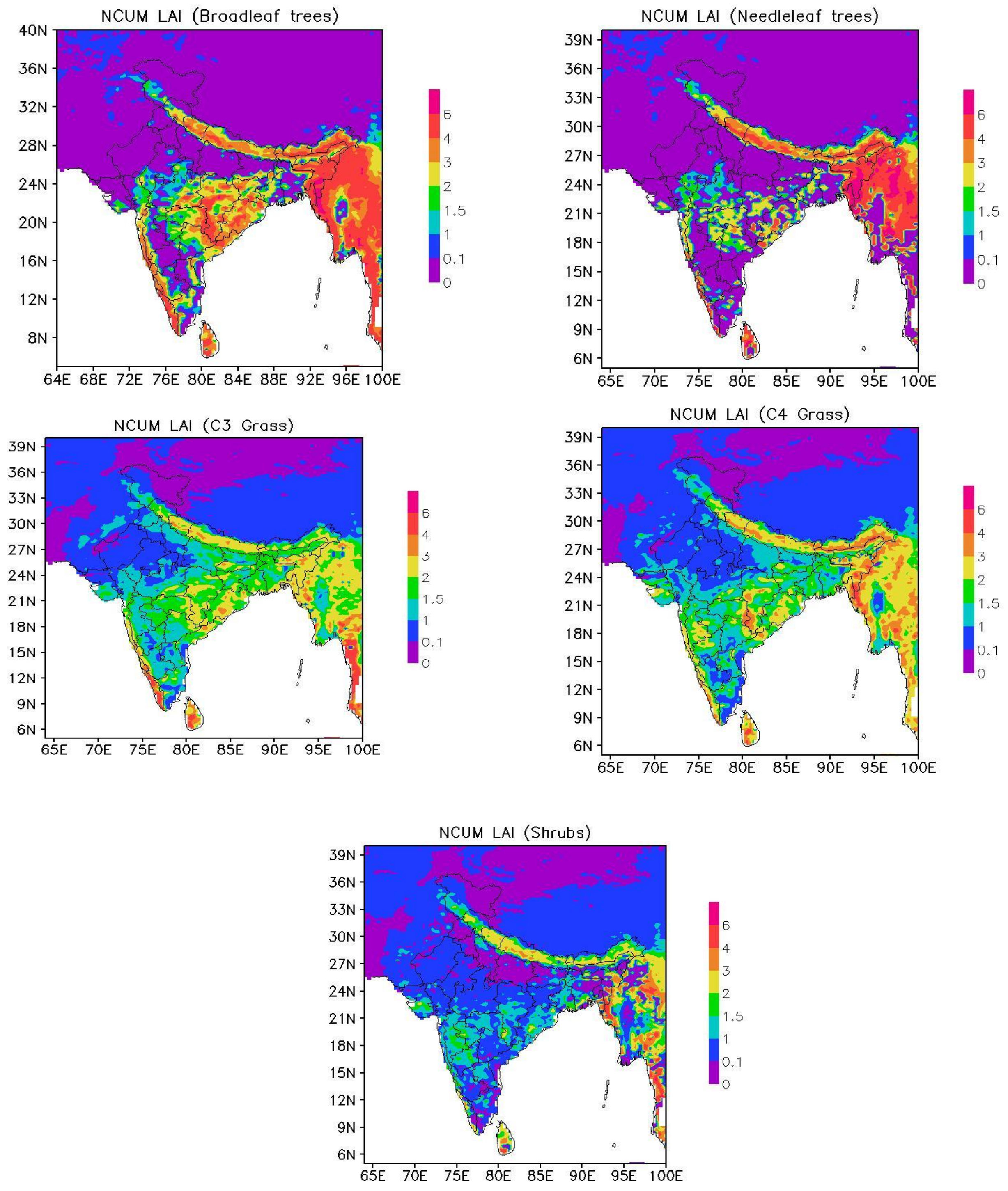

NCUM LAI (Broadleaf trees)
NCUM LAI (Needleleaf trees)
NCUM LAI (C3 Grass)
NCUM LAI (C4 Grass)
NCUM LAI (Shrubs)
40N 36N 32N 28N 24N 20N 16N 12N 8N
64E 68E 72E 76E 80E 84E 88E 92E 96E 100E
39N 36N 33N 30N 27N 24N 21N 18N 15N 12N 9N 6N
65E 70E 75E 80E 85E 90E 95E 100E
6
4
3
2
1.5
1
0.1
0

Figure 9: Leaf Area Index in NCUM for different plant functions