hygIRC: review of the implementation of a hygrothermal simulation model

Defo, M.

Publisher's version / Version de l'éditeur:
https://doi.org/10.4224/23003967
Internal Report (National Research Council of Canada. Construction); no. A1-012761.1, 2018-02-02

NRC·CNRC Construction

hygIRC — Review of the Implementation of a Hygrothermal Simulation Model

M. Defo

Internal Report: A1-012761.1

02 February, 2018

National Research Council Canada
Conseil national de recherches Canada

Canada

This page left blank

hygIRC — Review of the Implementation of a Hygrothermal Simulation Model

Author M. Defo, Research Officer

Approved_____
Program lead: Mr. Philippe Rizcallah
NRC, Construction Research Centre

Report No: A1-012761.1
Report Date: 02 February 2018
Contract No: A1-012761
Agreement date: 20 November 2017
Program: Building Regulations & Market Access

138 pages

Copy No. 1 of 5 copies

This page left blank

Summary

The purpose of this work was to review and document the implementation of heat, air and moisture transport equations in hygIRC. The lack of technical documentation has made it difficult to understand its implementation and to consider modifications to the program and as well, has hindered the development of hygrothermal models using more up-to-date software platforms. As such, the objectives of this exercise were to:

- Review the fundamental equations implemented in hygIRC
- Review how external and internal boundary conditions have been derived and implemented
- Review how material properties have been implemented

In this work, as no particular reference was found that describes the algorithm used for hygIRC, a systematic review of the program was undertaken, in which instructions were reviewed step by step such that one could discern, having reviewed the program in its entirety, how the transport equations, material properties and boundary had been implemented.

The primary findings of this report are summarized in the main body of the text whereas the implementation details are given in Annex I. In Annex II an example of a typical input file generated by the user interface is given. A description of the Delta-formulation for the Approximate Factorization method used for solving heat and moisture equations is provided in Annex III whereas in Annex IV are described the important variables appearing in the program.

In the actual program, some of the critical parameters that form part of the material properties and that can have significant effects on the simulation results are not considered. For examples, the effects of temperature and hysteresis on the sorption curve, the effect of temperature on moisture diffusivity, the effect of saturation percentage on air permeability are not considered in the program. If the intent of the simulation is to perform comparisons (i.e. two locations, two sets of climate data, etc.), this then is less critical. However, for the purposes of design and for which a clear understanding of the effects of climate loads on building envelope are needed to permit assessing the durability of the envelope components and assemblies, efforts should be made to properly characterize the material properties and to implement them in the program.

Table of contents

List of symbols and units

Symbol	Meaning	Units
c	Specific heat capacity	J/kgK
c	Cloudiness	
D	Liquid diffusivity	m^2/s
d	Day of the year	kg/mole
DRF	Driven-rain factor	s/m
DS	Rain drop size	mm
fl	Liquid fraction	
g	Gravity	m/s^2
g	Mass flux of air	kg/m^2s
H	Enthalpy	J/kg
h	Heat transfer coefficient	W/m^2K
I	Irradiation	W/m^2
k	Moisture permeability	kg/msPa
k	Air permeability	kg/msPa
L	Latent heat	J/kg
M	Molecular mass	kg/mol
P	Pressure	Pa
p	Partial pressure	K
q	Moisture flux	kg/m^2s
q	Heat flux	W/m^2
Q	Moisture source Heat source Air source	kg/m^3s J/m^3s kg/m^3s
RDF	Rain deposition factor	
RH	Relative humidity	
t	time	s, days
T	Temperature	°C or K
u	Moisture content	kg/kg
V	Velocity	m/s
WDR	Wind-driven rain	kg/m^2s
z	height	M
ρ	Density	kg/m^3
α	Sun elevation angle or solar altitude	
α	Absorption coefficient	
β	Wall inclination	
β	Mass transfer coefficient	kg/m^2s
ψ	Wall azimuth	
ϕ	Solar azimuth angle	
ω	Hour angle	
	Water vapour permeability	kg/msPa
δ	Earth declination angle	
θ	Angle of incidence of the solar rays on the wall surface	
θ	Angle between the normal to the wall and wind direction	
φ	Latitude	
λ	Thermal conductivity	W/mK
σ	Stephan-Bolztmann constant	
μ	Dynamic viscosity	Pa s
ε	Porosity, emissivity	

Subscripts

a	air
avg	Average
c	Heat, capillary
d	Direct
dif	Diffuse
DN	Direct normal
e	external
eff	Effective
f	Cell face or interface
h	heat
i	Internal, indoor, node number, ice
in	Internal, indoor
I	Index
in	Indoor
l	Liquid
L	Liquid
m	Moisture
n	Neutral plan
o	Dry Outdoor Sum
out	Outdoor
r	reference
r	radiation
r	Reflected, reference
S	Surface
sky	Sky
Surf, S	Surface
t	Terminal
T	Total
T	Temperature
v	Vapour
vs	Saturated vapour
w	Water, wind

hygIRC — Review of the Implementation of a Hygrothermal Simulation Model

by

Maurice Defo, Ph.D.

1.0 Introduction

The hygrothermal simulation model hygIRC has been in use within NRC for more than two decades. The first version seems to have been implemented in the 1990's based on the work of Ojanen et al. (1989) of VTT. VTT had developed a model named TRATMO2[1] to solve temperature, moisture and flow fields for building envelope structures. Conduction, convective flows and radiation were taken into account in the heat and moisture transfer equations, although the initial version of the TRATMO2 model did not include latent heat transfer. The finite difference method was used for the spatial discretization of the transfer equations, and the Crank-Nicolson scheme (Mujumdar 2005) for temporal discretization. An alternating direction method (ADI) was then used to solve matrix equations created for the temperature and moisture fields.

Since these early years, several improvements have been made to hygIRC. The method of discretization of the transfer equations has been changed to the use of a control volume method, and the delta-form of the approximate factorization (AF) method is used to solve moisture and heat equations (Salonvaara and Karagiozis 1994). Although the method is referred to in this paper (Salonvaara and Karagiozis 1994) and an expression of the final formula is given, the actual method or any details are not described, and neither is it mentioned in any subsequent papers written by Salonvaara and Karagiozis pertaining to this simulation model. In fact, most of the articles or reports found in the publications archive give only a general description of hygIRC, it usefulness and usage, or deals with the validation of the program (Karagiozis and Kumaran 1993, Karagiozis and Salonvaara 1995, Mukhopadhyaya and Kumaran 2000, Djebbar *et al*. 2002, Cornick *et al*. 2003, Van Reenen 2004, Maref *et al*. 2003, 2004, Burrows and Gallagher 2007, Saber et al. 2012). Moreover, the source code to the program was itself not well commented which would otherwise have facilitated the review process and provided a more in-depth view of this tool.

The lack of technical documentation has made it difficult to understand the implementation of hygIRC and to consider modifications to the program. As such, the objectives of this exercise are to:

[1] Transient Analysis Code for Thermal and Moisture Physical Behaviour of Constructions in 2-Dimensions

- Review the fundamental equations implemented in hygIRC
- Review how external and internal boundary conditions have been derived and implemented
- Review how material properties have been implemented

Overall, permit mastering the program it appeared necessary to document the different features with the expectation that this would in turn help the subsequent process of improving, if necessary, some of its current features. As no particular reference was found that describes the algorithm used for hygIRC, there was no choice but to undertake a systematic review of the program, indeed reviewing each of the instructions step by step such that one could discern, having reviewed the program in its entirety, how the transport equations, material properties and boundary had been implemented.

The primary findings of this report are summarized in the main body of the text whereas the implementation details are given in Annex I. In Annex II an example of a typical input file generated by the user interface is given. A description of the Delta-formulation for the Approximate Factorization method is provided in Annex III whereas in Annex IV are described the important variables appearing in the program.

2.0 Overview of hygIRC 1-D

The Construction Research Centre's hygIRC software is a simulation tool for modeling Heat, Air and Moisture (HAM) transport in building envelope structures. There are two versions of the hygIRC model: hygIRC 2-D and hygIRC 1-D. Focus in this report is on hygIRC 1-D, which is a simplified version of hygIRC 2-D; the 1-D version has a fully developed graphical user interface. The algorithms used in both versions are similar.

A general overview of hygIRC 1-D is given elsewhere (Cornick *et al.* 2003, Van Reenen 2004, Maref et al. 2004, Burrows and Gallagher 2007). The basic components of hygIRC 1-D are given in Figure 1 in which can be seen that hygIRC 1-D is comprised of:

- A Graphical User Interface (GUI: ***hygIRC.exe***) where all parameters of the problem once set are communicated to the Solver. The GUI was developed in Visual Basic version 5 or version 6.
- The Solver (the Core program: ***hygIRC.dll***) simultaneously solves the mass (air and moisture) and energy balance equations and communicates the simulation results to the GUI for visualization and export. The Core program was developed in FORTRAN (probably Fortran 77, although there are some additions that were implemented using Fortran 90).

A detailed description of the various components and parameters will be given in the subsequent sections.

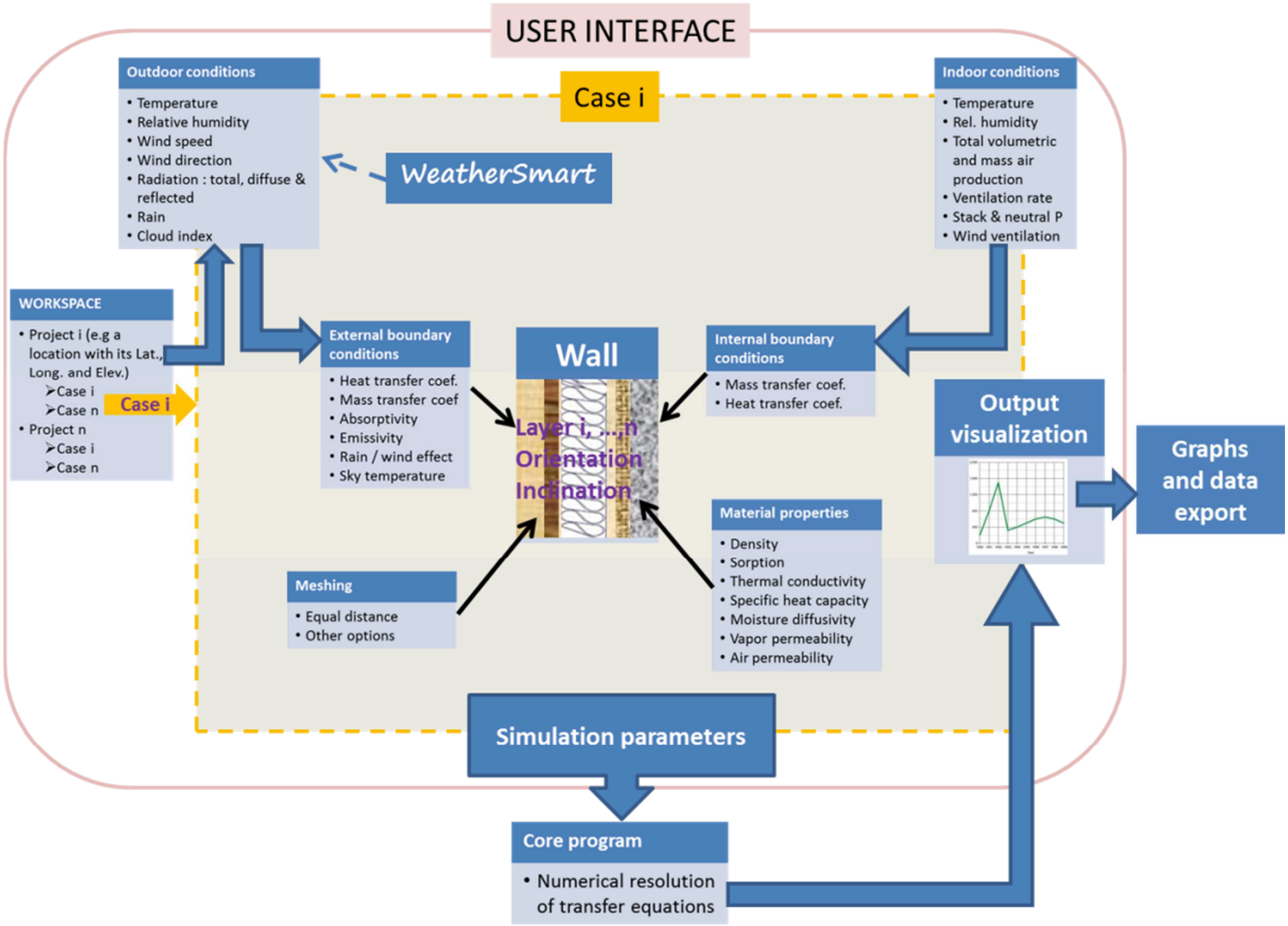

Figure 1. Overview of hygIRC 1D

2.1 The 1-D graphical user interface

The GUI allows users to:

(1) Select or define the location (city) characterized by its latitude, longitude and elevation.

(2) Design the building envelope — Configure the wall assembly in a series of layers and specify the orientation and inclination of the wall. These two parameters are used to derive the wind-driven rain and solar radiation that will impinge on the outside wall surface. The user then selects the composition of each layer and defines its thickness.

(3) Define the material properties of each layer — The materials database covers over 80 common North American construction materials and includes for each material: dry density, sorption and suction data, water vapour permeability, moisture diffusivity, air permeability, heat capacity, and thermal conductivity. Each material is linked to its default properties. However, the user has the option to import materials property data from other sources.

(4) Define the outdoor conditions — There is a database of climate data for 19 Canadian and 6 US locations. For each location, 30 to 40 individual years of hourly weather data are available. The basic hourly elements are dry bulb temperature, relative humidity, wind speed and direction, solar radiation, rain, and cloud cover. Users can select individual years from the climate database or sequences of years, in chronological or non-chronological order, for multi-year simulations. The user always has the option of providing 1-D hygIRC with user-defined weather data, provided the input files conform to the hygIRC format. There is also an optional module, the WeatherSmart module, that can be used to analyse the outdoor weather data from which can be obtained a set of representative environmental boundary conditions for wall simulations (Djebbar et al., 2001). This results in representative exterior and interior moisture conditions for a given reference year and location for the intended building under consideration.

(5) Define the indoor conditions (temperature and relative humidity) — There are several indoor condition models implemented in hygIRC 1-D. These include: models with controlled temperature and relative humidity, and models with uncontrolled relative humidity (class models, Johns' model, constant input, and ASHRAE Standard 160). Other parameters to consider in deriving the indoor conditions are the mechanical ventilation rate, the total volume and mass of air produced, the wind ventilation and the stack effect with neutral pressure level. This information also assists in simulating the envelope leakage-rate coefficient.

(6) Define the external boundary conditions — The user must enter the values for the coefficient of heat and mass transfer at the exterior surface of the wall, as well as the absorptivity and emissivity of the wall due to solar radiation. The user must also choose

whether or not to include the rain and wind effects and the sky temperature. When a specific weather file is chosen, the transfer coefficients are calculated at each time step.

(7) Define the indoor boundary conditions — The user must enter the values for the coefficients of heat and mass transfer occurring at the interior surface of the wall. If an indoor conditions data file is provided, the interior temperature and relative humidity are updated at each time step. Otherwise, interior conditions are calculated using exterior weather conditions.

(8) Simulation parameters — Apart from all the parameters mentioned above, the user must also define the meshing pattern for each layer (or accept the default scheme), and the time-step for numerical resolution of the transfer equations, which by default is one hour.

2.2 The Core program

The core program (***hyhIRC.dll***) is where the transport equations are implemented and solved.

2.3 Data exchange between the interface and the core program

Data are exchanged between the user interface and the core program through various input and output files, which may either saved, and retained for subsequent use, or temporary and deleted at the end of the simulation process.

2.3.1 Input files

The main input file generated from the interface is text file referred to as ***hygirc.in*** and that contains all the simulation parameters. An example of an hygirc.in file is given in Annex II.

Material properties of the respective wall layers selected are generated by the program from the hygIRC database and are subsequently stored under different names:

- *Aper.dat*: air permeability data
- *Cond.dat*: thermal conductivity data
- *Dens.dat*: density data
- *Difl.dat*: water diffusivity data
- *Hcap.dat*: heat capacity data
- *Vper.dat*: water vapour permeability data
- *Interp.dat*: sorption data
- *Psuc.dat*: suction pressure data

Other input files are generated by the interface or provided by the user, depending whether use of a file has been made rather than entering the data in the interface:

- *Exterior.dat*: when applicable, it is the weather data file that contains hourly data.

- *Interior.dat*: when applicable, it is the indoor hourly temperature and relative humidity. The first column indicates the hour, while the column 2 and 3 indicate the corresponding temperature and relative humidity. This file should contain the same number of lines as the weather file.
- *Sources.dat*: when applicable, it contains moisture and heat source data. This file should contain the same number of lines as in the weather file.
- *Initial.con*: when applicable, it contains the initial values of RH or u, and T for each nodal point.
- *Rad.dat*: when applicable, it contains the emissivity of the internal layers for computing surface-to-surface radiation.

2.3.2 Output files

Once the simulation has completed, the user can visualize the results within the GUI or export the output data to a third-party program for further analysis. The Graph utility allows visualization of the:

- Temporal variation of the total moisture content of the entire structure or for a particular layer,
- Temporal variation of the mass flux, and dry heat, air heat, latent or total heat fluxes at the interior or exterior wall surfaces,
- Spatial variation of the temperature, relative humidity or moisture-content for the entire structure,
- Spatial variation of the RHT(80) or RHT(95) in the entire structure,
- The spatial variation for Freeze/Thaw cycles,

The user can also choose to animate the results or compare data for two simulations.

The files containing these data are generated by the core program:

- *Meanr-h.out*: mean values for RH and T
- *Temper.dat*: T file
- *Moiscon.dat*: moisture content file
- *Relhud.dat*: relative humidity file

3.0 Transport equations

3.1 Moisture transfer

The conservation equation for the mass flow is written as:

$$\rho_o \frac{\partial u}{\partial t} + \nabla . q_m = Q_m \tag{1}$$

where:

u: moisture content ($kg_{moisture}$)/($kg_{dry\ material}$)

ρ_o: density of the dry porous material (kg/m^3)

q_m: total moisture flux (kg/m^{2}s)

Q_m: moisture source or sink (kg/m^{3}s), other than wind-driven rain

Wind-driven rain is modelled as a source term in the boundary condition of the exterior wall surface. The total moisture flux, q_m, includes four terms: the diffusion of liquid water under the gradient of the moisture content, u; the diffusion of vapour under the gradient of the partial vapour pressure, p_v; the convective mass transport of vapour due to air movement, and; the liquid transport under the effect of gravity:

$$q_m = -\rho_o D_w \nabla u - \delta_P \nabla p_v + \rho_v V_a + k_w \rho_w g \tag{2}$$

where:

D_w: moisture diffusivity (m^2/s)

δ_p: water vapour permeability (kg/ m s Pa)

p_v: partial water vapour pressure (Pa)

V_a: air velocity (m/s)

ρ_v: vapour density (kg/m^3)

k_w: moisture conductivity or permeability when u is above capillary region (s or kg/m s Pa)

ρ_w: water density (kg/m^3)

g: gravity (m/s^2)

Inserting Eq. 2 into Eq. 1 gives the mathematical model for moisture transport in the building envelope (Djebbar, 2002):

$$\rho_o \frac{\partial u}{\partial t} + \nabla \cdot (-\rho_o D_w \nabla u - \delta_P \nabla p_v + \rho_v V_a + k_w \rho_w g) = Q_m \tag{3}$$

The Soret effect (thermo-migration [Siau 1995]) is also not considered in the moisture transport model. However, it could be partly taken into consideration if the diffusion coefficient and the water vapour permeability are corrected for temperature. This thermo-diffusion term was present in the original model described by Ojanen *et al*. (1989). Perhaps the term was removed given that its effect on the simulation results appeared negligible.

3.2 Heat transfer

The equation of conservation of energy is:

$$\frac{\partial H}{\partial t} + \nabla \cdot q_h = Q_L + Q_h \tag{4}$$

where:

- H: enthalpy (J/kg)
- q_h: total heat flux (J/m^{2}s)
- Q_h: heat source (J/m^{3}s) or (W/m^3)
- Q_L: heat source due to freeze/thawing of water (J/m^{3}s)

The heat flux comprises four terms: the conduction due to the gradient of temperature within the material, the transport of latent heat of evaporation or condensation due to the diffusion and convection of water vapour, and the transport of sensible heat due to the bulk movement of air (advection):

$$q_h = -\lambda_{eff}\nabla T - L_v \rho_o \delta_P \nabla p_v + L_v \rho_v V_a + \rho_a c_a V_a T \tag{5}$$

where:

- λ_{eff}: effective thermal conductivity of the material (W/mK)
- L_v: enthalpy of evaporation or condensation (J/kg)
- ρ_a: dry air density (kg/m^3)
- c_a: dry air specific heat capacity (J/kg C)
- V_a: air velocity (m/s)

From Eq. 4, the heat source, Q_h, represents dissipation of heat other than the change of state of water, and Q_L represents a heat source due to the change of state between liquid water and ice (the process of freezing of water and thawing of ice):

$$Q_L = -L_i \left(\rho_o u \frac{\partial f_l}{\partial t} \right) \tag{6}$$

where:

- L_i: enthalpy of freeze/thaw (J/kg)
- f_l: liquid fraction having a value from 0 to 1

Inserting Eqs. 5 and 6 into Eq. 4 and replacing the enthalpy H by ($\rho_o cT$), we obtain the mathematical expression used to describe heat transport (Karagiozis and Salonvaara, 1995, Djebbar *et al.* 2002, Maref *et al*. 2003):

$$\frac{\partial(\rho_o cT)}{\partial t} - \nabla \cdot (\lambda \nabla T) + \nabla \cdot (\rho_a c_a V_a T) - L_v[\nabla \cdot (\rho_o \delta_P \nabla p_v)] + L_i\left(\rho_o u \frac{\partial f_l}{\partial t}\right) = Q_h \qquad (7)$$

where c is the effective heat capacity of material (J/kg K). In deriving the heat transport model, the transport of sensible heat by water and water vapour is not taken into consideration, as is generally the case (Hens 2012).

3.3 Air transfer

The mass conservation of air in open-porous materials can be written as:

$$\varepsilon \frac{\partial \rho_a}{\partial t} + \nabla \cdot g_a = Q_a \qquad (8)$$

where:

- ε: material porosity
- g_a: mass flux of air (kg/m^{2}s)
- ρ_a: density of air (kg/m^3), function of temperature.
- Q_a: air source or sink of air (kg/m^{3}s)

Ideally, there are no air sources or air sinks. As well, given that the value for the isothermal diffusivity of air is high (i.e.: for wool with a density of 40 kg/m^3, the isothermal air diffusivity is 6.7m^2/s compared to the thermal diffusivity of 1,1 x 10^{-6} m^2/s [Henz 2012]), the inertia of air is generally neglected. Thus, Eq. 8 is reduced to:

$$\nabla \cdot g_a = \nabla \cdot (\rho_a V_a) = 0 \qquad (9)$$

- V_a: air velocity (m^2/s)

For a laminar flow in porous medium, the momentum equation is usually reduced to the general form of the Darcy law:

$$V_a = -\frac{k}{\mu_a} \nabla (P_a - \rho_a g z) \qquad (10)$$

where:

- P_a: air pressure (Pa)
- k: material permeability (m^2)

μ_a: air dynamic viscosity (Pa s)

g: gravity (m^2/s)

z: height (m)

The density in the buoyancy term, $\rho_a g$, accounts for non-isothermal conditions through its variation with temperature. It makes the air transfer equation highly coupled to heat and moisture equations if this variation is taken into account in heat and moisture equations. The so-called Boussinesq approximation (Dean 1998, Bird et al. 2002) is commonly employed to solve the set of equations. It assumes that the air density (which is considered as a for pure fluid) is a linear function of the temperature, $\rho_a = \rho_{ref}\left[1 - \beta\left(T - T_{ref}\right)\right]$, where β is the thermal expansion coefficient, T_{ref} a reference temperature and ρ_{ref} the air density at reference temperature, and that the air density in the continuity, moisture and heat equations can be replaced by the reference density ρ_{ref}. In building physics, air velocities in porous building materials are very small and the temperature involved are generally less than 50°C so that it is also usually assumed that the flow is incompressible (Henz 2012). This allows the continuity equation to be simplified to:

$$\nabla \cdot V_a = 0$$

This leads to the mathematical model used to describe air transport in open-porous materials given by (Djebbar 2002):

$$\nabla \cdot \left[-\frac{k}{\mu_a}\nabla P_a - \rho_a g\right] = 0 \tag{11}$$

If the buoyancy term is neglected in the momentum equation, the velocity and pressure fields can be solved prior to the hygrothermal fields.

4.0 Initial conditions

The initial values of RH or u, and T can be set for each node, each layer, or as a constant value for the entire wall; these values can also be imported from a file.

5.0 Boundary conditions

In this section boundary conditions are described for which conditions are provided in respect to the transfer of moisture, thereafter the transfer of heat, and finally the transfer of air due to pressure differences across the indoor and outdoor boundaries. In each of the respective sections, the indoor boundary is first considered and afterward the outdoor boundary conditions.

5.1 Moisture transfer

5.1.1 Top and bottom boundary

For 1D and 2D with vertical cut cases, the top and bottom surfaces are assumed adiabatic (no mass transfer, $\beta = 0$); for the 2-D case with horizontal cut, the lateral surfaces are assumed adiabatic.

5.1.2 Indoor boundary

At the indoor surface:

$$q_{m,i} = \beta_i\left(p_{v,i} - p_{vsurf,i}\right) + \rho_v V_a \tag{12}$$

where

- $q_{m,i}$: moisture flux normal to the interior surface of the building envelope (kg/m^{2}s)
- $p_{vsurf,i}$: vapour pressure at the indoor surface (Pa)
- $p_{v,i}$: vapour pressure of the indoor air (Pa)
- βi: indoor surface film coefficient for convection (kg/m^{2}sPa or s/m)
- ρ_v: vapour density (kg/m^3)
- V_a: air velocity (m/s)

The vapour density is either the one calculated at the indoor wall surface in case of infiltration ($V_a > 0$) or the one prevailing indoor in case of air exfiltration ($V_a < 0$).

5.1.3 Outdoor boundary

The transfer of moisture across the boundary layer at the external wall surface is:

$$q_{m,e} = \beta_e\left(p_{v,o} - p_{vsurf,o}\right) + \rho_v V_a + WDR \tag{13}$$

where:

- $q_{m,e}$: moisture flux normal to the exterior surface (kg/m^{2}s)
- $p_{vsurf,o}$: vapour pressure at the external surface (Pa)
- $p_{v,o}$: vapour pressure of the outside air (Pa)
- β_e: outdoor surface film coefficient (kg/m^{2}sPa or s/m)
- WDR: wind driven rain (mm/h or kg/m^{2}s)
- ρ_v: vapour density (kg/m^3)
- V_a: air velocity (m/s)

If air is entering the structure (V_a >0), the vapour density used in Eq. 13 is the outdoor vapour density calculated using the equivalent outdoor temperature. In the opposite case when air is leaving the structure (V_a < 0), it is the vapour density prevailing at the external wall surface.

5.1.3.1 Wind driven rain

The rain deposition rate to the exterior surface of the wall is calculated as (Function WCORR):

$$WDR = RAIN_m \times \cos\theta \times V_z \times DRF \times RDF \tag{14}$$

in which:

$$\theta = \psi - \beta_w$$

$$V_z = \left(\frac{V}{3.6}\right)\left(\frac{Z}{10}\right)^{0.22}$$

$$DRF = \frac{1}{V_t}$$

$$V_t = -0.166033 + 4.91844 \times DS - 0.888016 \times DS^2 + 0.054888 \times DS^3$$

$$DS = \left(1.3 \times RAIN_m{}^{0.232}\right)\left(1 - \frac{1}{X_N}\right)^{\frac{1}{X_N}}$$

where:

WDR: wind driven rain (mm/h or kg/m^{2}s)

$RAIN_m$: average rainfall rate on a horizontal surface (mm/h)

θ: angle between the normal to the wall and the wind direction

ψ: wall orientation(north = 0, south = 180, east = 90 or west = 270)

β_w: wind direction (°)

V_z: wind speed (m/s) at height considered *Z* (m)

V: wind speed (m/s) from weather data

DRF: driven rain factor

V_t: terminal velocity of raindrop in still air (m/s)

DS: rain drop size (mm)

X_N: 2.25

RDF: Rain deposition factor, set to 0.4 in the program

Notes:

- A value of z = 1.8 m is actually used in the program, which corresponds to the critical height of wind driven rain for low-rise residential buildings. For other building types

(mid-rise, high-rise) another value of z should be used. The option to change it is not available in the GUI.

- The value of 0.4 assigned to RDF corresponds to the value at mid-height of low-rise buildings. For corners of small buildings and for taller buildings, the RDF values increase; additional information on values for the RDF can be found in Straube and Burnett (2005).
- The way the absorption of the rain at the outdoor wall surface is handled can be found in Section A1.17.8.

5.1.4 Convective mass transfer coefficients

The values of the convective mass transfer coefficient, β, can be obtained directly from the input file provided the use of default constant values was selected; these values are set to:

$$\begin{cases} \beta_e = 2.1 \times 10^{-7} \\ \beta_i = 5.9 \times 10^{-8} \end{cases}$$

These values can be modified in the GUI. Whereas if a weather file was selected to be used in the simulation, the exterior surface emission, β_e, is calculated at each time step using the air velocity and the analogy between heat and mass transfer. The following relation is used (subroutine WEATH):

$$\beta_e = h_c \left(\frac{2.1 \times 10^{-7}}{30 + 10^{-10}} \right) \tag{15}$$

where h_c is the external heat transfer coefficient (see section 6.2); the default external heat transfer coefficient has a value of 30.

5.2 Heat transfer

5.2.1 Top and bottom

For the 1D case, they are assumed adiabatic (zero heat flux). The same applies for the 2-D case with vertical cut. For 2-D case with horizontal cut, it is the lateral surfaces that are assumed adiabatic.

5.2.2 Outdoor boundary

The heat exchange at the exterior wall surface involves four terms: convective and radiative heat transfer, absorption of short and long wave radiation and emission of long wave radiation, latent heat due to vapour transfer by diffusion and mass convection and sensible heat of air flowing into or out of the wall:

$$q_{h,e} = h_o\left(T_{e,o} - T_{s,e}\right) - \varepsilon\sigma\left({T_{S,e}}^4 - {T_{sky}}^4\right) + L_v\beta_e\left(p_{vsurf,e} - p_{v,o}\right) + L_v\rho_v V_a + q_a \tag{16}$$

in which:

$$h_o = h_c + h_r \tag{17}$$

where:

- $q_{h,e}$: total heat flow across the exterior surface (W/m^2)
- h_o: sum of convective and radiative heat transfer at the exterior surface (W/m^{2}K)
- h_c: convection heat transfer coefficient (W/m^{2}K)
- h_r: radiation heat transfer coefficient (W/m^{2}K)
- $T_{s,e}$: exterior wall surface temperature (°C or K)
- $T_{e,o}$: equivalent exterior temperature or sol-air temperature (°C)
- T_{sky}: sky temperature (K)
- q_a: sensible heat of dry air flowing into or out of the wall due to bulk air movement (W/m^2)

The sensible heat due to wind driven rain is neglected.

The radiation heat coefficient, h_r, appearing in Eq. 17, considers only the long-wave radiation exchange between the surface and the terrestrial environment (Eq. 20), and the short-wave absorption and is taken into account in the computation of sol-air temperature (Eq. 23). The second term on the right hand side of Eq. 16 describes the long-wave radiation exchange between the wall surface and the sky; this term is added later in the program during computation of the heat flux at the external surface (subroutine QT).

The advection term, q_a, is computed as:

$$q_a = \rho_a c_a \left[V_f \, T_{a,f} - \max(V_f, 0) \times \Delta T_o \right] \tag{18}$$

in which: $\begin{cases} T_{a,f} = T_{S,e} \; if \; V_f < 0 \\ T_{a,f} = T_{e,o} \; if \; V_f > 0 \end{cases}$ and $\begin{cases} \Delta T_o = T_e - T_a \; if \; V_f > 0 \\ \Delta T_o = 0 \; otherwise \end{cases}$

where:

- V_f: air velocity at the air-solid interface (m/s)
- T_f: air temperature at the interface (°C)
- $T_{a,o}$: outside air temperature (°C)

5.2.2.1 Convection heat transfer coefficient

The default values for the convective heat transfer coefficients are set in the GUI to 30 for an outdoor surface and 8 for an indoor surface. When a weather file is used, the convective heat transfer coefficient for the exterior surface is adjusted as a function of the wind velocity V (km/h) acting on that surface:

$$\begin{cases} h_c = 5.82 + \dfrac{3.96V}{3.6} & if\ V\ \leq 18\ km/h \\ h_c = 7.68\left(\dfrac{V}{3.6}\right)^{0.75} & if\ V\ > 18\ km/h \end{cases} \tag{19}$$

The minimum value of h_c in the case V > 18 km/h (5 m/s) is set to 5 W/m^{2}K.

5.2.2.2 Radiation heat transfer coefficient

The radiation heat transfer coefficient is calculated as:

$$h_r = \varepsilon\sigma(\bar{T} + 273.15)^3 \tag{20}$$

where

ε: emissivity of the wall

σ: Stefan-Boltzmann constant

$\bar{T}$ (°C) is the average temperature between the surface temperature of the wall and the ambient air temperature:

$$\bar{T} = \frac{1}{2} \times \left(\frac{h_c T_a + \dfrac{\lambda}{\Delta x_{1/2}} T + L_v(Q_{V1} - Q_{V2})}{h_c + \dfrac{\lambda}{\Delta x_{1/2}}} + T_a \right) \tag{21}$$

where:

λ: thermal conductivity (W/mK)

T: temperature of the internal node near the surface (°C)

T_a: outdoor air temperature (°C)

L_v: latent heat of evaporation of water (J/kg)

Q_{V1}: vapour flux arriving to the surface (kg/m^{2}s)

Q_{V2}: vapour flux leaving the surface and entering the wall (kg/m^{2}s)

5.2.2.3 Sky temperature

The model used to compute sky temperature is given as:

$$T_{sky} = \left(F_1 {T_a}^4 + F_2 A^4\right)^{0.25} \tag{22}$$

in which:

$$F1 = 1 - F2$$

$$F_2 = \left[cos\left(\frac{0.5\beta\pi}{180}\right)\right]^2$$

$$A = (1 - 0.125c_I) \times \left(0.0552{T_a}^{1.5}\right) + 0.125c_I T_a$$

where:

T_a: outdoor air temperature (K)

c_I: cloudiness index

β: wall inclination (°)

5.2.2.4 Sol-air temperature

The equivalent exterior temperature is expressed by:

$$T_{e,o} = T_{a,o} + \frac{\alpha I_T}{h_o} \tag{23}$$

where:

α: external wall surface absorption coefficient

I_T: total irradiance (W/m^2)

ε: emissivity of the wall surface

Note: In literature, an additional term $(-\epsilon\Delta R/h_o)$ is added on the right hand side of Eq. 23 that accounts for the difference, if any, between the long wave sky radiation incident on the surface and surroundings and the radiation emitted by a blackbody at an outdoor temperature, $T_{a,o}$. For vertical surfaces, it is commonly accepted that ΔR = 0 (Hutcheon and Handegord 1983).

5.2.2.5 Total irradiation

The total irradiation, I_T (W/m^2), received on the wall surface is:

$$I_T = I_d + I_{dif} + I_r \tag{24}$$

where:

I_d: direct sun beam

I_{dif}: diffuse component

I_r: reflected component

i) Reflected component

The reflected component, I_r, is computed as:

$$I_r = R_I \times \left[\frac{1 - \cos\left(\frac{\beta\pi}{180}\right)}{2}\right] \tag{25}$$

where R_I is the reflected irradiance from weather data, and β is the inclination angle of the wall.

ii) Diffuse component

The diffuse component, I_{dif}, is computed as:

$$I_{dif} = D_{HI} \times \left[\frac{1 + \cos\left(\frac{\beta\pi}{180}\right)}{2}\right] \tag{26}$$

where D_{HI} is the diffuse irradiance falling on a horizontal surface.

iii) Direct solar radiation

The direct solar radiation, I_d, received by the wall is computed as:

$$I_d = \frac{I_{DN}}{\sin\alpha} \times \cos\theta \tag{27}$$

where:

I_{DN} : direct normal irradiance from weather data (W/m^2)

α: sun elevation angle or solar altitude (°)

θ: angle of incidence of solar rays on the wall (angle between the incoming solar rays and a line normal to the wall surface) (°)

<u>Elevation angle, α</u> (°)

$$\sin\alpha = \sin\delta \sin\left(\frac{\pi}{180}\varphi\right) + \cos\delta \cos\left(\frac{\pi}{180}\varphi\right) \cos|\omega| \tag{28}$$

where:

δ: earth declination angle (°)

ω: hour angle (°)

φ: latitude (°)

The earth declination angle is:

$$\delta = 23.45\frac{\pi}{180}sin\left(\frac{\pi(280.1 + 0.9863d)}{180}\right) \tag{29}$$

where d is the day of the year (1 to 365) calculated at each time step as the ratio of the cumulative hours of simulation in a given year over 24.

The hour angle is:

$$\omega = 15\left[(12 - t_d) - E_t - \frac{(t_{zo} - l)}{15}\right] \times \frac{\pi}{180} \tag{30}$$

where:

t_d: time of the day (1 to 24)

E_t: equation of time

t_{zo}: time zone

l: longitude (°)

The equation of time, E_t, is expressed as:

$$E_t = 0.1645\ sin(2B) - 0.1255\cos B - 0.0255\sin B \tag{31}$$

with $B = \frac{\pi}{180} \times (0.989d - 80.11)$

<u>Angle of incidence of the solar rays on the wall surface, θ</u>

The parameter θ is computed as follows:

$$\cos\theta = \cos\alpha\ sin\left(\frac{\pi}{180}\beta\right)\cos\gamma + \sin\alpha\ cos\left(\frac{\pi}{180}\beta\right) \tag{32}$$

in which:

$$\gamma = \phi - \frac{\pi}{180}\psi \tag{33}$$

$$\sin\phi = \frac{\cos\delta\ \sin|\omega|}{\cos\alpha} \tag{34}$$

where:

β: wall inclination angle (°)

ψ: wall azimuth or orientation angle (°)

ϕ: solar azimuth angle (°)

ω: hour angle (°)

δ: earth declination angle (°)

There are some constraints on the solar azimuth angle, ϕ, calculated by the Eq. 34, based on the parameters P1 and P2 expressed as:

$$P_1 = 24\frac{12-\omega}{2\pi} \tag{35}$$

$$P_2 = \frac{\tan\delta}{\tan\left(\frac{\pi}{3}\right)} \tag{36}$$

- $If\ (cos\ \omega < P2\ and\ P1 > 12)\ then\ \phi = 2\pi - \phi$
- $If\ (cos\ \omega = P2\ and\ P1 < 12)\ then\ \phi = \pi/2$
- $If\ (cos\ \omega = P2\ and\ P1 > 12)\ then\ \phi = 1.5\pi$
- $If\ (cos\ \omega > P2\ and\ P1 < 12)\ then\ \phi = \pi - \phi$
- $If\ (cos\ \omega > P2\ and\ P1 > 12)\ then\ \phi = \pi + \phi$

5.2.2.6 External wall surface temperature

In the course of the program, the external wall surface temperature is calculated using temperature obtained at a previous time step; this is given as:

$$T_{S,e} = \frac{a_i T_e + a_{i+1}T_{i+1} + L_v\left(Q_{v,i} - Q_{v,i+1}\right) - \rho_a c_a V_f \Delta T_o}{a_i + a_{i+1}} \tag{37}$$

with:

$$a_i = h_e + \rho_a c_a \max(V_f, 0)$$

$$a_{i+1} = \frac{\lambda_{i+1}(u_{i+1}, T_{i+1})}{\Delta x_{1/2}} - \rho_a c_a \min(V_f, 0)$$

$$\begin{cases} \Delta T_o = T_e - T_a\ if\ V_f > 0 \\ \Delta T_o = 0\ otherwise \end{cases}$$

where:

T_{i+1}:	temperature at node i+1 (°C)
$Q_{V,i}$:	vapour flux arriving at the surface (kg/m^{2}s)
$Q_{V,i+1}$:	vapour flux leaving the surface and entering the wall (kg/m^{2}s)
T_a:	outdoor air temperature (°C)
T_e:	sol-air temperature (°C)

$\lambda_{i+1}(u_{i+1}, T_{i+1})$: thermal conductivity (W/m K) at node i+1, calculated by the function COND

5.2.3 Indoor boundary

5.2.3.1 Indoor T and RH

They are either read in the input file for steady conditions, read in the interior weather file (which is generally the case for 1D where the interior conditions are generated automatically using any of the models available in the GUI), or calculated using the exterior weather conditions.

5.2.3.1.1 Case interior weather file exists

The interior weather file, if it exists, contains minimally for each hour, the indoor air temperature (T_{in}) and relative humidity (RH_{in}). It can also contain the specified pressure difference across the wall in which case it is also read and assigned to the variable *specified_pressure_difference*.

5.2.3.1.2 Case there is no interior weather file

If there is no indoor weather file, the indoor *T* and *RH* are calculated using indoor average daily conditions read from the file *tindoor.bcf* and the exterior weather conditions:

$$T_{in} = \max(T_{in\_avg}, T_{a,o} + 3) \tag{38}$$

$$p_{v,in} = \frac{u_{in} P_o}{0.622 + u_{in}} \tag{39}$$

$$RH_{in} = \frac{p_{v,in}}{p_{vs}(T_{in})} \tag{40}$$

in which:

$$u_{in} = u_{out} + u_{in,avg} \tag{41}$$

$$u_{out} = \frac{0.622 p_{v,out}}{P_o - p_{v,out}} \tag{42}$$

where:

T_{in}: indoor air temperature (°C)

T_{in_avg}: average indoor air temperature read from the file *tindoor.bcf* (°C)

$u_{in,avg}$: average moisture content of the air for the hour of the day

$T_{a,o}$: outdoor air temperature (°C)

u_{out}: moisture content of the outdoor air

Po: 101325 Pa

RH_{in}: indoor relative humidity, should be between RHLIM1 and RHLIM2, the maximum and minimum interior RH read from the file *tindoor.bcf*

5.2.3.2 Indoor heat transfer at the surface

At the indoor wall surface, heat flow includes convection, radiation, advection and latent heat:

$$q_{h,in} = h_i\left(T_{e,in} - T_{s,in}\right) + \varepsilon\sigma\left({T_{S,in}}^4 - {T_{sky}}^4\right) + L_v\beta_i\left(p_{vsurf,in} - p_{v,in}\right) + L_v\rho_v V_a + q_{a,in} \tag{43}$$

in which:

$$h_i = h_c + h_r \tag{44}$$

where:

$q_{h,in}$: total heat flow across the exterior surface (W/m^2)

h_i: sum of convective and radiative heat transfer at the interior surface (W/m^{2}K)

hc: indoor convection heat transfer coefficient (W/m^{2}K)

h_r: indoor radiation heat transfer coefficient (W/m^{2}K)

$T_{s,in}$: indoor wall surface temperature (°C or K)

$T_{e,in}$: equivalent indoor temperature or sol-air temperature (°C)

$T_{a,o}$: outside air temperature (°C)

T_{sky}: sky temperature (K)

q_a: sensitive heat of dry air flowing into or out of the indoor wall surface due to bulk air movement (W/m^2)

The sky temperature is the same as the one computed in Eq. 22. The radiation heat coefficient, h_r, appearing in Eq. 44, considers only the long-wave radiation exchange between the wall surface and the indoor environment, and the short-wave (if there is any) absorption taken into account in the computation of the indoor equivalent temperature (Eq. 47). The second term on the right hand side of Eq. 43, which describes the long-wave radiation exchange between the wall surface and the sky, is added later in the program during computation of the heat flux at the indoor surface.

The advection term, q_a, is computed as:

$$q_a = \rho_a c_a\left[V_f\, T_X - \max(V_f, 0) \times \Delta T_{in}\right] \tag{45}$$

in which: $\begin{cases} T_X = T_{S,in} \; if \; V_f \; < 0 \\ T_X = T_{e,in} \; if \; V_f \; > 0 \end{cases}$ and $\begin{cases} \Delta T_{in} = -\left(T_{e,in} - T_{a,in}\right) \; if \; V_f < 0 \\ \Delta T_{in} = 0 \; otherwise \end{cases}$

where:

V_f= air velocity at the solid/air interface (m/s)

T_X = air temperature at the interface (°C)

$T_{a,in}$: indoor air temperature (°C)

5.2.3.3 Indoor convective heat transfer coefficient

The default value for the convective heat transfer coefficient is set to 8 for indoor surface. This value remains unchanged in the course of the program.

5.2.3.4 Indoor radiation heat transfer coefficient

The radiation component is:

$$h_r = \varepsilon_{in}\sigma(\bar{T}_{in} + 273.15)^3$$

where:

ε_{in}: emissivity of the indoor wall surface

$\bar{T}_{in}$ is the average between surface temperature and the indoor air temperature:

$$\bar{T}_{in} = \frac{1}{2} \times \left(\frac{h_c T_{a,in} + \frac{\lambda_i}{\Delta x_{1/2}} T_i + L_v(Q_{V1} - Q_{V2})}{h_c + \frac{\lambda_i}{\Delta x_{1/2}}} + T_{a,in} \right) \tag{46}$$

where:

λ_i: thermal conductivity (W/m K) of node near the surface

Ti: temperature of the internal node near the surface (°C)

$T_{a,in}$: indoor air temperature (°C)

L_v: latent heat of evaporation of water (J/kg)

Q_{V1}: vapour flux arriving to the surface (kg/m^{2}s)

Q_{V2}: vapour flux leaving the surface (kg/m^{2}s)

5.2.3.5 Indoor equivalent temperature

$$T_{e,in} = T_{a,in} + \frac{\alpha_{in} I_{T,in}}{h_i} \tag{47}$$

$I_{T,in}$ accounts for the short-wave that heat the inside wall and long-wave exchange with surroundings.

5.2.3.6 Indoor surface temperature

The indoor surface temperature is:

$$T_{S,in} = \frac{a_i T_i + a_{i+1} T_{e,in} + L_v\left(Q_{v,i} - Q_{v,i+1}\right) - \rho_a c_a V_f \Delta T_{in}}{a_i + a_{i+1}} \tag{48}$$

with:

$$a_i = \frac{\lambda_i(u_i, T_i)}{\Delta x_{1/2}} + \rho_a c_a \max(V_f, 0)$$

$$a_{i+1} = h_i - \rho_a c_a \min(V_f, 0)$$

$$\begin{cases} \Delta T_{in} = 0 \; if \; V_f > 0 \\ \Delta T_{in} = -\left(T_{e,in} - T_{a,in}\right) \; otherwise \end{cases}$$

where:

T_i: temperature at node i near the surface (°C)

$Q_{V,i}$: vapour flux from node i to the surface (kg/m^{2}s), obtained from array QMOVX

$Q_{V,i+1}$: vapour flux from or to the environment (kg/m^{2}s), obtained from array XV

$T_{e,in}$: effective or equivalent indoor temperature (°C)

h_i: equivalent heat transfer coefficient ($h_c + h_r$) (W/m^{2}K)

$\lambda_i(u_i, T_i)$: thermal conductivity (W/m K) using u and T at node i

5.3 Pressure equation

At the external surface:

$$P = Pe \tag{49}$$

At the internal surface:

$$P = Pi \tag{50}$$

In the case of 1D problem, a zero flow is imposed at the bottom and top boundaries of the wall.

If there is a specified wind pressure difference, ΔP, between the exterior and interior, the pressure at the exterior wall is:

$$Pe = \frac{1}{2} \times \Delta P$$

and the pressure at the interior wall is:

$$Pi = -\frac{1}{2} \times \Delta P$$

If there is no specified pressure difference, the exterior and interior pressures are calculated following many steps. First, the wind pressure, P_w (Pa) is calculated using weather data:

$$P_w = 0.06 \times Coef \times (10^{-10} + V_z)^2 \tag{51}$$

in which:

$$Coef = \frac{A + C\,\Phi_w + E\,\Phi_w{}^2}{1 + B\,\Phi_w + D\,\Phi_w{}^2}$$

$$V_z = \left(\frac{V_r}{3.6}\right) \times \left(\frac{z}{z_r}\right)^{0.22}$$

where:

V_z: wind speed (m/s) at height z (**set to 1.8 m**) in the program

V_r: wind speed (m/s) at reference height, set to 10 m in the program

Φ_w: angle of impediment of wind on the wall (°) $= |\Phi - \beta_w|$ where Φ is the wind direction, and if $\Phi_w > 180$, $\Phi_w = 360 - \Phi_w$

β_w: wall orientation (°)

A, B, C, D, and E are constants equal to 0.5914, -0.01463, -0.01093, 0.000105, and 2.367 x 10^{-5}, respectively.

Then, the pressures on the external and internal faces of the wall are calculated as:

For outdoor:

$$P_e = P_w - \left[(H - H_n)\frac{gP_0M_a}{R(T_{a,o} + 273)}\right] \tag{52}$$

For indoor:

$$P_i = P_{vent} - \left[(H - H_n)\frac{gP_0M_a}{R(T_{a,in} + 273)}\right] \tag{53}$$

where:

H: height at which the pressure is calculated (m)

H_n: height at neutral plan (m), set to 3 m by default.

g: gravity (ms^{-2})

$T_{a,o}$: outdoor air temperature (°C)

$T_{a,in}$: indoor air temperature (°C)

P_w: wind pressure (Pa)

P_{vent}: ventilation pressure (Pa)

P_i: indoor pressure (Pa)

P_0: barometric pressure (101300 Pa)

M_a: molecular mass of air (29 kg $mole^{-1}$)

R: universal gas constant (8314.33 kg $K^{-1}mole^{-1}$

The second term on the right hand side of the equation is the stack pressure.

6.0 Material properties

6.1 Density

The material density, ρ (kg m^{-3}), is provided as data file containing ρ(u). However, ρ is the same for all u values and is the oven dry density (u = 0).

6.2 Heat capacity

The specific heat capacity c (J $kg^{-1}K^{-1}$) is provided as data table containing c(u). However, for the whole range of moisture content, the value of c is constant and is probably c_o, the specific heat capacity at oven-dry state. The value of c is updated in the program as:

$$c = c_o + 4200u \tag{54}$$

The volumetric heat capacity is $\rho_o c = \rho(c_o + 4200u)$. The calculation of the effective heat capacity of the material does not include that of the air and vapour as it can be considered that their volumetric heat capacity is negligible.

6.3 Thermal conductivity

The heat conductivity λ (W m^{-1} K^{-1}) is provided as a data file containing λ(u).

6.4 Sorption Isotherm

The sorption isotherm is provided as data file containing u(RH), where RH is the relative humidity. A single curve is given for each material, which could be adsorption or desorption

curve or average between both. According to Van Reenen, data were acquired in absorption and desorption and then averaged to minimize the hysteresis effect. The temperature effect is not taken into account.

6.5 Water vapour Permeability

The water vapour permeability, δ_P (kg $m^{-1}s^{-1}Pa^{-1}$), is provided as a data file containing δ_P(RH). In the program, the temperature effect is taken into account as follow:

$$\delta_p = \delta_{po} + a\left(\delta_{pu} - \delta_{po}\right) \tag{55}$$

where:

δ_{po}: vapour permeability at dry state

δ_{pu}: vapour permeability at moisture content u as provided in the table

a: coefficient (a = 1 if T ≥ 0°C; a = 0 if T < -5°C; a varies from 0 to 1 when T is between -5 and 0°C)

6.6 Liquid diffusivity

The moisture diffusivity, D_w (m^2/s), which is determined considering u as the potential for moisture flow is given as $D_w(u)$ in a data file. D_w includes liquid and vapour diffusivities so that the true liquid diffusivity needs to be calculated. In the program, the water vapour diffusivity assuming u as the potential is obtained by equating the fluxes of vapour due to moisture gradient and the one due to partial vapour pressure gradient:

$$\rho D_V \frac{\partial u}{\partial x} = \delta_p \frac{\partial p_v}{\partial x} \tag{56}$$

which gives:

$$D_V = \frac{\delta_p}{\rho} \times \frac{\partial P_V}{\partial u} \tag{57}$$

From there, the liquid water diffusivity with u as the potential is given as:

$$D_L = D_w - D_V \tag{58}$$

The minimum value for D_L is set to 10^{-16} and a correction factor f is used that reflects the ratio between the dynamic viscosity of water at a reference temperature to that of the actual temperature, from which one obtains: $D_L = D_L \times f \times (0.01\ if\ T < 0\ or\ 1\ if\ T > 0)$.

The liquid water permeability, k_w, given the suction pressure as the potential for moisture flow, is calculated as:

$$k_w = -\rho_o D_L \frac{\partial u}{\partial P_c} \tag{59}$$

6.7 Air permeability

The permeability provided in the database is the intrinsic permeability of the material (m^2 or m^3_{air}/m_{mat}). It is provided as a data file containing $k(\underline{u})$. However, k is constant for all u. The air permeability should thus be:

$$k_a = \frac{k}{\mu_a} \tag{60}$$

where k_a is the air permeability ($m^3_{air}\ m^{-1}s^{-1}Pa^{-1}$) and μ_a is the dynamic viscosity of air (Pa.s). The value set for μ_a is 1.7 x 10^{-5}Pa.s, which is independent of temperature.

Note:

- There is no correction in the program for k_a as function of the degree of saturation.

6.8 Suction pressure

The suction pressure, s, is provided as a single curve for s as function of u. According to Van Reenen, the data provided was gathered in desorption using pressure plate method. Temperature effect on suction pressure is not considered.

7.0 Implementation of transport equations

The numerical solution for the transport equations comprises the: (i) Discretization of the physical domain into finite cells or elements; (ii) Numerical discretization of the transport equations into a system of algebraic equations defined over each cell, and the; (iii) Solution of the set of equations with appropriate boundary conditions.

7.1 Domain discretization

In hygIRC, the 1D problem is implemented as a pseudo 2D with height and width of the wall equal to 1 m. This implies that the procedure described herein is also applicable to a 2D problem. The finite volume method (FVM) is used for spatial discretization of the momentum, moisture and heat equations. Figure 2 shows a typical control volume: the nodal points around P are labelled W(i-1,j), E(i+1,j), S(i,j-1) and N(i,j+1); Δx_c and Δy_c are the cell sizes, and; Δx and Δy are the nodal distances. The positive fluxes are in the direction from west to east or from outdoor to indoor, and from south to north or from bottom to top.

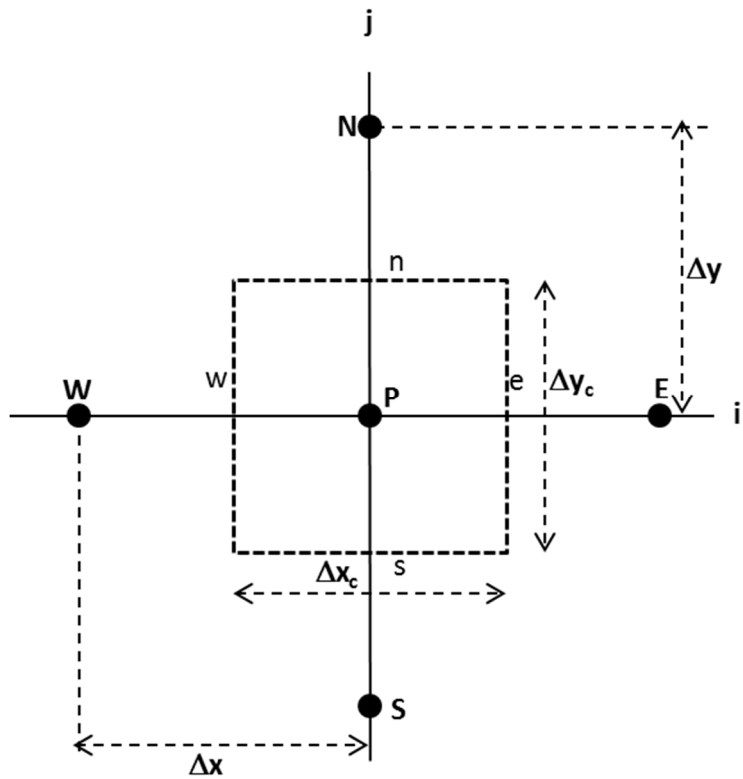

Figure 2. A typical control volume: the control volume surrounds the central node P or node (i,j) and is bounded by the west (w), east (e), south (s), and north (n) faces.

7.2 Pressure equation

The air flow in cracks, joints and other apertures are not considered in the case of a 1D problem. In instances where air flow is considered (for materials permeable to air), the air flow equation is first solved to obtain the values of pressure and then values for the air velocity at nodal points; thereafter, the moisture and heat equations are solved utilizing the velocities obtained in the solution for pressure values and from which the convective mass or heat fluxes are computed.

7.2.1 Discretization of pressure equation

In the case of a 2D system, and assuming constant air density, the pressure equation (Eq. 11) can be rewritten as:

$$\frac{\partial}{\partial x}\left(k_x \frac{\partial P}{\partial x}\right) + \frac{\partial}{\partial y}\left(k_y \frac{\partial P}{\partial y}\right) = 0 \tag{61}$$

where P is the overall air flow potential, encompassing the actual wind pressures, pressure arising from the stack effect and fan pressures, and k the air permeability. It is assumed initially that that the entire domain is at a relative pressure of zero. The boundary conditions for the 1D case are:

$$\begin{cases} x = 0\ (external\ surface), P = P_e \\ x = L\ (internal\ surface), P = P_i \end{cases}$$

Equation 61 is solved using FVM where:

- The physical domain is subdivided into a finite number of small control volumes,
- The governing equation is integrated on each control volume to obtain a discretized form of the equation.

Considering a typical control volume for an interior node as shown in Figure 2, one can write:

$$\int_s^n \int_w^e \left[\frac{\partial}{\partial x}\left(k_x \frac{\partial P}{\partial x}\right)\right] dxdy + \int_w^e \int_s^n \left[\frac{\partial}{\partial y}\left(k_y \frac{\partial P}{\partial y}\right)\right] dydx = 0 \tag{62}$$

Applying the divergence theorem to Eq. 62 in the control volume gives:

$$\left[\left(k_x \frac{\partial P}{\partial x}\right)_e - \left(k_x \frac{\partial P}{\partial x}\right)_w\right]\Delta y + \left[\left(k_y \frac{\partial P}{\partial y}\right)_n - \left(k_y \frac{\partial P}{\partial y}\right)_s\right]\Delta x = 0 \tag{63}$$

The first derivative terms in Eq. 63 can be approximated by considering a piece-wise linear profile for the first derivative. Thus, the flux quantities across the control volume surfaces are:

$$k_x \left.\frac{\partial P}{\partial x}\right|_e \approx \frac{k_{x,e}(P_E - P_P)}{\Delta x} \tag{64}$$

$$k_x \left.\frac{\partial P}{\partial x}\right|_w \approx \frac{k_{x,w}(P_P - P_W)}{\Delta x} \tag{65}$$

$$k_y \left.\frac{\partial P}{\partial y}\right|_s \approx \frac{k_{y,s}(P_P - P_S)}{\Delta y} \tag{66}$$

$$k_y \left.\frac{\partial P}{\partial y}\right|_n \approx \frac{k_{y,n}(P_N - P_P)}{\Delta y} \tag{67}$$

Substituting these approximations into Eq. 63 and rearranging, one obtains the finite volume approximation of Eq. 61 for an interior node:

$$a_P P_P = a_W P_W + a_E P_E + a_S P_S + a_N P_N \tag{68}$$

where the coefficients are:

$a_W = \frac{k_{x,w}}{\Delta x}\Delta y$, $a_E = \frac{k_{x,e}}{\Delta x}\Delta y$, $a_N = \frac{k_{y,n}}{\Delta y}\Delta x$, $a_S = \frac{k_{y,s}}{\Delta y}\Delta x$, and $a_P = a_W + a_E + a_S + a_N$

The node P occupies a position with indices (i, j), the node W, a position with indices (i-1, j), the

node E, a position with indices (i+1, j), the node S, a position with indices (i, j-1), the node N, a position with indices (i, j+1). Eq. 68 can thus be rewritten as:

$$a_{i,j}P_{i,j} = a_{i-1,j}P_{i-1,j} + a_{i+1,j}P_{i+1,j} + a_{i,j-1}P_{i,j-1} + a_{i,j+1}P_{i,j+1} \tag{69}$$

Evaluating Eq. 69 at successive internal nodes leads to a system of n1 x n2 equations with n1 x n2 unknown pressures where n1 and n2 are the number of internal nodes in x and y direction, respectively. For a 1D case, n2 is equal to 1.

After assigning the boundary nodes with the prescribed indoor or outdoor pressures (Dirichlet boundary condition) for cells with one or two faces at the boundary, ordering and moving all the knowns values to the right, one obtains the matrix system that is to be solved:

$$[A]\{x\} = \{c\} \tag{70}$$

where the elements of matrix A are the known coefficients of the system of equations, $\{x\}$ the vector of unknown pressures and $\{c\}$ the vector of constants or known values.

In hygIRC, the known pressure at the boundary is not directly introduced in Eq. 69 for cells with a face at the boundary. Instead, a similar equation is considered for a boundary node with all the coefficients equal to zero, with the exception of $a_{i,j}$. This leads to a system on N1 x N2 equations where N1 is the number of nodes in x direction and N2 the number of nodes in y direction. For the 1D case, N2 is equal to 3.

The LU decomposition and back substitution are then used to solve the system of algebraic equations. The results are stored in the array P(N1, N2).

7.2.2 Calculation of air velocity

The pressure values in the array P are used to calculate the velocities at the cell interfaces as:

$$V_{x(i,j)} = Kx_{i,j}\left(P_{i,j} - P_{i+1,j}\right)$$
$$V_{y(i,j)} = Ky_{i,j}\left[\left(P_{i,j} - P_{i,j+1}\right) - \rho_a g \Delta y\right] \tag{71}$$

where K is the permeance, g the gravity and ρ_a is computed as:

$$\rho_a = \frac{P_0 M_a}{R_a\left(\frac{T_{i,j} + T_{i,j+1}}{2} + 273\right)} \tag{72}$$

7.3 Moisture and heat transfer equations

The moisture and heat transfer equations are solved using the delta-form of the approximate factorization which is described in Annex III. More details about the implementation can be found in ANNEX II.

7.3.1 Discretization of moisture equation

The moisture transfer equation (Eq. 3) can be rewritten as:

$$\rho_o \frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = Q_m \tag{73}$$

where F and G are the moisture fluxes in x and y directions, respectively:

$$F = -\rho_o D_{w,x} \frac{\partial u}{\partial x} + k_{w,x}\rho_w g - \delta_{P,x} \frac{\partial p_v}{\partial x} + \rho_v V_x$$

$$G = -\rho_o D_{w,y} \frac{\partial u}{\partial y} + k_{w,y}\rho_w g - \delta_{P,y} \frac{\partial p_v}{\partial y} + \rho_v V_y$$

Time integration and factorization following the steps described in Annex III give:

$$\left[\left(I + \frac{\theta \Delta t}{\rho_o} \frac{\partial}{\partial x} A_x^n\right)\left(I + \frac{\theta \Delta t}{\rho_o} \frac{\partial}{\partial y} A_y^n\right)\right] \Delta u^{n+1} = \frac{\Delta t}{\rho_o}\left[-\frac{\partial \mathrm{F}}{\partial x} - \frac{\partial \mathrm{G}}{\partial y} + Q_m\right]^n \tag{74}$$

with:

$$A_x^n = \frac{\partial}{\partial u}\left(-\rho_o D_{w,x} \frac{\partial u}{\partial x} + k_{w,x}\rho_w g - \delta_{P,x} \frac{\partial p_v}{\partial x} + \rho_v V_x\right)$$

$$A_y^n = \frac{\partial}{\partial u}\left(-\rho_o D_{w,y} \frac{\partial u}{\partial x} + k_{w,y}\rho_w g - \delta_{P,y} \frac{\partial p_v}{\partial x} + \rho_v V_y\right) \tag{75}$$

θ is the time approximation weighting factor ($\theta = 0$: Euler explicit scheme; $\theta = 1/2$: Crank-Nicholson scheme; $\theta = 1$: Euler implicit scheme).

<u>Discretization in the first direction (x sweep):</u>

$$\left(1 + \frac{\theta \Delta t}{\rho_o} \frac{\partial}{\partial x} A_x^n\right) z = \frac{\Delta t}{\rho_o}\left[-\frac{\partial \mathrm{F}^n}{\partial x} - \frac{\partial \mathrm{F}^n}{\partial y} + Q_m^n\right] \tag{76}$$

Integrating of 76 over the control volume (Figure 2), we get:

$$\int_s^n \int_w^e z\, dxdy + \frac{\theta \Delta t}{\rho_o} \int_s^n \int_w^e \frac{\partial}{\partial x} A_x^n z\, dxdy \tag{77}$$

$$= \frac{\Delta t}{\rho_o}\left[-\int_s^n \int_w^e \frac{\partial \mathrm{F}^n}{\partial x} dxdy - \int_w^e \int_s^n \frac{\partial \mathrm{G}^n}{\partial y} dydx + \int_s^n \int_w^e Q_m^n \, dxdy\right]$$

Or:

$$z_P \Delta x_c \Delta y_c + \frac{\theta \Delta t}{\rho_o}[(A_x^n z)_e - (A_x^n z)_w]\Delta y_c$$
$$= \frac{\Delta t}{\rho_o}\{-[(\mathrm{F}^n)_e - (\mathrm{F}^n)_w]\Delta y_c - [(\mathrm{G}^n)_n - (\mathrm{G}^n)_s]\Delta x_c + Q_m^n \Delta x_c \Delta y_c\} \tag{78}$$

Dividing both sides by $V_c = \Delta x_c . \Delta y_c$ (cell volume):

$$z_P + \frac{\theta \Delta t}{V_c \rho_o}[(A_x^n z)_e - (A_x^n z)_w]\Delta y_c$$
$$= \frac{\Delta t}{V_c \rho_o}\{-[(\mathrm{F}^n)_e - (\mathrm{F}^n)_w]\Delta x_c - [(\mathrm{G}^n)_n - (\mathrm{G}^n)_s]\Delta y_c + Q_m^n\} \tag{79}$$

Δx_c and Δy_c are the cell dimensions in x and y direction, respectively.

The approximation of A_x and the intermediate solution z at the east and west faces of the control volume leads to:

$$a_P z_P = a_E z_E + a_W z_W + RHS_P \tag{80}$$

where RHS_P is the right hand side of equation 79 computed at point P, and:

$$a_P = 1 - (a_P + a_W)$$

RHS_P contains the fixed value of the source term. Expressions for a_E and a_W depend on the scheme used to approximate A_x and z at the cell faces, especially the convective terms. After computing the coefficients and solving for z which is the final solution in the case of 1D problem, the following system is integrated, discretized and solved for the final solution Δu:

$$\left(1 + \frac{\theta \Delta t}{\rho_o}\frac{\partial}{\partial y} A_y^n\right)\Delta u^{n+1} = z \tag{81}$$

7.3.2 Discretization of the heat equation

The heat transport equation (Eq. 7) can be rewritten in the case of a 2D problem as:

$$\rho_o c \frac{\partial T}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} + Q_L = Q_h \tag{82}$$

where F and G are the total heat fluxes in x and y direction:

$$F = \rho_a c_a V_x T + \lambda_x \frac{\partial T}{\partial x} + \rho_o L_v \delta_{P,x} \frac{\partial p_v}{\partial x}$$

$$G = \rho_a c_a V_y T + \lambda_y \frac{\partial T}{\partial y} + \rho_o L_v \delta_{P,y} \frac{\partial p_v}{\partial y}$$

Q_L is the heat source due to the freezing of water or melting of ice:

$$Q_L = L_i \left(\rho_o u \frac{\partial f_l}{\partial t} \right)$$

The fraction of liquid, f_l, is defined in the program as a function of the temperature only:

$$\begin{cases} f_l = 0 \; if \; T \leq -3°\text{C} \\ f_l = 1 \; if \; T \geq 0°\text{C} \\ f_l \; varies \; between \; 1 \; and \; 0 \; as \; a \; 4th \; order \; polynomial \; when \; T \; varies \; between \; 0 \; and - 3°\text{C} \end{cases}$$

After time integration and factorization, we get:

$$\left[\left(I + \frac{\theta \Delta t}{\rho_o c} \frac{\partial}{\partial x} A_x^n + H^n \right) \left(I + \frac{\theta \Delta t}{\rho_o c} \frac{\partial}{\partial y} A_y^n \right) \right] \Delta T^{n+1} = \frac{\Delta t}{\rho_o c} \left[-\frac{\partial \mathrm{F}^n}{\partial x} - \frac{\partial \mathrm{G}^n}{\partial y} - Q_L^n + Q_h^n \right] \quad (83)$$

where:

$$A_x^n = \frac{\partial F}{\partial T} = \frac{\partial}{\partial T} \left(\rho_a c_a V_x \frac{\partial T}{\partial x} + \lambda_x \frac{\partial T}{\partial x} + \rho_o L_v \delta_{P,x} \frac{\partial p_v}{\partial x} \right)$$

$$A_y^n = \frac{\partial G}{\partial T} = \frac{\partial}{\partial T} \left(\rho_a c_a V_y \frac{\partial T}{\partial y} + \lambda_y \frac{\partial T}{\partial y} + \rho_o L_v \delta_{P,y} \frac{\partial p_v}{\partial y} \right) \quad (84)$$

$$H^n = \frac{\partial Q_L}{\partial T}$$

As was completed for the moisture equation, the heat equation is solved in two steps:

Step 1: Solve for intermediate solution z

$$\left(I + \frac{\theta \Delta t}{\rho_o c} \frac{\partial}{\partial x} A_x^n + H^n \right) z = \frac{\Delta t}{\rho_o c} \left[-\frac{\partial \mathrm{F}^n}{\partial x} - \frac{\partial \mathrm{G}^n}{\partial y} - Q_L^n + Q_h^n \right] \quad (85)$$

Spatial integration over the control volume leads to:

$$z_P(1+H^n)+\frac{\theta\Delta t}{\rho_o}[(A_x^n z)_e-(A_x^n z)_w]\Delta x = \frac{\Delta t}{\rho_o}\{-[(\mathrm{F}^n)_e-(\mathrm{F}^n)_w]\Delta x\ -[(\mathrm{G}^n)_n-(\mathrm{G}^n)_s]\Delta y-Q_L^n+Q_h^n\} \tag{86}$$

Approximations of A_x and z at the cell faces lead in this case to the following system to be solved for discretization in the first direction (x-sweep):

$$a_P z_P = a_E z_E + a_W z_W + RHS_P \tag{87}$$

where:

$$a_P = 1-(a_P+a_W)+H$$

Then for y-sweep, the final solution is found by solving:

$$\left(I+\frac{\theta\Delta t}{\rho_o c}\frac{\partial}{\partial y}A_y^n\right)\Delta T^{n+1}=z \tag{88}$$

7.3.3 Computation of fluxes at the cell faces

Referring to the 1D case of Figure 3; different scenarios must be considered that include: (i) air-solid interface (outdoor); (ii) solid-solid interface in the same material; (iii) solid-solid interface between different materials, and; (iv) solid-air interface (indoor).

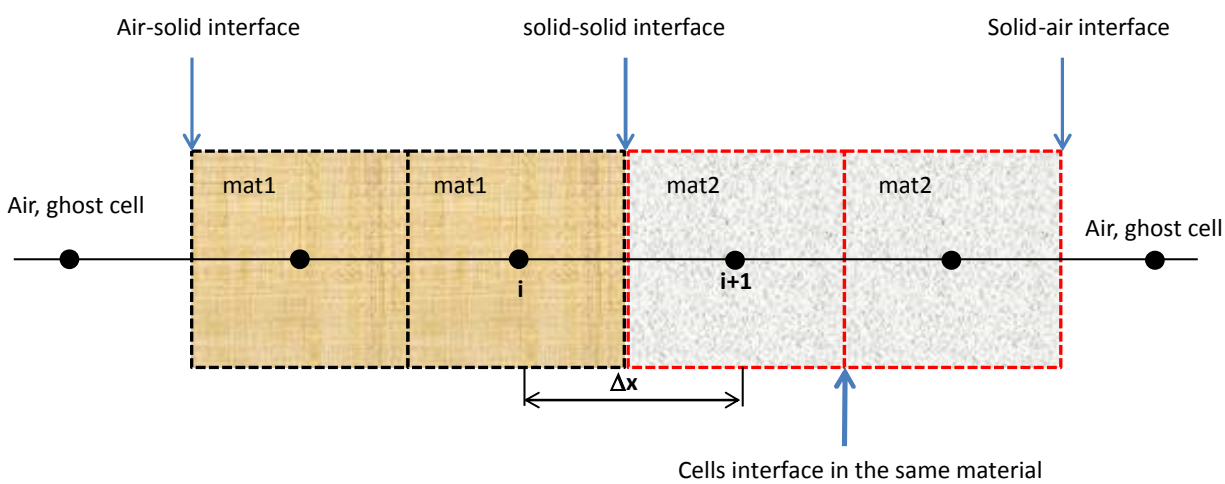

Figure 3. Arrangement of cells in x direction.

7.3.3.1 Moisture flux

Only the flux in the x direction is described, for which the total moisture flux is the sum of the fluxes derived, respectively, for liquid and vapour phases:

$$q_m = q_l + q_v = -\rho_o D_{w,x} \frac{\partial u}{\partial x} + k_{w,x}\rho_w g - \delta_{P,x} \frac{\partial p_v}{\partial x} + \rho_v V_{a,x} \tag{89}$$

where:

$$q_l = -\rho_o D_{w,x} \frac{\partial u}{\partial x} + k_{w,x}\rho_w g$$

$$q_v = -\delta_P \frac{\partial p_v}{\partial x} + \rho_v V_{a,x}$$

7.3.3.1 i) Case for two adjacent cells belong to the same material

For the case where the two adjacent cells, i and i+1, belong to the same material, the liquid moisture flux at the interface of these two cells is calculated as:

$$q_l = \rho_o D_w \frac{u_i - u_{i+1}}{\Delta x} + \rho_w \rho_o \bar{D}_L \frac{\partial u}{\partial s} g \cos\left(\frac{\beta\pi}{180}\right) \tag{90}$$

where β is the wall inclination angle and $\bar{D}_L$ the average liquid water diffusivity. Due to the convection term, a particular scheme is used to estimate q_v at the interface:

$$q_v = -\delta_{p,x}\beta \frac{p_{v,i+1} - p_{v,i}}{\Delta x} + \frac{M_w}{R(\bar{T} + 273)} V_{a,x} \left[(0.5 + \alpha)p_{v,i} + (0.5 - \alpha)p_{v,i+1}\right] \tag{91}$$

where R is the gas constant, M_w is the molecular mass of water and $\bar{T}$ is the average temperature at nodes i and i+1, Δx is the distance between nodes i and i+1, $\delta_{p,x}$ is the vapour permeability at the interface, p_v is the vapour pressure, and α and β are computed as:

$$\alpha = \frac{1 + 0.005{P_e}^2}{1 + 0.05P_e^2}$$

$$\beta = \frac{P_e \times |P_e|}{10 + 2{P_e}^2} \tag{92}$$

and for which P_e is the Peclet number, computed as:

$$P_e = \frac{\rho_v V_{a,x}}{\frac{\delta_{p,x}}{\Delta x}} = \frac{V_{a,x}\, M_w\, \Delta x}{\delta_{p,x} R(\bar{T} + 273)} \tag{93}$$

7.3.3.1 ii) Case for two adjacent cells belong to different materials

When the two adjacent cells belong to different materials, there is a special algorithm used to compute the flux at the interface. It consists of searching iteratively for the partial vapour

pressure at the interface that equates the total moisture flux upstream (in material-1) and downstream (in material-2); the algorithm is as follows:

1) Get or calculate the temperature at the interface of material-1 and-2
2) Start with a given *RH*, calculate the vapour pressure at the interface, $p_{v,f}$.
3) Calculate the total moisture flux QM1, from point i to the interface in material-1
4) Calculate the total moisture flux QM2, from the interface to point i+1 in material-2
5) Calculate the difference in moisture flux between materials, F = QM1 - QM2
6) Change *RH* and return to step 2 and repeat until F is less than 10^{-13}

Before starting the loop, the temperature at the interface, T_f, is retrieved from the array TSURF if the interface is at the boundary of the domain, or calculated if the interface is inside the wall, using the following formula:

$$T_f = \frac{a_i T_i + a_{i+1} T_{i+1} + L_v (q_l - q_v)}{a_i + a_{i+1}} \tag{94}$$

with:

$$a_i = \frac{\lambda_i}{\Delta x_{1/2}} + \max(V_f, 0)$$

$$a_{i+1} = \frac{\lambda_{i+1}}{\Delta x_{1/2}} - \max(V_f, 0)$$

And where $\Delta x_{1/2}$ is the half distance between the two nodal points, λ is the thermal conductivity, q_l and q_v are the liquid and vapour flux values obtained at previous time step. If material i is air, the thermal conductivity is replaced by the convective heat transfer coefficient. T_f is then used to calculate the saturated vapour pressure at the interface from which the partial vapour pressure at the interface is calculated from the relative humidity value tested at each iteration.

7.3.3.1 iii) Case for material-1 as being a solid

For the case where one of the two adjacent cells is a solid material, the total moisture flux is:

$$q_{m1} = \rho_o D_w \frac{u_i - u_f}{\Delta x_{1/2}} + \rho_w \rho_o \overline{D}_L \frac{\partial u}{\partial s} g \cos\left(\frac{\beta \pi}{180}\right) + \delta_{p,f} \frac{p_{v,i} - p_{v,f}}{\Delta x_{1/2}} + \frac{V_f \, M_w p_{v,i}}{R \, (T_f + 273)} \tag{95}$$

where subscript *f* is the value at the interface and $\delta_{p,f}$ is interpolated from the vapour permeability and relative humidity table using T_f and RH_f. In instances when V_f is negative, the convective term is calculated using $p_{v,f}$ rather than $p_{v,i}$. The value of moisture content, u_f, at the face of cell i is calculated as the average between the moisture content u of the material (u_i) and the moisture content derived from the sorption curve using the value of the relative humidity that is being tested.

7.3.3.1 iv) Case for material-1 as being air (outdoor boundary)

For the case where material-1 is air, In the liquid flux is zero and:

$$q_{m1} = \beta_e(p_{v,a} - p_{v,f}) + \frac{V_f\, M_w p_{v,a}}{R\,(T_a + 273)} \tag{96}$$

for which $p_{v,a}$ is the vapour pressure of the air. Again, if the value of V_f is negative, the convective term is calculated using $p_{v,f}$ and T_f.

7.3.3.1 v) Case for material-2 as being a solid

The value of the total moisture flux for material-2, q_{m2}, is computed using an equation similar to Eq. 95, with the exception that the position i is now the interface and i+1 is the node within material-2.

7.3.3.1 vi) Case material-2 as being air (e.g. indoor boundary)

The value of the total moisture flux for material-2 being air is given as:

$$q_{m2} = \beta_i(p_{v,f} - p_{v,a}) + \frac{V_f\, M_w p_{v,f}}{R\,(T_f + 273)} \tag{97}$$

where β_i is the indoor moisture transfer coefficient. If the value of V_f is negative, the convective term is computed using p_{vi+1} and T_{i+1}.

<u>Note</u>:

- Gravity outflow: in case the interface is at the left of the domain, the term $\Delta q_{lg} = q_{lg2} - q_{lg1}$ is added to q_{m1}. The moisture flows due to gravity in material-1 and material-2 are, respectively, given as q_{lg1} and q_{lg2}.
- For the external boundary and for the nodes below the top node, the rain flux and rain rundown (if any) are taken into account by adding their values to F.

7.3.3.2 Heat flux

For the x direction, the heat flux is expressed as:

$$q_{h,x} = -\lambda_x \frac{\partial T}{\partial x} - L_v \rho_o \delta_{P,x} \frac{\partial p_v}{\partial x} + \rho_a c_a V_{a,x} T \tag{98}$$

7.3.3.2 i) Case for solid-solid interface or two adjacent cells belonging to same material

The heat flux, q_h at the interface between two adjacent cells of the same material is given as:

$$q_h = \beta \frac{\lambda_f (T_i - T_{i+1})}{\Delta x} + L_v q_v + \rho_a c_a V_{x,f}[(0.5 + \alpha)T_i + (0.5 - \alpha)T_{i+1}] \tag{99}$$

where λ_f is the thermal conductivity at the interface between the two cells (when the two materials are different, λ_f is the harmonic mean of the two thermal conductivities), α (same as in Eq. 92), β, a correction factor (same as in Eq. 92), but with P_e now defined as:

$$P_e = \frac{V_{x,f}\ \rho_a\ c_a \Delta x}{\lambda_f}$$

7.3.3.2 ii) Case for air-solid (outside wall surface)

The long-wave radiation exchange between the surface and the sky is taken into account at the air-solid interface so that the total heat flux is:

$$q_h = h_e(T_e - T_S) - \varepsilon\sigma\left({T_S}^4 - {T_{sky}}^4\right) + \beta_e\left(p_{v,a} - p_{v,f}\right)L_v$$
$$+\rho_a c_a \left[V_f T_X - \max(V_f, 0)\ \Delta T_o\right] \tag{100}$$

in which:

$$\begin{cases} T_X = T_{S,e}\ if\ V_f\ < 0 \\ T_X = T_{e,o}\ if\ V_f\ > 0 \end{cases}$$

$$\begin{cases} \Delta T_o = T_e - T_a\ if\ V_f > 0 \\ \Delta T_o = 0\ otherwise \end{cases}$$

where:

h_e: equivalent heat transfer coefficient (W/m^{2}K)

T_e: sol-air or effective outdoor temperature (°C)

T_S: effective surface temperature (K)

ε: emissivity of the wall

σ: Stefan-Boltzmann constant

T_{sky}: sky temperature (K)

$p_{v,a}$: vapour pressure of the ambient air (Pa)

$p_{v,f}$: vapour pressure at te interface (Pa)

7.3.3.2 iii) Case for solid-air (indoor wall surface)

The total heat flux solid-air interface is given as:

$$q_h = \frac{\lambda_i}{\Delta x_{1/2}}\left(T_i - T_{S,in}\right) + \varepsilon\sigma\left({T_{S,in}}^4 - {T_{sky}}^4\right) + \beta_i\left(p_{v,in} - p_{v,f}\right)L_v + \rho_a c_a\left[V_f T_X - \max\left(V_f, 0\right)\Delta T_{in}\right] \quad (101)$$

in which:

$$\begin{cases} \Delta T_{in} = -\left(T_{e,in} - T_{a,in}\right)\ if\ V_f < 0 \\ \Delta T_{in} = 0\ otherwise \end{cases}$$

$$\begin{cases} T_X = T_{S,in}\ if\ V_f\ < 0 \\ T_X = T_{e,in}\ if\ V_f\ > 0 \end{cases}$$

where:

$T_{e,in}$: effective or equivalent indoor temperature

$T_{a,in}$: indoor air temperature

λ_i: thermal conductivity computed at node i

$T_{S,in}$: indoor equivalent surface temperature

T_i: temperature at the node i

Eq. 101 represents the implementation of the heat flux at the indoor surface. The first term describes the conduction from node i to the internal surface of the wall. We were expecting a convective term using indoor equivalent heat transfer coefficient.

7.3.4 Computation of the right hand side of equations 82 and 89

7.3.4.1 Case for external nodes

7.3.4.1 i) Moisture equation

$$RHS(1, i, j) = p_v - {p_v}^n \quad (102)$$

p_v is the value of boundary vapour pressure at current iteration and p_v^n is the value of the boundary vapour pressure at previous time. The result is nul since boundary p_v remains unchanged throughout the iteration process.

7.3.4.1 ii) Heat equation

$$RHS(2, i, j) = T - T^n \quad (103)$$

T is the equivalent outdoor temperature at the current iteration. The result is nul since boundary T remains unchanged throughout the iteration process.

7.3.4.2 Case for internal nodes

7.3.4.2 i) Moisture equation

$$RHS(1,i,j) = \frac{\Delta t}{\rho_o}\{(\alpha_x q_{m,w}\Delta x_w - \beta_x q_{m,e}\Delta x_e) + (\alpha_y q_{m,s}\Delta y_s - \beta_y q_{m,n}\Delta y_n) + Q_m\} - (u - u^n) \tag{104}$$

u is the moisture content at current iteration, u^n is the moisture content at previous time step, the subscripts w, e, s, and n represent the cell west, east, south and north face, respectively, and α and β are defined as:

$$\alpha_x = \frac{2}{\Delta x_w(\Delta x_w + \Delta x_e)}$$

$$\beta_x = \frac{2}{\Delta x_e(\Delta x_w + \Delta x_e)}$$

$$\alpha_y = \frac{2}{\Delta y_s(\Delta y_s + \Delta y_n)}$$

$$\beta_y = \frac{2}{\Delta y_n(\Delta y_s + \Delta y_n)}$$

7.3.4.2 ii) Heat equation

$$RHS(2,i,j) = \frac{\Delta t}{\rho_o c}\left[(\alpha_x q_{h,w}\Delta x_w - \beta_x q_{h,e}\Delta x_e - q_{\varepsilon,x}) + (\alpha_y q_{h,s}\Delta y_s - \beta_y q_{h,n}\Delta y_n - q_{\varepsilon,y}) + Q_h - \rho_o u^n L_i \frac{\Delta f_l}{\Delta t}\right] - (T - T^n) \tag{105}$$

where $\Delta f_l = f_l^{n+1} - f_l^{n}$, T is the temperature at point P at current iteration, T^n is the temperature at previous time step at node point P. q_ε represents the internal surface to surface radiation and is calculated as:

$$q_{\varepsilon,x} = 5.67 F_x F_{\varepsilon,x}\left[\left(\frac{T_i + 273}{100}\right)^4 - \left(\frac{T_{i+ICR} + 273}{100}\right)^4\right] \tag{106}$$

where T_i is the temperature at node P, T_{i+ICR} is the temperature at node at position i+ICR, ICR being the position of the layer with which layer i exchanges radiations, f is the liquid fraction at current iteration, f_n is the liquid fraction at previous time step, F and F_ε are geometric and the emissivity factors calculated as:

$$F_\varepsilon = \frac{1}{\frac{100}{\varepsilon_i} + \frac{100}{\varepsilon_{i+ICR}} - 1}$$

$$\begin{cases} F_x = \beta_x \Delta x_e \; if \; ICR > 0 \\ F_x = \alpha_x \Delta x_w \; if \; ICR < 0 \end{cases}$$

β and α are the same as above. Similar calculations are performed for $q_{\varepsilon,y}$.

7.3.5 Computation of coefficients

Only the coefficients for x-sweep will be presented for the 1D case when j = 2. For j = 1 or 3 (bottom and top nodes), all coefficients are nul except a_P which have a value of 1.

In the program, the coefficients are computed, "interface by interface", which implies that for an interface between cell i and i+1, A_w of i+1 and A_e of i are calculated. Different scenarios are considered: (i) air-solid interface; (ii) solid-solid interface in the same material; (iii) solid-solid interface with i and i+1 belonging to different materials, and; (iv) solid-air interface.

7.3.5.1 Moisture equation

7.3.5.1 i) Case for an air-solid interface (external boundary)

$$A_W(i+1) = \left[-\beta + \beta\left(\frac{\partial p_v}{\partial u}\right)_f \left(\frac{\beta}{D1}\right)\right]\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{107}$$

$$A_E(i) = -\beta\left(\frac{\partial p_v}{\partial u}\right)_f \frac{\left(\frac{\rho_{o2} D_{L2}}{0.5\,\Delta x_i} + \frac{\delta_{p2}}{0.5\,\Delta x_i} + \rho_{v1}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_2}\right)}{D1}\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{108}$$

with:

$$D1 = \beta\left(\frac{\partial p_v}{\partial u}\right)_f + \left[\frac{\rho_{o2} D_{L2}}{0.5\,\Delta x_i} + \frac{\delta_{p2}}{0.5\,\Delta x_i} + \rho_{v1}\left|V_{a,f}\right|\left(\frac{\partial p_v}{\partial u}\right)_f\right] \tag{109}$$

$$\beta = \beta_e + \rho_v\left|V_{a,f}\right| \tag{110}$$

$\left(\frac{\partial p_v}{\partial u}\right)_f$ means the value at the face interface; $\frac{\partial p_v}{\partial u_2}$ is the derivative computed using temperature and moisture content at node i+1 in material 2; $\left|V_{a,f}\right|$ means the absolute value of the air velocity at the interface; ρ_{o2} is the density of material 2; D_L is the liquid water diffusivity; β_e is the mass transfer coefficient; and ρ_v is the vapour density and is calculated using the air temperature. $\left(\frac{\Delta y_1 + \Delta y_2}{2}\right)$ is the east or west area of the cell.

7.3.5.1 ii) Case for a solid-air interface (indoor boundary)

$$A_W(i+1) = -\beta\left(\frac{\partial p_v}{\partial u}\right)_f \frac{\left(\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_i} + \frac{\delta_{p1}}{0.5\,\Delta x_i} + \rho_{v2}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_1}\right)}{D2}\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{111}$$

$$A_E(i) = -\beta + \beta\left(\frac{\partial p_v}{\partial u}\right)_f \left(\frac{\beta}{D2}\right)\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{112}$$

With:

$$D2 = \beta\left(\frac{\partial p_v}{\partial u}\right)_f + \left[\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_i} + \frac{\delta_{p1}}{0.5\,\Delta x_i} + \rho_{v2}\left|V_{a,f}\right|\left(\frac{\partial p_v}{\partial u}\right)_f\right] \tag{113}$$

$$\beta = \beta_i + \rho_v\left|V_{a,f}\right| \tag{114}$$

7.3.5.1 iii) Case for a solid-solid interface in the same material

$$A_w(i+1) = \frac{-\rho_0 \overline{D}_L}{\Delta x_i} - \left[\left(\frac{\delta_{p1}}{\Delta x_i} + \max\left(V_{a,f}, 0\right)\rho_{v1}\right)\frac{\partial p_v}{\partial u_1}\right]\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{115}$$

$$A_e(i) = \frac{-\rho_0 \overline{D}_L}{\Delta x_i} - \left[\left(\frac{\delta_{p1}}{\Delta x_i} - \min\left(V_{a,f}, 0\right)\rho_{v2}\right)\frac{\partial p_v}{\partial u_2}\right]\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{116}$$

where $\overline{D}_L$ is the average liquid diffusivity at node i and i+1.

7.3.5.1 iv) Case for a solid-solid interface but with different materials

$$A_W(i+1) = -\left(\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_{c1}} + \frac{\delta_{p1}}{0.5\,\Delta x_{c1}} + \rho_{v1}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_1}\right)$$
$$\left(1 - R_{pv}\frac{\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_{c1}} + \frac{\delta_{p1}}{0.5\,\Delta x_{c1}} + \rho_{v1}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_1}}{D}\right)\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{117}$$

$$A_E(i) = -\left(\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_{c1}} + \frac{\delta_{p1}}{0.5\,\Delta x_{c1}}\rho_{v1}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_1}\right)R_{pv}$$
$$\left(\frac{\frac{\rho_{o2} D_{L2}}{0.5\,\Delta x_{c2}} + \frac{\delta_{p2}}{0.5\,\Delta x_{c2}} + \rho_{v2}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_2}}{D}\right)\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{118}$$

with:

$$D = \left(\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_{c1}} + \frac{\delta_{p1}}{0.5\,\Delta x_{c1}} + \rho_{v1}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_1}\right)R_{pv} \tag{119}$$

$$+\left(\frac{\rho_{o2}D_{L2}}{0.5\,\Delta x_{c2}}+\frac{\delta_{p2}}{0.5\,\Delta x_{c2}}+\rho_{v2}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_2}\right)$$

$$R_{pv}=\frac{\frac{\partial p_v}{\partial u_2}}{\frac{\partial p_v}{\partial u_1}} \tag{120}$$

Δx_c is the cell size, subscript 1 refers to value at node 1 (i) and 2 refers to value at node 2 (i+1).

7.3.5.1 v) Final coefficients

After computing the coefficients A_W and A_E at the interfaces, the coefficients a_W, a_E and a_P for an internal node i are computed as:

$$a_w(i)=\frac{\Delta t}{\rho_o V_i}A_W(i-1) \tag{121}$$

$$a_E(i)=\frac{\Delta t}{\rho_o V_i}A_E(i) \tag{122}$$

$$a_P(i)=1-(a_W+a_P)\frac{\Delta t}{\rho_o V_i} \tag{123}$$

where $A_W(i-1)$ is the value of A_W at the interface i-1 and $A_E(i)$is the value of A_E at the interface i, and V_i is the cell volume computed as:

$$V_i=\left(\frac{1}{2}\Delta x_{i-1}+\frac{1}{2}\Delta x_i\right)\left(\frac{1}{2}\Delta y_{i-1}+\frac{1}{2}\Delta y_i\right) \tag{124}$$

7.3.5.2 Heat equation

7.3.5.2 i) Case for an air-solid interface (outdoor boundary)

$$A_w(i+1)=\frac{-\bar{\lambda}_h}{\Delta x_1}-\max\left(V_{a,f},0\right)\rho_a c_{pa}-l\,L_{v1}\,C1\left(\frac{\Delta y_1+\Delta y_2}{2}\right) \tag{125}$$

$$A_e(i)=\frac{-\bar{\lambda}_h}{\Delta x_1}+\min\left(V_{a,f},0\right)\rho_a c_{pa}+l\,L_{v1}\,C2\left(\frac{\Delta y_1+\Delta y_2}{2}\right) \tag{126}$$

with:

$$C1=\left[-\beta\left(\frac{\partial p_v}{\partial u}\right)_f\left(\frac{-\beta\frac{\partial p_v}{\partial T_S}A-\left[\left(\frac{\delta_{p2}}{0.5\,\Delta x_i}+\rho_{v1}\left|V_{a,f}\right|\right)A\right]}{D1}\right)\right]-\left[\left(\beta\frac{\partial p_v}{\partial T_S}\right)A\right] \tag{127}$$

$$C2 = \left[-\beta \left(\frac{\partial p_v}{\partial u}\right)_f \left(\frac{-\beta \frac{\partial p_v}{\partial T_S} B - \left[\left(\frac{\delta_{p2}}{0.5\,\Delta x_i} + \rho_{v1}\left|V_{a,f}\right|\right) B\right]}{D1} \right) \right] - \left[\left(\beta \frac{\partial p_v}{\partial T_S}\right) B\right] \tag{128}$$

$$A = \frac{h_e + \rho_a c_{pa}\left|V_{a,f}\right|}{\left(h_e + \rho_a c_{pa}\left|V_{a,f}\right|\right) + \left(\frac{\lambda_2}{0.5\Delta x_i} + \rho_a c_{pa}\left|V_{a,f}\right|\right)} \tag{129}$$

$$B = \frac{\frac{\lambda_2}{0.5\Delta x_i} + \rho_a c_{pa}\left|V_{a,f}\right|}{\left(h_e + \rho_a c_{pa}\left|V_{a,f}\right|\right) + \left(\frac{\lambda_2}{0.5\Delta x_i} + \rho_a c_{pa}\left|V_{a,f}\right|\right)} \tag{130}$$

$$\bar{\lambda}_h = \frac{\Delta x_{c1} + \Delta x_{c2}}{\frac{\Delta x_{c1}}{\Delta x_{1/2} h_e} + \frac{\Delta x_{c2}}{\lambda_2}} \tag{131}$$

$\frac{\partial p_v}{\partial T_S}$ is the derivative at the interface; β and D are the same as in Eqs. 110 and 109. l is always zero. That means the connection between moisture and temperature fields through latent heat no longer exists in the LHS but still in effect in the RHS.

7.3.5.2 ii) Case for a solid-air interface (indoor boundary)

$$A_w(i+1) = \frac{-\bar{\lambda}_h}{\Delta x_1} - \max\left(V_{a,f}, 0\right) \rho_a c_{pa} - l\, L_{v1}\, C1 \left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{132}$$

$$A_e(i) = \frac{-\bar{\lambda}_h}{\Delta x_1} + \min\left(V_{a,f}, 0\right) \rho_a c_{pa} + l\, L_{v1}\, C2 \left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{133}$$

with:

$$C1 = \left[\beta \left(\frac{\partial p_v}{\partial u}\right)_f \left(\frac{-\beta \frac{\partial p_v}{\partial T_S} B - \left[\left(\frac{\delta_{p1}}{0.5\,\Delta x_i} + \rho_{v2}\left|V_{a,f}\right|\right) B\right]}{D2} \right) \right] + \left[\left(\beta \frac{\partial p_v}{\partial T_S}\right) B\right] \tag{134}$$

$$C2 = \left[\beta \left(\frac{\partial p_v}{\partial u}\right)_f \left(\frac{-\beta \frac{\partial p_v}{\partial T_S} A - \left[\left(\frac{\delta_{p1}}{0.5\,\Delta x_i} + \rho_{v2}\left|V_{a,f}\right|\right) A\right]}{D2} \right) \right] + \left[\left(\beta \frac{\partial p_v}{\partial T_S}\right) A\right] \tag{135}$$

$$\bar{\lambda}_h = \frac{\Delta x_{c1} + \Delta x_{c2}}{\frac{\Delta x_{c1}}{\lambda_1} + \frac{\Delta x_{c2}}{\Delta x_{1/2} h_i}} \tag{136}$$

7.3.5.2 iii) Case for a solid-solid interface

$$A_w(i+1) = -\left(\frac{\lambda_1}{\Delta x_i} + \max(V_{a,f}, 0)\rho_a c_{pa}\right)\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{137}$$

$$A_e(i) = -\left(\frac{\lambda_1}{\Delta x_i} - \min(V_{a,f}, 0)\rho_a c_{pa}\right)\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{138}$$

It should be noted that λ_1 is the harmonic mean of the thermal conductivity at nodes i and i+1.

7.3.5.2 iv) Case for a solid-solid interface with different materials

$$A_w(i+1) = \left[\frac{-\bar{\lambda}_h}{\Delta x_1} - \max\left(V_{a,f}, 0\right)\rho_a c_{pa}\right]\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{139}$$

$$A_{eT}(i) = \left[\frac{-\bar{\lambda}_h}{\Delta x_1} + \min\left(V_{a,f}, 0\right)\rho_a c_{pa}\right]\left(\frac{\Delta y_1 + \Delta y_2}{2}\right) \tag{140}$$

Where:

$$\bar{\lambda}_h = \frac{\Delta x_{c1} + \Delta x_{c2}}{\frac{\Delta x_{c1}}{\lambda_1} + \frac{\Delta x_{c2}}{\lambda_2}} \tag{141}$$

7.3.5.2 v) Final coefficients

The final coefficients are computed as:

$$a_w(i) = \frac{\Delta t}{\rho_o c\, V_i} A_W(i-1) \tag{142}$$

$$a_E(i) = \frac{\Delta t}{\rho_o c\, V_i} A_E(i) \tag{143}$$

$$a_P(i) = 1 - (a_W + a_P)\frac{\Delta t}{\rho_o c\, V_i} + \frac{\rho_o u_i L_{ice}}{\rho_o c}\frac{\partial f_l}{\partial T} \tag{144}$$

7.3.6 Solutions of moisture and heat transfer equations

Assembling the coefficients leads to a tri-diagonal matrix that is solved in the same manner as for the pressure equation (see Eq #). At the end of each iteration the solution is updated as:

$$U_i = UITER_{i-1} + \Delta U_i \tag{145}$$

$$\begin{aligned} UITER_i &= (1-\alpha)UITER_{i-1} + \alpha(UITER_{i-1} + \Delta U_i) \\ &= UITER_{i-1} + \alpha \Delta U_i \end{aligned} \tag{146}$$

where α is the under-relaxation coefficient. The minimum is set to a value of 0.2 for moisture equation and to 0.3 for heat equation and these values varies with the time step and iteration number.

To verify for convergence, the limits are set for Δu, ΔT and ΔRH as follows:

$$\begin{cases} \Delta u_{max} = 0.001 \\ \Delta T_{max} = 0.1 \\ \Delta RH_{max} = 0.01 \end{cases}$$

As long as any of the values given for the incremental solutions, Δu, ΔT or ΔRH is greater than their respective maximum values and the number of iterations is less than the maximum iteration value (set to 15 for the first two time steps and 21 after), the iteration process continues.

The incremental solutions are the maximum values among nodal incremental solutions:

$$\begin{cases} \Delta u = \max\{|u_i - u_{i-1}|_{n=2}, |u_i - u_{i-1}|_{n=3}, \dots, |u_i - u_{i-1}|_{n=n-1}\} \\ \Delta T = \max\{|T_i - T_{i-1}|_{n=2}, |T_i - T_{i-1}|_{n=3}, \dots, |T_i - T_{i-1}|_{n=n-1}\} \\ \Delta RH = \max\{|RH_i - RH_{i-1}|_{n=2}, |RH_i - RH_{i-1}|_{n=3}, \dots, |RH_i - RH_{i-1}|_{n=n-1}\} \end{cases}$$

where n is the number of nodes, u and T are obtained from the solution array U, and RH is derived from the sorption curve.

Once convergence is reached, the solution arrays are updated and other variables such as the mold index, RHT, and freeze/thaw cycle are calculated and updated.

7.3.7 Other computations

7.3.7.1 Mould Index

HygIRC can be used to assess the risk to the formation of mould, based on the mould risk index model developed by Hukka and Viitanen (1999) and that was based on the work of Viitanen (1997). The algorithm used to assess the risk mould index, M_n(i), at node i and at time step n, and given the temperature T_i and the relative humidity RH_i at node i, is:

$$M_n(i) = min\{max[(M_{n-1}(i) + \Delta M_n \times \Delta t_d), \quad 0], \ 10\} \tag{147}$$

where:

ΔM_n : incremental mold index at time step n

M_{n-1}: cumulative mold index up to previous time step

M_n: cumulative mold index up to time step n

Δt_d: time step (**days**)

RH_c is computed as:

$$\begin{cases} RH_c = 1 - 3.13T_i + 0.16{T_i}^2 - 0.00267{T_i}^3 \\ \quad if\ T\ \leq 0°C, \qquad RH_c = 1 \\ \quad if\ T\ > 20°C, \qquad RH_c = 0.8 \end{cases} \tag{148}$$

The formula to calculate the incremental risk ΔM_n depends on the temperature T_i, and relative humidity RH_i.

<u>If $RH_i \leq RH_c$ or if $Ti < 0$ or $Ti > 50$ (unfavorable conditions)</u>

$$\begin{cases} \Delta M_n = -0.032 \quad if\ t_d\ \leq 0.25 \\ \Delta M_n = -0.016 \quad if\ t_d\ > 1 \\ \Delta M_n = 0 \qquad if\ 0.25 < t_d \leq 1 \end{cases} \tag{149}$$

where:

t_d: cumulative time duration (in days) where RH_i stays less than RH_c; reinitialized to 0 when RH_i greater than RH_c.

<u>If $RH_i > RH_c$ and Ti between 0 and 50 (favorable conditions)</u>

$$\Delta M_n = \frac{R1 \times R2}{7t_m} \tag{150}$$

in which t_m is the response time (in days) needed for the initiation of mold growth, defined as:

$$T_m = e^{[-0.68ln(T_i) - 13.9ln(100*RH_i) + 0.14I_{sp} - 0.33I_{sq} + 66.02]} \tag{151}$$

and R1 and R2 are correction factors defined as:

$$\begin{cases} R1 = \dfrac{2}{\dfrac{T_v}{T_m} - 1} \quad if\ M_{n-1}\ > 1 \\ R1 = 1 \quad if \quad M_{n-1}\ < 1 \quad (no\ growth) \end{cases} \tag{152}$$

$$R2 = \max\left[0,\ \ 1 - e^{[2.3 \times \min(M_{n-1} - M_{max},\ 0)]}\right] \tag{153}$$

where I_{sp} is the wood species index (0 for pine – default value set in subroutine MOLD, and 1 for spruce), and I_{sq} is the index for the wood quality (1 for kiln dry surface - default value set in subroutine MOLD, and 0 for re-sawn kiln dry wood surface).

Note that there is no option to modify these values in the GUI. T_v (time needed for the first visual appearance of mold) and M_{max} (maximum mold index) are defined as:

$$T_v = e^{[-0.74ln(T_i)-12.72ln(100*RH_i)+0.06I_{sp}+61.5]} \tag{154}$$

$$M_{max} = 1 + 9.4 \times \left(\frac{RH_c - RH_i}{RH_c - 1}\right) - 4.4 \times \left(\frac{RH_c - RH_i}{RH_c - 1}\right)^2 \tag{155}$$

7.3.7.2 RHT analysis

The user can define the type and the number of analyses required when using the working version of hygIRC; this is done in the file *Control.in.* When the simulation is set in the GUI, the default analyses are RHT80 and RHT95 where the minimum RH and T are 80% and 5°C and 95% and 5°C, respectively. There is no option for changing these values in the GUI.

At each time step and for each analysis (i.e. RHT80, RHT95) as well as for each internal node i, RHT is calculated as:

$$RHT = (RH - RH_{min}) \times (T - T_{min}) \times \Delta t/3600 \tag{156}$$

where Δt is the time step (s), RH and T are relative humidity and temperature of the node, RH_{min} and T_{min} are minimums relative humidity and temperature. If T > T_{min} or RH < RH_{min}, RHT = 0. The cumulative sum of RHT is also calculated:

$$RHT_{sum} = \sum_{i=istart}^{n} RHT_i \tag{157}$$

where n is the number of steps, *istart* is the starting hour set in the file *Control.in* to start computing the sum of the values obtained for RHT. At the end of the simulation, RHT_{sum} is written in the file *rht_index.out*.

For each output cycle (set to 1 at the beginning of the main file or read in file *Control.in*), the sum and the average value for RHT are calculated and written in binary file:

$$RHT_{sumOutput} = \sum_{i=1}^{n_c} RHT_i \tag{158}$$

$$RHT_{sumOutputAverage} = \frac{\sum_{i=1}^{n_c} RHT_i}{n_c} \tag{159}$$

where n_c is the total number of steps per cycle. At the end of each cycle, $RHT_{sumOutput}$ is reinitialized to 0.

7.3.7.3 Time of wetness

The time of wetness for each analysis (RHT80, RHT95) and for each internal node i is computed following two steps:

1) Sum all the steps for which *RH(i)* > *RHmin* and *T(i)* > *Tmin,* i.e.:

   Start: n_w (i) = 0

   For each step n ≥ n_{start}:

   If *RH(i)* > *RHmin* and *T(i)* > *Tmin*

   n_w (i) = n_w (i) + 1

2) When the simulation is completed, compute the time of wetness for each analysis as:

$$t_w(i) = \frac{n_w(i)}{n - n_{start}} \times RHT_{sum}(i) \tag{160}$$

where t_w is the time of wetness, n is the total number of steps, n_{start} is the starting hour (step) for summing RHT. The time of wetness is stored in the array RHTT. RHTT is written in the file *time_of_wetness.out*.

7.3.7.4 Freeze/thaw cycles

Assuming *Tf* is the critical temperature for freezing of a given porous material (e.g. -5°C), *Tt* the temperature for thawing (e.g. 0°C) and *RHc*, the critical value for RH (e.g. 80%), then one freeze/thaw cycle occurs when *T* drops below *Tf* and then rises above *Tt*. The algorithm to calculate the number of freeze/thaw cycles at node i is:

Start: isFrozen(i) = false; FreezeThawCycles(i) = 0

If at step n-1, the water was not in a frozen state (isFrozen(i) = false)

If at step n, *RH* ≥ *RHc* and *T* < *Tf*

Set isFrozen(i) to true

else at time step n-1, isFrozen(i) was true

If at step n, *T* > *Tt*

FreezeThawCycles(i) = FreezeThawCycle(i) + 1

Set IsFrozen(i) to false

The number of freeze/thaw cycles is written in the file: *freeze_thaw_cycles.out*.

8.0 General algorithm of hygIRC 1D

Read input data
Time step t + Δt
- Get weather data (WEATH)
- Compute boundary pressures (stack, wind, ventilation)
- **Air flow module (DARCY)**
  - For each cell face
    - Compute harmonic mean air permeability (GENAKV)
  - For each node
    - Compute coefficients a_E, a_W, a_S, a_N, and a_P
    - Get RHS
    - Assemble coefficients and RHS
  - Solve pressure equation (SOLVPRES)
  - Compute cell faces velocities (stored in VELO)
- Compute total radiation (SOLAR)
- Compute WDR (WINCORR)
- Compute effective outdoor T
  - Update U, UOLD, U000, UITER
  - Compute and update RHU using new effective T
- Read source file if any (store in QSRC)
- **Solve u and T equations (DUSTAR) (x-sweep)**
- **Iteration i + 1**
  - Update material properties (MATERP)
  - For each interface
    - Compute moisture fluxes (QM, QMOLDI)
    - Compute heat fluxes at cell faces (QT)
    - Compute flux derivative (COEFST)
  - For each node
    - Compute RHS (RIGHT)
    - Compute coefficients a_E, a_W and a_P
  - Assemble coefficients and RHS
  - Solve u and T equations (UMATSOL)
  - Check convergence
- **Next iteration**
- Update UOLD, U000 and UITER (INITMOIS)
- Compute Mold Index (MOLD)
- Perform RHT analysis (RHT,, Time of Wetness, Freeze/thaw)

Next step

Figure 4 - General algorithm of hygIRC 1D

9.0 Conclusions

After reviewing the implementation of hygIRC, there are a number of observations that can be highlighted; these include the following:

Transport equations

- The moisture diffusion due to thermal gradients is not included in the moisture equation; even if its effect is generally neglected in building science (Janssen 2011), it should be considered to ensure the moisture transport evaluation is not underestimated or overestimated, especially in the case where high temperature gradients are present.
- The latent heat transport due to the diffusion of vapour is considered in the computation of the heat flux. However, this term is not considered in the computation of the derivative $\partial q_h / \partial T$ which is used in the approximate factorization method for the Δ-form. Not sure what was the reason.
- The sensible heat of rain is not accounted in the heat equation; that could lead to an incorrect evaluation of heat transfer in case a large amount of rain is absorbed.

Material properties

- The air permeability, obtained by dividing the intrinsic permeability of material by the dynamic viscosity of air is not corrected for the degree of saturation of material, nor the level of pressure. For some materials that include micropores (e.g. openings in the membranes of punctuations in softwoods), the slip flow effect (Knudsen diffusion) is present which tends to increase the apparent permeability to gas.
- The temperature effect on moisture diffusivity is taken into account through viscosity of water.
- The sorption and suction are provided as a single curve for each material which is the average of desorption and absorption curve. Temperature effect is not considered. The sorption curve affects the storage capacity of the material (an indication for durability) whereas the suction curve is used in the program to compute the water conductivity from moisture diffusivity.

Some of the critical parameters that form part of the material properties can have significant effects on the simulation results. If the intent of the simulation is to perform comparisons (i.e. two locations, two sets of climate data, etc.), this then is less critical. However, for the purposes of design and for which a clear understanding of the effects of climate loads on building envelope are needed to assess the durability of the envelope components and assemblies, efforts should be made to properly characterize the material properties.

10. References

Beam, R.M., and Warming, R.F. 1978. An implicit factored scheme for the compressible Navier-Stokes equations. AIAA Journal, vol. 16(4):393-402.

Bird, R. B., Stewart W.E., and Lightfoot, E. N. 2002. Transport phenomena. 2nd Edition.Johns Wiley & Sons New York.

Burrows, J., and Gallagher, J.F. 2007. hygIRC software for designing better envelopes. Solplan Review, 134, p. 18.

Cornick, S., Maref, W., Abdulghani, K., van Reenen, D. 2003. 1-D hygIRC: a simulation tool for modeling heat, air and moisture movement in exterior walls. IRC Building Science Insight 2003 Seminars Series pp. 1-10.

Dean, W. M. 1998. Analysis of transport phenomena. Oxford University Press.

Djebbar, R. Kumaran, M.K., and Van Reenen, D. 2002. Use of hygrothermal numerical modeling to identify optimal retrofit options for high-rise buildings. 12th International Heat Transfer Conference, Grenoble, France, Sept. 18, 2002, pp. 165-170.

Djebbar, R., van Reenen, D., and Kumaran, M. K. 2001. "Indoor and outdoor weather analysis tool for hygrothermal modelling," 8th Conference on Building Science and Technology (Toronto, 2/22/2001), pp. 139-157, May 2001 (NRCC-44686) http://irc.nrc-cnrc.gc.ca/fulltext/nrcc44686/

Hens, H. 2012. Building physics. Heat, Air and Moisture. Fundamentals methods with examples and exercises. 2nd Edition. Ernst & Sons.

Hukka, A. and Viitanen, H.A., 1999. A mathematical model of mould growth on wooden material. *Wood Science and Technology*, *33*(6), pp.475-485.

Hutcheon, N.B. and Handegord, G.O.P. 1983. Building science for a cold climate. John Wiley & Sons.

Janssen, H. 2011. Thermal diffusion of water vapour in porous materials: fact or fiction. Int. J. Heat Mass Transf. 54(7/8):1548-1562.

Karagiozis, A. N.; Kumaran, M. K. 1993. Computer model calculations on the performance of vapor barriers in Canadian residential buildings. *ASHRAE Transactions, 99, 2, pp. 991-1003.*

Karagiosis, A. N. and Salonvaara. 1995. Influence of material properties on the hygrothermal performance of high-rise residential wall. ASHRAE Transactions, ASHRAE Symposium, Chicago, IL, USA, p. 647-655.

Maref, W., Cornick, S., Abdulghani, K., and van Reenen, D. 2004. An advanced hygrothermal design tool “hygIRC 1-D”. Proceedings of eSim 2004, Vancouver, B.C., June 10-11 pp. 190-195.

Maref, W., Lacasse, M.A., and D. Booth. 2003. An approach to validating computational models for hygrothermal models for hygrothermal analysis – full-scale experiments. Proceedings of the 3rd International Conference on Computational Heat and Mass Transfer, Banff, Alberta, May 26-30, pp. 243-251.

Mujumdar, P. 2005. Computational methods for Heat and Mass Transfer. Taylor and Francis, New York.

Mukhopadhyaya, P. and Kumaran, M.K. 2000. Prediction of moisture response of wood frame walls using IRC's advanced hygrothermal model (hygIRC). Proceedings of the 2nd Annual Conference on Durability and Disaster Mitigation in Wood-Frame Housing, Madison, Wis., Nov. 6-8, 2000, pp. 221-22.

Ojanen, T., Salonvaara, M., Kohonen, R., and Nieminen, J. 1989. Moisture transfer in building structure. Numerical methods. ESPO, Technical Research Centre of Finland. Research Report 595. 113p.

Saber, H.H., Maref, W., Elmahdy, H., Swinton, M.C., and Glazer, R. 2012. 3D heat and air transport model for predicting the thermal resistances of insulated wall assemblies, Journal of Building Performance Simulation, 5:2, 75-91, DOI: 10.1080/19401493.2010.532568

Salonvaara, M. and Karagiosis, A. 1994. Moisture transport in building envelopes using an approximate factorization solution method. CFD Society of Canada, Toronto, Ontario.

Straube, J.F., and Burnett, E.F.P. 2005. Building Science for Buiulding Enclosures. Building Science Press Inc., Westford, MA. 549p.

Siau, J. F. 1995. Wood: influence of moisture on physical properties. Dept. of Wood Sci. and Forest Prod. Virginia Polytechnic Institute and State University, Blacksburg, VA. 227p.

Shih, T. I.-P and W. J. Chyu. 1991. Approximate factored form with source terms. Technical notes. AIAA Journal 29(10):759-760.

Van Reenen, D. 2004. An advanced hygrothermal design tool “hygIRC 1-D”. IRC-ORAL-611.

Viitanen, H. 1997. Factors affecting the development of mould and decay in wooden material and wooden structures, Dissertation, The Swedish University of Agricultural Sciences Dep. of Forest Products, Uppsala.

Annex I
Step-by-step study of the program

For the working version of the program, the main subroutine is runhygIRC which call the subroutine hygIRC. In the run version, there is no main subroutine as only the dll is created. When all the necessary parameters to simulate a case have been completed in GUI, the input file (hygirc.in) and properties files are created. Then hygIRC invokes different subroutines that we are going to describe sequentially.

A1.1 Subroutine DIRECT

It creates filenames of material properties and stores in array of strings PROP(1:8) where:

PROP(1) = ‘aper.dat’

PROP(2) = ‘cond.dat’

PROP(3) = ‘dens.dat’

PROP(4) = ‘difl.dat’

PROP(5) = ‘hcap.dat’

PROP(6) = ‘vper.dat’

PROP(7) = ‘interp.dat’

PROP(8) = ‘psuc.dat’

These names correspond to the filenames generated by the GUI for respective property.

A1.2 Reading the content of material property files

The contents of each individual file, which contain 50 data points for each material, if read and stored in specific arrays as follow:

For isotropic property (density, heat capacity, sorption, and suction):

$$XMAT_i = \begin{bmatrix} u_{1,1} & \cdots & u_{1,50} \\ \vdots & \ddots & \vdots \\ u_{n,1} & \cdots & u_{n,50} \end{bmatrix} \text{ and } YMAT_i = \begin{bmatrix} y_{1,1} & \cdots & y_{1,50} \\ \vdots & \ddots & \vdots \\ y_{n,1} & \cdots & y_{n,50} \end{bmatrix}$$

where i = 3, for density, 5 for heat capacity, 7 for suction, u is the moisture content, y is the property, and n is the material index. For sorption, data are stored in UHMAT and RHMAT for u and RH.

For anisotropic property (air permeability, thermal conductivity, vapour permeability, liquid diffusivity), the data is stored in a 3D-array:

- Air permeability: XMAT1(n, 50, 2) and YMAT1(n, 50, 2) where XMAT1(:, :, 1) and YMAT1(:, :, 1) stores u and k_a in x direction and XMAT1(:, :, 2) and YMAT1(:, :, 2) stores u and k_a in y direction, respectively
- Thermal conductivity: XMAT2(n, 50, 2) and YMAT2(n, 50, 2) where XMAT2(:, :, 1) and YMAT2(:, :, 1) stores u and λ in x direction and XMAT2(:, :, 2) and YMAT2(:, :, 2) stores u and λ in y direction, respectively.
- Liquid diffusivity: XMAT4(n, 2, 50) and YMAT2(n, 2, 50) where XMAT2(:, 1, :) and YMAT2(:, 1, :) stores u and D in x direction and XMAT2(:, 1, :) and YMAT2(:, 1, :) stores u and D in y direction, respectively.
- Vapour permeability: XMAT6(n, 50, 2) and YMAT2(n, 50, 2) where XMAT2(:, :, 1) and YMAT2(:, :, 1) stores u and δ in x direction and XMAT2(:, :, 2) and YMAT2(:, :, 2) stores u and λ in δ direction, respectively.

Related to suction pressure, two other arrays are created via Subroutine INDER to store the first and second derivatives:

- $YMAT7B = \begin{bmatrix} \left(\frac{\partial u}{\partial P}\right)_{1,1} & \cdots & \left(\frac{\partial u}{\partial P}\right)_{1,50} \\ \vdots & \ddots & \vdots \\ \left(\frac{\partial u}{\partial P}\right)_{n,1} & \cdots & \left(\frac{\partial u}{\partial P}\right)_{n,50} \end{bmatrix}$
- $YMAT7C = \begin{bmatrix} \left(\frac{\partial^2 u}{\partial u \partial P}\right)_{1,1} & \cdots & \left(\frac{\partial^2 u}{\partial u \partial P}\right)_{1,50} \\ \vdots & \ddots & \vdots \\ \left(\frac{\partial^2 u}{\partial u \partial P}\right)_{n,1} & \cdots & \left(\frac{\partial^2 u}{\partial u \partial P}\right)_{n,50} \end{bmatrix}$

Other transformations that seem to yield the product of *Dw* and *u* are performed on moisture diffusivity data in Subroutine CALPOT at each data point i:

$$D_{x_i} = D_{x_{i-1}} u_{i-1} + \frac{1}{2}\left(D_{x_{i-1}} - D_{x_i}\right)(u_i - u_{i-1})$$

$$D_{y_i} = D_{y_{i-1}} u_{i-1} + \frac{1}{2}\left(D_{y_{i-1}} - D_{y_i}\right)(u_i - u_{i-1})$$

The new data are stored in DWPOT and their values are used throughout the program.

A1.3 Subroutine INPUT

Read data in the file hygirc.in and stores in various arrays.

A1.3.1 Grid data

The number of nodes in x and y directions are read in N1 and N2 respectively, where N1 = n1 + 2 and N2 = n2 + 2. n1 and n2 are the total number of nodes speficied in the GUI.

The element sizes are read by Subroutine READ1D and stored in DXM and DYM for x and y direction, respectively:

- $DXM = [\Delta x_1 \quad \Delta x_2 \quad \Delta x_3 \quad \cdots \quad \Delta x_{N1-1} \; \Delta x_{N1}]$ where Δx_2 to Δx_{N1-1} are the elements size calculated by the GUI as a function of the number of nodes and thickness of materials and $\Delta x_1 = \Delta x_2$, $\Delta x_{N1} = \Delta x_{N1-1}$.
- $DYM = [\Delta y_1 \quad \Delta y_2 \quad \Delta y_3 \quad \cdots \quad \Delta y_{N2-1} \; \Delta y_{N2}]$ where Δy_2 to Δy_{N2-1} are the elements size calculated by the GUI as a function of the number of nodes and thickness of materials and $\Delta y_1 = \Delta y_2$, $\Delta y_{N1} = \Delta y_{N1-1}$. In 1D, N2 = 3 and the element size is 1.

A1.3.2 Layer parameters

The number of layers are read in NLAYERX and NLAYERY for x and y directions, respectively.

The layer parameters are read in PARAMX and PARAMY:

$PARAMX = \begin{bmatrix} nl_1 & xl_1 & sl_1 \\ \vdots & \vdots & \vdots \\ nl_{1n} & xl_n & sl_n \end{bmatrix}$ where nl_i is the number of nodes for layer i, sl_i is the stretching factor for layer i, xl_i is the expanding factor and n is the number of layers of the case. While PARAMX is initialized as PARAMX(NMAX, 3) in the Common block where NMAX is the maximum number of materials allowed. Its dimension is set to PARAMX(20, 3) in INPUT. PARAMY as the same structure and has only one line for 1D case.

The layer thicknesses are read in THICKX and THICKY:

$THICKX = [tl_1 \quad tl_2 \ldots \quad tl_n]$ where tl_i is the thickness of layer i. THICKY has the same structure and has only one layer of thickness 1 in 1D.

A1.3.3 Simulation parameters

Simulation parameters are read in:

- ISNUMER = 0, for steady state case, 1 for general transient case, and 2 for transient case with weather file.
- DTIME for time step in s
- MAXTIME for duration in s
- DLPRINT for the time step to output the simulation results in s
- YERHOUR for the starting hour when using weather file

A1.3.4 Initial conditions

The moisture potential is read in IMOIST (0 = RH and 1 = u)

The initial condition index is read in IINC (0 = to be read from file, 1 = to be read in input file). In the case 0, the name of the initial condition file is read (initial.con). In the case IINC = 1, the initial RH or u and T are read in the array U(2, N1, N2) where U(1, 2:N1-1, 2:N2-1) stores either RH or u and U(2, 2:N1-1, 2:N2-1) stores T. In 1D where N2 = 3, Figure A1.1 shows how data are stored in U.

A1.3.5 Boundary conditions

The index for boundary condition is read in IBOUND:

$$IBOUND = [i_b \quad i_t \quad i_l \quad i_r]$$

where i_b is the boundary condition index for bottom side, i_t for top, i_l for left (exterior) and i_r for right (interior):

- i = 0 indicates there is an exterior file in which case it reads the weather filename
- i = 1 indicates there is an interior file in which case it read the indoor conditions file name
- i = 2 indicates constant value

From IBOUND, the boundary condition indexes are redefined and stored in IBCP:

$$IBCP = (1{:}NLAYERX, \quad 1{:}4) = \begin{bmatrix} 12 & 22 & 30 & 41 \\ \vdots & \vdots & \vdots & \vdots \\ 12 & 22 & 30 & 41 \end{bmatrix}$$

where IBCP(:, 1) store the bottom values, IBCP(:, 2) stores the top values, IBCP(:, 3) stores the left values and IBCP(:, 4) stores the right values.

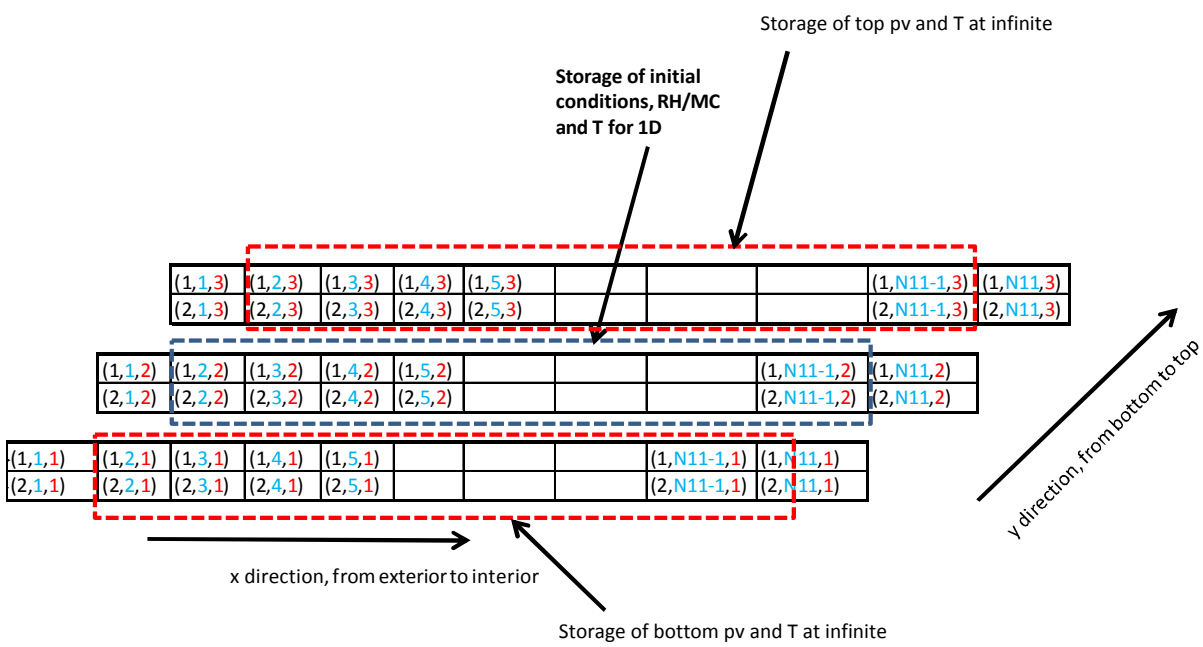

Figure A1.1. Map of array U the case of 1D

A1.3.6 Convective heat transfer coefficients

The Subroutine READ2D reads heat transfer coefficients and store in HTX(2, N2) for bottom and top nodes, and in HTY(2, N1) for left and right nodes. These values are recalculated at each time step using wind velocity when there is a weather file.

$$HTX = \begin{bmatrix} 0 & hc_{l1} & \cdots & \cdots & hc_{ln2} & 0 \\ 0 & hc_{r1} & \cdots & \cdots & hc_{rn2} & 0 \end{bmatrix}$$

$$HTY = \begin{bmatrix} 0 & hc_{b1} & \cdots & \cdots & hc_{bn1} & 0 \\ 0 & hc_{t1} & \cdots & \cdots & hc_{tn1} & 0 \end{bmatrix}$$

where l stands for left, r for right, b for bottom and t for top. In 1D, hc_{l1} is by default 30 and hc_{r1} is by default 8, all other elements of HTX and HTY are 0.

A1.3.7 Convective mass transfer coefficients

The Subroutine READ2D reads mass transfer coefficients and store in BMTX(2, N2) for bottom and top nodes, and in BMTY(2, N1) for left and right nodes. These values are recalculated at each time step using wind velocity when there is a weather file.

$$BMTX = \begin{bmatrix} 0 & hm_{l1} & \cdots & \cdots & hm_{ln2} & 0 \\ 0 & hm_{r1} & \cdots & \cdots & hm_{rn2} & 0 \end{bmatrix}$$

$$BMTY = \begin{bmatrix} 0 & hm_{b1} & \cdots & \cdots & hm_{bn1} & 0 \\ 0 & hm_{t1} & \cdots & \cdots & hm_{tn1} & 0 \end{bmatrix}$$

where l stands for left, r for right, b for bottom and t for top. In 1D, hm_{l1} is by default 2.1 x 10^{-7} and hm_{r1} is by default 5.9 x 10^{-8}, all other elements of BMTX are 0. For BTMY, all elements are set by default to 10^{-40}.

A1.3.8 Absorptivity coefficients

The Subroutine READ2D reads radiation absorption coefficients and store in ABSORX(2, N1) for the left and right nodes and in ABSORY(2, N2) for the bottom and top nodes.

$$ABSORX = \begin{bmatrix} 0 & \alpha_{l1} & \cdots & \cdots & \alpha_{ln2} & 0 \\ 0 & \alpha_{r1} & \cdots & \cdots & \alpha_{rn2} & 0 \end{bmatrix}$$

$$ABSORY = \begin{bmatrix} 0 & \alpha_{b1} & \cdots & \cdots & \alpha_{bn1} & 0 \\ 0 & \alpha_{t1} & \cdots & \cdots & \alpha_{tn1} & 0 \end{bmatrix}$$

where l stands for left, r for right, b for bottom and t for top. In 1D, α_{l1} is by default 0.5 and α_{r1} is by default 0, all other elements of ABSORX and ABSORY are 0.

A1.3.9 Emissivity coefficients

The Subroutine READ2D reads radiation absorption coefficients and store in EMISSX(2, N1) for the left and right nodes and in EMISSY(2, N2) for the bottom and top nodes.

$$EMISSX = \begin{bmatrix} 0 & \varepsilon_{l1} & \cdots & \cdots & \varepsilon_{ln2} & 0 \\ 0 & \varepsilon_{r1} & \cdots & \cdots & \varepsilon_{rn2} & 0 \end{bmatrix}$$

$$EMISSY = \begin{bmatrix} 0 & \varepsilon_{b1} & \cdots & \cdots & \varepsilon_{bn1} & 0 \\ 0 & \varepsilon_{t1} & \cdots & \cdots & \varepsilon_{tn1} & 0 \end{bmatrix}$$

where l stands for left, r for right, b for bottom and t for top. In 1D, ε_{l1} is by default 0.9 and ε_{r1} is by default 0, all other elements of ABSORX and ABSORY are 0.

A1.3.10 Heat flux at boundary (including solar energy)

The index for heat flux at boundary is read in IHFLX (1 = use weather file, 0 = constant).

In the case 0, the Subroutine READ2D reads heat flux at boundary and stores in QFLX(2, N1) for the left and right nodes and in QLFY(2, N2) for the bottom and top nodes.

$$QLFX = \begin{bmatrix} 0 & q_{l1} & \cdots & \cdots & q_{ln2} & 0 \\ 0 & q_{r1} & \cdots & \cdots & q_{rn2} & 0 \end{bmatrix}$$

$$QFLY = \begin{bmatrix} 0 & q_{b1} & \cdots & \cdots & q_{bn1} & 0 \\ 0 & q_{t1} & \cdots & \cdots & q_{tn1} & 0 \end{bmatrix}$$

where l stands for left, r for right, b for bottom and t for top. In hygIRC 1D, all elements of QFLX and QFLY are 0 and there is no mean to set them in GUI.

A1.3.11 Ambient air temperature

The indexes for air temperature (T_a) surrounding the wall are read in ITEMP (1 = use weather file, 0 = constant):

$ITEMP = [i_b \quad i_t \quad i_l \quad i_r]$ where i = 0 for constant and i = 1 for file.

In the case 0, T_a are read from input file and store in U as shown in Figure A1.2.

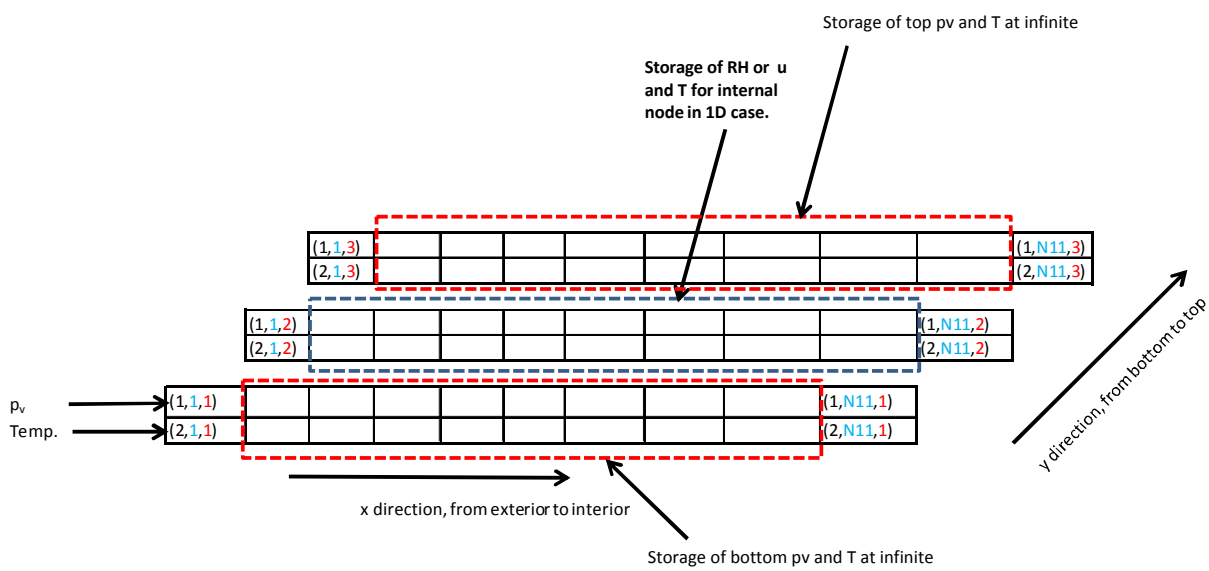

Figure A1.2. Storage boundary air temperature and vapour pressure in 1D case. The vapour pressure is stored in lieu of moisture content for boundary nodes.

T_a is also stored in TEMPX and TEMPY:

$$TEMPX = \begin{bmatrix} 0 & T_{\infty l1} & \cdots & \cdots & T_{\infty ln2} & 0 \\ 0 & T_{\infty r1} & \cdots & \cdots & T_{\infty rn2} & 0 \end{bmatrix}$$

$$TEMPY = \begin{bmatrix} 0 & T_{\infty b1} & \cdots & \cdots & T_{\infty bn1} & 0 \\ 0 & T_{\infty t1} & \cdots & \cdots & T_{\infty tn1} & 0 \end{bmatrix}$$

A1.3.12 Ambient moisture

The index for moisture potential at boundary is read into IMOISP (0 = RH, 1 = u).

The indexes for b, t, l and r are stored in IMOIS:

IF IMOIS = 0, data will be read from input file. If IMOIS = 1, data will be read from file (exterior or interior weather file).

In case IMOISP = 0 and IMOIS = 0, RH at boundary are read and stored in BMOISX and BMOISY:

$$BMOISX = \begin{bmatrix} 0 & RH_{l1} & \cdots & \cdots & RH_{ln2} & 0 \\ 0 & RH_{r1} & \cdots & \cdots & RH_{rn2} & 0 \end{bmatrix}$$

$$BMOISY = \begin{bmatrix} 0 & RH_{b1} & \cdots & \cdots & RH_{bn1} & 0 \\ 0 & RH_{t1} & \cdots & \cdots & RH_{tn1} & 0 \end{bmatrix}$$

From the boundary temperature T, the saturated vapour pressure at the boundary is calculated as (function P1SAT):

$$\begin{cases} p_{vs} = \dfrac{0.0043(120+T)}{60} \; if\ T < -60℃ \\ p_{vs} = (C0 + C1T + C2T^2 + C3T^3)^{2+C} \; if\ T > 60℃\ and\ T < 31.5℃ \\ p_{vs} = B0 + B1T + B2T^2 + B3T^3 + \dfrac{B}{T} \; if\ T > 31.5℃ \end{cases}$$

where C0, C1, C2, C3 are constants equal to 24.430699, 0.92661237, 0.012390661, 6.1925582 10^{-5}, respectively; and C = max(0, T-26.5)*0.2*25.641, and B0 B1, B2, B3 and B are constants equal to -31100.153, 1285.1524, -21.475947, 0.21305921, and 311749.59, respectively.

Then, the partial pressure of water vapour at boundary is computed and store in U as shown in Figure A1.2.

A1.3.13 Wind effects

The indexes for b, t, l and r are stored in IWIND:

$$IWIND = [i_b \quad i_t \quad i_l \quad i_r]$$

where:

- i = 0, no wind

- i = 1, constant wind
- i = 2, read from file

In case constant (i = 1), the values of wind speed (V) and wind direction (WINDIR) are read from input file, then, the heat and mass transfer coefficients are modified as follow, using the factor *f*:

$$\begin{cases} f = 5.82 + 3.96\dfrac{V}{3.6}\ if\ V\ \leq 18\ m/s \\ f = max\left[7.68\left(\dfrac{V}{3.6}\right)^{0.75}, 5\right]\ otherwise \end{cases}$$

<u>Bottom and top:</u>

$h_c = f$

$\beta = 10^{-10} f$

<u>Left:</u>

$h_c = f$

$$\beta = f\left(\frac{2.7 \times 10^{-7}}{30 + 10^{-10}}\right)$$

<u>Right:</u>

$h_c = f$

$$\beta = f\left(\frac{5.9 \times 10^{-8}}{8 + 10^{-10}}\right)$$

HTX, HTY, BMTX and BMTY are updated accordingly.

A1.3.14 Material blocks

The number of material blocks is read into MATBLK. The material indexes are read into IMAT(N1, N2). In 1D, IMAT is structured as follow in the case there are 3 layers, m1 nodes in layers 1, m2 nodes in layer 2 and m3 nodes in layer 3:

$$IMAT = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 1_1 & -1 \\ \vdots & \vdots & \vdots \\ -1 & 1_{m1} & -1 \\ -1 & 2_1 & -1 \\ \vdots & \vdots & \vdots \\ -1 & 2_{m2} & -1 \\ -1 & 3_1 & -1 \\ \vdots & \vdots & \vdots \\ -1 & 3_{m3} & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

The material index is the same as that found in property files.

A1.3.15 Sky temperature

The sky temperature indexes for b, t, l and r are stored in ISKY:

$ISKY = [i_b \quad i_t \quad i_l \quad i_r]$ where:

- i = 0, not used
- i = 1, constant
- i = 2, read from file (i.e. calculated from weather file data)

In case constant, the value of the sky temperature is read, converted to absolute temperature and stored in SKYTEMP and in SKYT(j), j = 1,4. If i = 2, the name of the weather file is read.

A1.3.16 Rain parameters

The rain indexes for b, t, l and r are stored in IRAIN:

$IRAIN = [i_b \quad i_t \quad i_l \quad i_r]$ where:

- i = 0, not used
- i = 1, constant
- i = 2, read from file (i.e. weather file)

In case constant (case left), the value of the rain flux is read into RAINFLX:

$$RAINFLX = [R_b \quad R_t \quad R_l \quad R_r]$$

A1.3.17 Heat and moisture sources

In the GUI, one can choose 0, 1, or 2 for heat and moisture source indexes. 0 means no source, 1 and 2 mean to read sources from file. In these last 2 cases, values of -5005 and -6006 are written in the input file and read into QSRC(2,N1,N2) and use later in the program.

In the case of 1D, moisture sources in the domain are stored in QSRC(1, 2:N1-1, 2) and heat sources are stored in QSRC(2, 2:N1-1, 2).

A1.3.18 Air flow parameters

The index for air flow module, IPCTRL is automatically set to 1(use air flow module) in 1D GUI. There is no option to set it to 0 in GUI.

Stack

The index for stack effect is read in ISTACK (1 = Yes, 0 = no stack).

The distance above neutral pressure level is read into PRESN after set it to negative value.

Indoor ventilation

The index for indoor ventilation is read in IPRESD (1 = Yes, 0 = no ventilation). If IPRESD = 1, the ventilation pressure (Pa) is read into DPRES.

Wind ventilation

The indexes for the wind ventilation are read into IWINDPR = $[i_b \quad i_t \quad i_l \quad i_r]$ where i = 0 if no wind ventilation and i = 1 if there is a wind ventilation. Only the index for the left side is active for 1D.

Air flow resistance boundary

The air flow resistance at boundary is automatically set to 10^{20} for top and bottom and 10^{-20} for left and right. There is no option in the GUI to play with. They are stored in PRES(4,N1).

$$PRES = \begin{bmatrix} 0 & r_{b1} & r_{b2} & \cdots & r_{bn1} & 0 \\ 0 & r_{t1} & r_{t2} & \cdots & r_{tn1} & 0 \\ 0 & r_l & 0 & \cdots & 0 & 0 \\ 0 & r_r & 0 & \cdots & 0 & 0 \end{bmatrix}$$

Air flow boundary type

The indexes for air flow boundary type is stored in IBC(4, N1) as:

$$IBC = \begin{bmatrix} 0 & i_{b1} & i_{b2} & \cdots & i_{bn1} & 0 \\ 0 & i_{t1} & i_{t2} & \cdots & i_{tn1} & 0 \\ 0 & i_{l1} & i_{l2} & \cdots & i_{ln2} & 0 \\ 0 & i_{r1} & i_{r2} & \cdots & i_{rn2} & 0 \end{bmatrix}$$

where i = 1 for pressure type and i = 2 for velocity type. By default they are set to 1 and there is no mean in the GUI to change it. In 1D, there is only one node on the left and one on the right, only i_{l1} and i_{r1} are active in the exterior and interior, respectively.

Pressures at boundary

As the default value of IBC is 1, the pressure at boundary is read and stored in array P(N1, N2):

$$P = \begin{bmatrix} 0 & P_{l,2} & \cdots & P_{l,n2-1} & 0 \\ P_{b,2} & & & & P_{t,2} \\ \vdots & & & & \vdots \\ P_{b,n1-1} & & & & P_{t,n1-1} \\ 0 & P_{r2} & \cdots & P_{r,n2-1} & 0 \end{bmatrix}$$

Default values of pressure at the boundary are all 0s. The internal node pressures are calculated later in the program and stored in P too.

<u>Conversion of RH to u</u>

Before exiting the INPUT subroutine, if the moisture potential index IMOIST was 1 (case initial condition given in RH), RH are converted into u using the sorption curve and all the nodal RH values in U are replaced by u values.

A1.3.19 Subroutine REALDXDY

This Subroutine creates a new vectors DX and DY from DXM and DYM, respectively, that contain the distance between internal nodes :

- $DX = [\Delta x_1 \quad \Delta x_2 \quad \Delta x_3 \quad \cdots \quad \Delta x_{N1-1} \; 0]$ where:

DX(1) = DXM(1)

For i ≥ 2 and i < N1-1, DX(i) = (DXM(i) + DXM(i+1))/2

DX(N1-1) = DXM(N1-1)

DX(N1) = 0

A1.3.20 Merging of HTX and HTY and BMTX and BMTY

The heat transfer coefficients stored in HTX and HTY are modified as follow:

For left side: $hc_{new} = hc_{old} \times 0.5 \times DX(1)$

For right side: $hc_{new} = hc_{old} \times 0.5 \times DX(N1-1)$

For bottom side: $hc_{new} = hc_{old} \times 0.5 \times DY(1)$

For right side: $hc_{new} = hc_{old} \times 0.5 \times DY(N2-1)$

The new values are stored in CNDX and CNDY as:

$$CNDX = \begin{bmatrix} 0 & hc_{l1} & hc_{l2} & \cdots & hc_{ln2} & 0 \\ hc_{b1} & & & & & hc_{t1} \\ hc_{b2} & & & & & hc_{t1} \\ \vdots & & & & & \vdots \\ hc_{bn1} & & & & & hc_{tn1} \\ 0 & hc_{t1} & hc_{t2} & \cdots & hc_{tn2} & 0 \end{bmatrix} = CNDY$$

The mass transfer coefficients BMTX and BMTY are merged into AMUX and AMUX as:

$$AMUX = \begin{bmatrix} 0 & hm_{l1} & hm_{l2} & \cdots & hm_{ln2} & 0 \\ hm_{b1} & & & & & hm_{t1} \\ hm_{b2} & & & & & hm_{t1} \\ \vdots & & & & & \vdots \\ hm_{bn1} & & & & & hm_{tn1} \\ 0 & hm_{t1} & hm_{t2} & \cdots & hm_{tn2} & 0 \end{bmatrix} = AMUY$$

A1.4 Subroutine CORRGRID

Corrgrid recalculates the element size in DXM and DYM in the case where there material i and i+1 have the same layer thickness but with different nodes or have the same number of nodes but different thicknesses or if the last dx in material i differs from the first dx in material i+1.

Corrgrid creates also the vectors LAYERX (0:NLAYERX) and LAYERY(0:NLAYERY) which stores the index of starting number of nodes in each layer in x and y direction, respectively.

A1.4.1 Call to REALDXDY

After modifying DXM, there is another call to REALDXDY to modify DX and DY, the distance between nodes as seen previously.

A1.5 Subroutine LISTPRO

Create file list.pro and for each material, write the material number in list.pro, get the density (function DENS) and write in the file, and starting from RH = 0 to 100% and by increment of 1% when RH < 95% and 0.1% when RH > 95%:

- Get the moisture content u from sorption curve (function SORP)
- Get air permeability (function APER)
- Get heat conductivity (function COND)
- Get heat capacity (function HCAP) and compute volumetric heat capacity : $\rho c = \rho_u[c_o + 4200u]$
- Get vapour permeability δ_p (function VPER)
- Get liquid diffusivity D_L as follows :

  Get D_w (function DIFL) from table

  Compute vapour diffusivity with u as potential:

$$D_V = \frac{\delta_p}{\rho} \times \frac{\partial P_V}{\partial u}$$

  Compute liquid diffusivity D_L

$$D_L = D_w - D_V$$

The minimum of value D_L is set to 10^{-16} and it is corrected by two factors as:

$$D_L = D_L \times f1 \times f2$$

where:

$$f1 = \frac{1.004}{C0 + C1T + C2T^2 + C3T^3 + C4T^4}$$

$$\begin{cases} f2 = 0.01 \; if \; T < 0 \\ f2 = 1 \; if \; T > 0 \end{cases}$$

with C0 = 1.76632, C1 = -0.0538082, C2 = 0.000942267D0, C4 = -8.52254 10^{-6}, and C4 = 3.01283 10^{-8}. At 20°C, f1 = 1.00029. It seems to be the ratio of water viscosity at reference temperature and water viscosity at temperature T.

- Get the liquid water diffusivity assuming suction as the potential for moisture flow:

$$k_w = -\rho_o D_L \frac{\partial u}{\partial P_c}$$

- Write the results in list.pro

A1.6 Subroutine XYCALC

Creates vectors X(N1) and Y(N2) where it stores the right boundary position of each cell, from index 2 to N1-1.

A1.7 Subroutine PLOTBND

Creates the file "geom.dat" where it writes the geometry of each layer of the structure and of each element.

A1.8 Subroutine ISTOCHAS

This routine either reads stochastic properties (air permeability, heat conductivity, moisture diffusivity and vapour permeability) from file stoch.dat if it exists or creates array SF(4, N1, N2) that contains random values that are used later in the program to randomized the above mentioned properties (in subroutine Materp). When the deterministic properties are going to be used, all elements of array SF are set to 1.

A1.9 Reading initial conditions from file

When it was set to read initial conditions from file (IINC = 0), the file "initial.con" is read here. Arrays U and RH are updated and U is assigned to UOLD (solution at previous time step).

A1.10 Subroutine INCON

Get dry density (ρ_o) and specific heat capacity (c_o) for each material and stores in vector RHO(1:n_{mat}) and CP(1:n_{mat}).

A1.11 Set pressures around neutral plan level

When it was selected to use air module, the stack pressure and/or the ventilation pressure are used to set pressure around the neutral plan for external and internal boundary nodes:

For outdoor: $P_e = -\left[(H - H_n) \times \frac{gP_0M_a}{R(T_{out}+273)}\right]$

For indoor: $P_i = P_{vent} - \left[(H - H_n) \times \frac{gP_0M_a}{R(T_{in}+273)}\right]$

With:

H =height at which the pressure is calculated (m)

H_n = height at neutral plan (m), set to 3 m by default.

g = gravity (ms^{-2})

T_{out} = outdoor temperature

T_{in} = indoor temperature

P_{vent} = ventilation pressure (Pa)

Pi = indoor pressure (Pa)

P_0 = barometric pressure (101300 Pa)

M_a = molecular mass of air (29 kg $mole^{-1}$)

R = universal gas constant (8314.33 kg $K^{-1}mole^{-1}$

The nodal values are stored in array P as:

$$P = \begin{bmatrix} P_{e,1} & \cdots & P_{e,N2} \\ \vdots & \vdots & \vdots \\ P_{i,1} & \cdots & P_{i,N2} \end{bmatrix}$$

A1.12 Compute and store relative humidity

T at nodal points and at boundary are retrieved from U and assigns to TEM(N1,N2)

RH at nodal points and at boundary are computed and store in RH(N1,N2). For RH at nodal points within the domain, p_v is first derived from the sorption curve using the function UP and RH is then calculated by dividing the p_v by $P_{VS}(T)$. For boundary RH:

$$RH = \frac{p_v}{P_{vs(T)}}$$

where $p_v = \frac{u_{air \times P}}{0.622 + u_{air}}$

Note: No T effect is taken into account when interpolating RH from sorption curve. However, the current T is used to calculate the P_{VS}.

A1.13 Subroutine MASS

Compute and store the mass of water (volume weighted) in each layer in the array TMAS, the total mass of water in the wall in the variable TOTMAS, and the moisture content in each layer in the array TOTU:

- For mass of water in each cell i

$$m_i = A_i u_i \rho_{oi}$$

where A_i is the area of the cell:

$$A_i = \left[\frac{\Delta x_{i-1}}{2} + \frac{\Delta x_i}{2}\right]\left[\frac{\Delta y_{i-1}}{2} + \frac{\Delta y_i}{2}\right]$$

Δx_{i-1} and Δx_i are the distance between nodes i-1 and i, and node i and i+1, respectively. They are obtained from the vector DX. If material i is not equal to material i+1, Δx_i is replaced by $\Delta x_{c,i}$ where $\Delta x_{c,i}$ is the cell size obtained from the vector DXM. Assuming a unit depth of the wall, the unit of the m_i is kg.

- For the mass of water in each layer

$$TMAS(l) = \sum_{1}^{n_c} m_i$$

where n_c is the total number of cells in material i.

- For the mass of water in the whole wall:

$$\text{TOTMAS} = \sum_{1}^{n_l} TMAS(l)$$

where n_l is the total number layers.

- For the moisture content of each layer:

$$TOTU(l) = \frac{TMAS(l)}{Vol_l}$$

The unit of moisture content, assuming a unit depth for the wall, is kg.m^{-3}. It is displayed in the GUI in kg m^{-2}.

A1.14 Subroutine MASST

Compute and store the T (volume weighted) of each layer in the array TTEM, the total T in the wall in the array TOTEMP, and the T in each layer in the array TETOTU:

- For each cell i

$$\bar{T}_i = A_i T_i$$

where A_i is the area of the cell:

$$A_i = \left[\frac{\Delta x_{i-1}}{2} + \frac{\Delta x_i}{2}\right]\left[\frac{\Delta y_{i-1}}{2} + \frac{\Delta y_i}{2}\right]$$

Δx_{i-1} and Δx_i are the distance between nodes i-1 and i, and node i and i+1, respectively. They are obtained from the vector DX. If material i is not equal to material i+1, Δx_i is replaced by $\Delta x_{c,i}$ where $\Delta x_{c,i}$ is the cell size obtained from the vector DXM.

- For each layer

$$TTEM(l) = \sum_{1}^{n_l} \bar{T}_i$$

  where n_l is the total number of cells in material i.

- For the T in the whole wall:

$$\text{TOTEMP} = \sum_{1}^{n_l} TTEM(l)$$

  where n_l is the total number layers.

- For the T of each material:

$$TETOTU(l) = \frac{TTEM(l)}{Vol_l}$$

A1.15 Subroutine INSTU

Compute mass of water in each material and in the whole structure and write in tec files and in INSTU.OUT at each time step. The algorithm used is the same as the one described in subroutine MASS.

A1.16 Subroutine INSTT

Compute air mass and heat fluxes at the boundary of the wall and write in tec files and in INSTT.OUT at each time step. The equations used are almost the same for all faces. We will describe the equation used for the bottom side of the wall.

Mass flux of air

$$Q_{ma} = \sum_{i=2}^{n1-1} q_{ma,i}$$

with:

$$q_{ma,i} = \rho_a \, V_{a,i} \, A_i$$

where $q_{ma,i}$ is the mass flow rate of air at the bottom of cell i, $Q_{m,a}$ is the mass flow rate of air at the bottom face of the wall (kg/s), n1 is the number of nodes (or cells) in x direction, V_a is the air velocity (m/s) obtained from the array VELO, and A_i is the bottom area of cell i ($A = 0.5(\Delta x_{i-1} + \Delta x_i)$). The same formula is used for top, left and right side of the wall.

Latent heat flow rate

$$Q_{hl} = \sum_{i=2}^{n1-1} q_{hl,i}$$

with:

$$q_{hl,i} = q_{v,i}\, L_{v,i}\, A_i$$

where $q_{hl,i}$ is the latent heat flow rate at the bottom of cell i (W), $Q_{h,l}$is the latent heat flow rate at the bottom of the wall (W), $q_{v,i}$ is the vapour flux (kg/m^{2}s) at the bottom of cell i, obtained from the array QMOVY and L_v is the latent heat of evaporation (J/kg). The same equations are used for the left side of the wall. For the latent heat flow rate at the top and right sides of the wall, the vapour flux is obtained from the array XV.

Sensible air heat flow rate

$$Q_{ha} = \sum_{i=2}^{n1-1} q_{ha,i}$$

With:

$$q_{ha,i} = \min\left(\rho_a V_{a,i} A_i, 0\right) T_s\, c_a + \max(\rho_a V_{a,i} A_i, 0) T_a\, c_a$$

Where $q_{ha,i}$is air heat flow rate at the bottom of the cell i (W), Q_{ha} is the total sensible heat flow rate at the bottom of the wall (W), T_S is the wall surface temperature, T_a is the air temperature (bottom side), c_a is the specific heat capacity of air (J/kgK).

Dry heat flow rate

$$Q_{hd} = \sum_{i=2}^{n1-1} \left(q_{h,i} - q_{hl,i} - q_{ha,i}\right)$$

where Q_{hd} is the total dry heat at the bottom of the wall (W) and q_h is the total heat flow rate at the bottom of the cell i, obtained from the array QTY .

A1.17 LOOP from 1 to max time

For each time step the following instructions are run.

A1.17.1 Update solutions

- Assigns UOLD to U000
- Assigns U to UITER
- Assigns U to UOLD
- Calls function UP to compute vapour pressure for internal nodes using sorption curve and stores in array PVN(N1, N2)

- Computes relative humidity for internal and boundary nodes and stores in array RHU(N1,N2)

A1.17.2 Subroutine MATERP

- Compute latent heat of water at each node and store in the array HDX(N1,N2)

$$L = (2.501 \times 10^6 - 2300T) + 333 \times 10^3 f$$

Where f = 1 if T < -3°C, f = 0 if T > 0°C, and f varies from 0 to 1 when T is between 0 and -3°C.

Then the harmonic mean is calculated before storage in HDX as:

$$\bar{L}_i = \frac{2L_i L_{i+1}}{L_i + L_{i+1} + 10^{-20}}$$

In the first call to MATERP, the variable FDH is set equal to PCFAC which is zero. As consequence, HDX (i,j) = 0. But in a subsequent call of MATERP by the subroutine DUSTAR, PCFAC is set to 1.

- Compute liquid diffusivity D_L (assuming u as potential) for internal nodes and store in AMUX(N1, N2) and AMUY(N1,N2):
- Update volumetric heat capacity for internal nodes and store in RCP

$$\rho c = \rho_0 [c_o + 4200u]$$

where u is the moisture content obtained at previous time step.

- Update thermal conductivity at x and y directions for internal nodes, compute harmonic mean and store in CNDX and CNDY:

$$\bar{\lambda}_i = \frac{2\lambda_i \lambda_{i+1}}{\lambda_i + \lambda_{i+1} + 10^{-20}}$$

- Update water vapour permeability and store in VAPV(2, N1,N2) where VAPV(1, :, :) stores values in x direction and VAPV(2, :, :) stores the values in y direction.

There is a correction made on water vapour permeability in function of the temperature in the function VPER:

$$\delta_p = \delta_{po} + a(\delta_{pu} - \delta_{po})$$

where:

δ_{po} = vapour permeability at dry state

δ_{pu} = vapour permeability at moisture content u as provided in the table

a = coefficient (a = 1 if T ≥ 0°C; a = 0 if T < -5°C; a varies from 0 to 1 when T is between -5 and 0°C)

A1.17.3 Set boundary pressures

If there is a specified wind pressure difference, ΔP, between the exterior and interior, the pressure at the exterior wall is:

$$Pe = \frac{1}{2} \times \Delta P$$

and the pressure at the interior wall is:

$$Pi = -\frac{1}{2} \times \Delta P$$

If there is no specified pressure difference, the exterior and interior pressures are calculated following many steps. First, the wind pressure, P_w (Pa) is calculated by the function WINDPRES using weather data:

$$P_w = 0.06 \times Coef \times (10^{-10} + V_z)^2$$

in which:

$$V_z = \left(\frac{V_r}{3.6}\right) \times \left(\frac{z}{z_r}\right)^{0.22}$$

V_z = wind speed at height z (**set to 1.8 m**) in the program

V_r = wind speed at reference height, set to 10 m in the program

$$Coef = \frac{A + C\,\Phi_w + E\,{\Phi_w}^2}{1 + B\,\Phi_w + D\,{\Phi_w}^2}$$

Φ_w = angle of impediment of wind on the wall

$= |\Phi - W_{or}|$ where Φ is the wind direction

W_{or} = wall orientation. If $\Phi_w > 180$, $\Phi_w = 360 - \Phi_w$

A = 0.5914

B = -0.01463

C = -0.01093

D = 0.000105

E = 2.367 x 10^{-5}

Then, the pressures on the external and internal faces of the wall are calculated as:

For outdoor:

$$P_e = P_w - \left[(H - H_n) \times \frac{g P_0 M_a}{R(Te + 273)}\right]$$

For indoor:

$$P_i = P_{vent} - \left[(H - H_n) \times \frac{gP_0M_a}{R(Ti + 273)}\right]$$

with:

H =height at which the pressure is calculated (m)

H_n = height at neutral plan (m), set to 3 m by default.

g = gravity (ms^{-2})

Te = outdoor temperature

Ti = indoor temperature

P_w = wind pressure (Pa)

P_{vent} = ventilation pressure (Pa)

Pi = indoor pressure (Pa)

P_0 = barometric pressure (101300 Pa)

M_a = molecular mass of air (29 kg $mole^{-1}$)

R = universal gas constant (8314.33 kg $K^{-1}mole^{-1}$

The second term on the right hand side of the equation is the stack pressure.

The nodal values are stored in array P as:

$$P = \begin{bmatrix} P_{e,1} & \cdots & P_{e,N2} \\ \vdots & \vdots & \vdots \\ P_{i,1} & \cdots & P_{i,N2} \end{bmatrix}$$

A1.17.3 Outdoor and indoor conditions: subroutine WEATH

- Read content of weather file at each time step if it was chosen to use weather file. At minimum, the weather file contains the hour, the temperature, the relative humidity, the wind speed, the wind direction, the total radiation, the diffuse radiation, the reflected radiation, the total rain and the cloudiness index. Optionally, it can contain the specified wind driven rain and/or the specified effective temperature.
- If applicable (i.e. if there is no interior data file), compute indoor temperature and relative humidity using outdoor data
- Modify heat and mass transfer coefficients using wind velocity
- Compute sky temperature
- If interior data file exists, read interior temperature and relative humidity
- Update arrays TEMPX, U, UOLD, UITER with new data.

Outdoor conditions

Convection transfer coefficients

They are either read in input file if it was chosen to use constant values (set in the GUI) or calculated using the wind velocity obtained at each time step from exterior weather file.

When it was chosen to use weather file and to take into account the wind effect on external transfer coefficient (i.e. IWIND(3) = 2), the external convection transfer coefficients are calculated as follow:

$$h_c = f$$

$$\beta_e = f\left(\frac{2.7 \times 10^{-7}}{30 + 10^{-10}}\right)$$

$$\begin{cases} f = 5.82 + \dfrac{3.96\,V}{3.6} \; if\; V \le 18\; km/h \\ f = \max\left[7.68\left(\dfrac{V}{3.6}\right)^{0.75}, 5\right] \; otherwise \end{cases}$$

The values 2.7 x 10-7 and 30 are the default values of β_e and h_c read in put file. The new values of β_e and h_c are then assigned to each external boundary nodes and stored in BMTX and HTX, respectively.

The value of the outside air temperature (T) and relative humidity (RH) read in the weather file are assigned to each exterior boundary node and stored in the matrix TEMPX and RHU, respectively. The outdoor air temperature is also stored in U, UOLD or UITER. *T* and *RH* are used to compute vapour pressure in the air as:

$$p_v = p_{vs}(T) \times \frac{RH}{100}$$

in which p_{vs} is the saturated vapour pressure calculated by the function P1SAT as:

$$\begin{cases} p_{vs}(T) = \dfrac{0.0043(120 + T)}{60} \; if\; T < -60°\text{C} \\ p_{vs}(T) = (C_0 + C_1 T + C_2 T^2 + C_3 T^3)^2 + C \;\; if\; T \ge -60°\text{C}\; and\; \le 31.5°\text{C} \\ p_{vs}(T) = B_0 + B_1 T + B_2 T^2 + B_3 T^3 + \dfrac{B_4}{T} \; otherwise \end{cases}$$

in which C0 = 24.430699; C1 = 0.92661237; C2 = 0.012390661; C3 = 6.1925582.10^{-5}; C = max(0, T-26.5)*0.2D0*25.641; and B0 = -31100.153; B1 = 1285.1524; B2 = -21.475947; B3 = 0.21305921; and B4 = 311749.59.

Once calculated, the outdoor partial vapour pressure is assigned to exterior boundary nodes and stored in U, UOLD, and UITER.

Sky temperature

The model used to compute sky temperature is:

$$T_{sky} = \left(F_1 {T_a}^4 + F_2 A^4\right)^{0.25}$$

In which:

$$F_1 = 1 - F_2$$

$$F_2 = \left[cos\left(\frac{0.5\alpha\pi}{180}\right)\right]\left[cos\left(\frac{0.5\alpha\pi}{180}\right)\right]^2$$

$$A = (1 - 0.125c_I) \times \left(0.0552{T_a}^{1.5}\right) + 0.125c_I T_a$$

T_a = air temperature (K)

c_I = cloudiness index

α = wall inclination (°)

T_{sky} is stored in the variable SKYTEMP. It is also assigned to each external node and saved in the vector SKYT.

Interior conditions

The interior conditions are either read in input file (values set in the GUI), read from interior weather file, or calculated from exterior weather file.

Case interior weather file exists

The interior weather file, if it exists, contains minimally for each hour, the indoor air temperature (T_{in}) and relative humidity (RH_{in}). It can also contain the specified pressure difference across the wall in which case it is also read and assigned to the variable *specified_pressure_difference*.

Case there is no interior weather file

If there is no indoor file (rarely the case in 1D since the interior weather file is generated automatically from various models or imported), the indoor T and RH are calculated using indoor average daily conditions read from the file tindoor.bcf and the exterior weather conditions:

$$T_{in} = \max(T_{in\_avg}, T_{out} + 3)$$

$$p_{v,in} = \frac{u_{in} P_o}{0.622 + u_{in}}$$

$$RH_{in} = \frac{p_{v,in}}{p_{vs}(T_{in})} \times 100$$

in which:

$$u_{in} = u_{out} + u_{in,avg}$$

$$u_{out} = \frac{0.622 p_{v,out}}{P_o - p_{v,out}}$$

where:

$T_{in,avg}$ is the average indoor air temperature read from the file tindoor.bcf; $u_{in,avg}$ is the average moisture content of the air for the hour of the day, obtained from the vector XADD where the content of the indoor air was read for every hour of the day; T_{out} is the outside air temperature, u_{out} is the outside moisture content of the air; Po is equal to 101325 Pa. RH_{in} should be between RHLIM1 and RHLIM2, the maximum and minimum interior RH read from the file tindoor.bcf.

Storage of indoor conditions

Once read or calculated, T_{in} is assigned to internal boundary nodes stored in U, UOLD, UITER, and in TEMPX, $p_{v,in}$ is also assigned to internal boundary nodes and stored in U, UOLD, and UITER.

A1.17.4 Compute air permeability at the cell faces: subroutine GENAKV

The subroutine GENAKV compute air permeability at the cell faces as:

$$\bar{k}_x = \frac{\Delta x_{c,i} + \Delta x_{c,i+1}}{\frac{\Delta x_{c,i}}{k_{x,i}} + \frac{\Delta x_{c,i+1}}{k_{x,i+1}}}$$

$$\bar{k}_y = \frac{\Delta y_{c,i} + \Delta y_{c,i+1}}{\frac{\Delta y_{c,i}}{k_{y,i}} + \frac{\Delta y_{c,i+1}}{k_{y,i+1}}}$$

In which $\bar{k}_x$ is the air permeability in x direction at the interface of cell i and i+1; $\Delta x_{c,i}$ and $\Delta x_{c,i+1}$ are the cell sizes of i and i+1 in x direction, obtained from the vector DXM; $k_{x,i}$ and $k_{x,i+1}$ are the air permeability of the cells i and i+1 in x directions, computed by the function APER. The same definitions apply for the y direction. In the case of a uniform grid size distribution, $\bar{k}_x$ turns to be the harmonic mean of the permeability between nodes i and i+1.

At the air/solid or solid/air interface where the node i or i+1 is at the boundary, k_x is computed as:

$$k_x = \frac{0.5 \Delta x_i}{F_r + 10^{-20}}$$

Where Fr is the boundary flow resistance from the array PRES (set to 10^{-20} in the input file) and Δx_i is the distance between node i and i+1.

Before storage, the value of the interface k is divided by **1.7 x 10^{-5}** which seems to be the air dynamic viscosity to convert from intrinsic permeability to air permeability ($k_a = k/\mu$).

A1.17.5 Subroutine DARCY : solve pressure equation

- Solve the pressure equation at each time step.

The discretization of the pressure equation using finite volume approach leads to a system of algebraic equations (using (i,j) notation):

$$a_{i,j}P_{i,j} = a_{i-1,j}P_{i-1,j} + a_{i+1,j}P_{i+1,j} + a_{i,j-1}P_{i,j-1} + a_{i,j+1}P_{i,j+1} \quad \text{(A.1)}$$

By introducing the boundary conditions, the final system to solve is:

$$[A]\{x\} = \{c\}$$

Where $\{x\}$ is the vector of unknown pressures, $\{c\}$ is the vector of known pressures and $[A]$ is the array of coefficients: $a_W = \frac{k_{x,w}}{\Delta x}\Delta y$, $a_E = \frac{k_{x,e}}{\Delta x}\Delta y$, $a_N = \frac{k_{y,n}}{\Delta y}\Delta x$, $a_S = \frac{k_{y,s}}{\Delta y}\Delta x$, and $a_P = a_W + a_E + a_S + a_N$

The first step to solve the pressure equation is to call the subroutine GENAKV which:

- calls the subroutine APER to get the intrinsic permeability (k) at each nodal points,
- computes the harmonic mean at the cell interfaces
- computes the air permeability ($k_a = k/\mu_a$) where μ_a (= 1.7 10^{-5} Pa.s) is the dynamic viscosity of the air

The harmonic mean values are stored in array AKV(1,:,:) for x direction and AKV(2,:,:) for y direction.

The second step is then the computation the coefficients a_W, a_E, a_S, a_N, and a_P which are stored in the array AP, and the RHS which stores the constants or known values of P.

In assembling the system of equations, instead of introducing the boundary prescribed pressure into Eq. A. 1 , a similar equation is considered:

$$a_P P = P_{Prescribed}$$

where $a_P = 1$. This leads to a system on N1 x N2 equations where N1 is the number of nodes in x direction and N2 the number of nodes in y direction. For 1D case, N2 is equal to 3. The subroutine DARCY calculates and store all the coefficients in the matrix APP(5,N1,N2) as:

$$APP(1,i,j) = \frac{a_W}{a_P}$$

$$APP(2,i,j) = \frac{a_E}{a_P}$$

$$APP(3,i,j) = \frac{a_S}{a_P}$$

$$APP(4,i,j) = \frac{a_N}{a_P}$$

$$APP(5,i,j) = 1$$

It also creates the vector of constants or known values RHSP (N1xN2, 1), where $RHSP(i,j) = \frac{0}{a_{i,j}}$ or $RHSP(i,j) = \frac{Pn}{a_{i,j}}$ with P_n the known pressure for the boundary node i,j.

The subroutine DARCY then calls the subroutine SOLVPRES which uses a LU decomposition and back substitution to solves the system of equations. In fact, SOLVPRES creates 4 vectors:

- AA(:) which stores the known non-zero coefficients of APP;
- ULL(:) which stores the elements of RHSP;
- IRNH(:) which stores the row indices of APP;
- ICNH(:) which stores the column indices of APP.

These 4 vectors are transmitted to subroutine MA28AD which performs the LU decomposition. The factors obtained as well as the vector ULL are then used by the subroutine MA28CD for the final resolution. The results are stored in the array P(N1, N2).

A1.17.6 Radiation and effective outdoor temperature

Total irradiation: Subroutine SOLAR

SOLAR computes the total radiative heat flux, Q_r (W/m^2), received on a wall surface. The results is stores in the variable QWALL:

$$Q_r = I_d + I_{dif} + I_r$$

where:

I_d = direct sun beam

I_{dif} = diffuse component

I_r = reflected component

The reflected component, I_r is computed as:

$$I_r = RI \times \left[\frac{1 - \cos\left(\frac{\beta\pi}{180}\right)}{2}\right]$$

where RI is the reflected irradiance from weather data and β is the inclination angle of the wall.

The diffuse component, I_{dif}, is computed as:

$$I_{dif} = DHI \times \left[\frac{1 + \cos\left(\frac{\beta\pi}{180}\right)}{2}\right]$$

where DHI is the diffuse irradiance falling on horizontal surface, from weather data, and β is the inclination angle of the wall.

The direct beam, I_d, received by a wall is computed as:

$$I_d = \frac{DNI}{\sin \alpha} \times \cos \chi$$

where DNI is the direct normal irradiance from weather data. α, the sun elevation angle and χ are computed by the subroutine AFSUN using the information on wall orientation and inclination, latitude, longitude, time zone, hour of the day, and hour of the year as follows:

- Elevation angle, α

$$\sin \alpha = \sin \delta \sin\left(\frac{\pi}{180}\varphi\right) + \cos \delta \cos\left(\frac{\pi}{180}\varphi\right) \cos|\omega|$$

Where δ = earth declination angle, ω = hour angle, and φ = latitude. The earth declination angle is :

$$\delta = 23.45 \frac{\pi}{180} sin\left(\frac{\pi(280.1 + 0.9863d)}{180}\right)$$

where d is the day of the year (1 to 365) calculated the ratio of the cumulative hour of simulation in a given year (variable HYEAR) and 24. The hour angle is:

$$\omega = 15\left[(12 - t_d) - E_t - \frac{(t_{zo} - l)}{15}\right] \times \frac{\pi}{180}$$

Where t_d is the time of the day (1 to 24), E_t is the equation of time, t_{zo} is the time zone and *l* is the longitude. The equation of time is expressed as:

$$E_t = 0.1645\, sin(2B) - 0.1255 \cos B - 0.0255 \sin B$$

with $B = \frac{\pi}{180} \times (0.989d - 80.11)$

- Parameter χ

$$\cos \chi = \cos \alpha\, sin\left(\frac{\pi}{180}\beta\right) \cos(F) + \sin \alpha\, cos\left(\frac{\pi}{180}\beta\right)$$

where β is the wall inclination angle and $F = S_{Az} - \frac{\pi}{180} W_{Az}$. W_{AZ} is the wall azimuth angle and S_{AZ} is the sun azimuth angle given by:

$$\sin S_{AZ} = \frac{\cos \delta \sin|\omega|}{\cos \alpha}$$

There are some limitations on S_{AZ} based on the parameters P1 and P2 expressed as:

$P_1 = 24 \frac{12-\omega}{2\pi}$ and $P_2 = \frac{\tan \delta}{\tan\left(\frac{\pi}{3}\right)}$

- If (cos ω < p2 and P1 > 12) then $S_{AZ} = 2\pi - S_{AZ}$
- If (cos ω = p2 and P1 < 12) then $S_{AZ} = \pi/2$
- If (cos ω = p2 and P1 > 12) then $S_{AZ} = 1.5\pi$
- If (cos ω > p2 and P1 < 12) then $S_{AZ} = \pi - S_{AZ}$
- If (cos ω > p2 and P1 > 12) then $S_{AZ} = \pi + S_{AZ}$

Solar heat flux is stored in arrays QFLX(2, N2) and QFLY(2, N1).

Equivalent outdoor temperature

The sol-air temperature is computed as:

$$T_e = T_a + \frac{\alpha I_T}{h_o + 10^{-20}} \tag{A.2}$$

where:

Te: effective outdoor or sol-air temperature (°C)

T_a: ambient air temperature (°C)

α: wall surface absorptivity

$\Delta x_{1/2}$ = half nodal distance (m)

I_T: total radiation heat flux received (W/m^2)

h_o = equivalent surface heat transfer coefficient (convection and radiation) (W/m^{2}K)

Once computed, the sol-air temperature is stored in U, UITER, and UOLD in lieu of ambient air temperature. The outdoor relative humidity, which is stored in RHU and was initially read in the weather file (in the subroutine WEATH), are updated as:

$$RH = RH(T_a) \times \frac{p_{vs}(T_a)}{p_{vs}(T_e)} \tag{A.3}$$

where p_{vs} is the saturated vapour pressure.

The equivalent heat transfer coefficient, h_o, is computed as:

$$h_o = (h_c + \varepsilon\sigma\bar{T}^3) \tag{A.4}$$

where h_c is the convective heat transfer coefficient, ε the emissivity of the surface and σ the Stefan-Boltzmann's constant. $\bar{T}$ seems to be some sort of average between ambient air temperature and the surface temperature, calculated as follows:

$$\bar{T} = \frac{1}{2} \times \left(\frac{h_c T_a + \frac{\lambda}{\Delta x_{1/2}} T + L_v (Q_{V1} - Q_{V2})}{h_c + \frac{\lambda}{\Delta x_{1/2}}} + T_a \right)$$

λ = thermal conductivity

T = temperature of the internal node near the surface

L_v = latent heat

Q_{V1} = vapour flux arriving to the surface obtained from the array QMOVX

Q_{V2} = vapour flux leaving the surface and entering the wall, obtained from the array XV

Similar equations are used for indoor, top and bottom faces of the wall. Then, the new effective outdoor temperature is assigned to corresponding positions in U, UOLD and UITER, and the new equivalent heat transfer coefficient is stored in CNDX.

A1.17.7 Wind Driven Rain: Function WCORR

The function WCORR computes the rain deposition rate on a wall as function of the wind speed, wind direction, normal rain, wall orientation and wall inclination:

$$WDR = RAIN_n \times \cos\theta \times V_z \times DRF \times RDF$$

where:

WDR: wind driven rain (mm/h)

$RAIN_m$: average rainfall rate on a horizontal surface (mm/h)

θ: angle between the normal to the wall (wall orientation) and the wind direction

= ψ - β_W, where ψ is the wall orientation and β_W is the wind direction

V_z: wind speed (m/s) at height considered Z (m)

$V_z = \left(\frac{V}{3.6}\right)\left(\frac{Z}{10}\right)^{0.22}$, V being the wind speed (m/s) from weather data

<u>Note</u>: a value of z = 1.8 m is actually used in the program, which corresponds to critical height of wind driven rain for low-rise residential building. For other building types (mid-rise, high-rise) other z should be considered.

DRF: driven rain factor. It is a function of the terminal velocity of raindrop in as still air, *Vt* (m/s), and the rain drop size , *DS*(mm):

$V_t = -0.166033 + 4.91844 \times DS - 0.888016 \times DS^2 + 0.054888 \times DS^3$ and $DRF = \frac{1}{V_t}$

$DS = \left(1.3 \times RAIN_m{}^{0.232}\right)\left(1 - \frac{1}{XN}\right)^{\frac{1}{XN}}$, where $XN = 2.25$

RDF = Rain deposition factor, set to 0.4 in the program. This value corresponds to the middle of low-rise building. For high-rise and corners of building, the value change (i.e. 0.9).

Note: the value of WDR is set equal to 0 if T at boundary < 0 or $\beta_e < 0.5*7.4\ 10^{-9}*hc$.

The value of WDR is assigned into array RAINX(2, N2) if left or right side are concerned, or into RAINY(2, N1) if bottom or top side is concerned: RAINX(1, 2:N2-1), and RAINY(1, 2:N1-1) stores values for left and bottom sides, respectively and RAINX(2, 2:N2-1) and RAINY(2, 2:N1-1) stores values for right and top sides, respectively.

A1.17.8 Moisture and heat equation resolution: subroutine DUSTAR

- The subroutine DUSTAR performs the discretization and the iterative resolution of moisture and heat equations using delta-form of the approximate factorization.

Moisture equation

The moisture transfer equation can be rewritten as:

$$\rho_o \frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = Q_m \tag{A.5}$$

where F and G are the moisture fluxes in x and y directions, respectively:

$$F = -\rho_o D_{w,x} \frac{\partial u}{\partial x} + k_{w,x}\rho_w g - \delta_{P,x}\frac{\partial p_v}{\partial x} + \rho_v V_x$$

$$G = -\rho_o D_{w,y} \frac{\partial u}{\partial y} + k_{w,y}\rho_w g - \delta_{P,y}\frac{\partial p_v}{\partial y} + \rho_v V_y$$

Time integration and factorization following the steps described above give:

$$\left[\left(I + \frac{\theta \Delta t}{\rho_o}\frac{\partial}{\partial x}A_x^n\right)\left(I + \frac{\theta \Delta t}{\rho_o}\frac{\partial}{\partial y}A_y^n\right)\right]\Delta u^{n+1} = \frac{\Delta t}{\rho_o}\left[-\frac{\partial \mathrm{F}}{\partial x} - \frac{\partial \mathrm{G}}{\partial y} + Q_m\right]^n \tag{A.6}$$

where:

$$A_x^n = \frac{\partial}{\partial u}\left(-\rho_o D_{w,x}\frac{\partial u}{\partial x} + k_{w,x}\rho_w g - \delta_{P,x}\frac{\partial p_v}{\partial x} + \rho_v V_x\right)$$

$$A_y^n = \frac{\partial}{\partial u}\left(-\rho_o D_{w,y}\frac{\partial u}{\partial x} + k_{w,y}\rho_w g - \delta_{P,y}\frac{\partial p_v}{\partial x} + \rho_v V_y\right) \tag{A.7}$$

Discretization in the first direction (x sweep):

$$\left(I + \frac{\theta \Delta t}{\rho_o}\frac{\partial}{\partial x}A_x^n\right)z = \frac{\Delta t}{\rho_o}\left[-\frac{\partial \mathrm{F}^n}{\partial x} - \frac{\partial \mathrm{F}^n}{\partial y} + Q_m^n\right] \tag{A.8}$$

Integrating A.8 over the control volume (Figure 2), we get:

$$\int_s^n \int_w^e z\,dxdy + \frac{\theta \Delta t}{\rho_o}\int_s^n \int_w^e \frac{\partial}{\partial x}A_x^n z\,dxdy$$

$$= \frac{\Delta t}{\rho_o}\left[-\int_s^n \int_w^e \frac{\partial \mathrm{F}^n}{\partial x}dxdy - \int_w^e \int_s^n \frac{\partial \mathrm{G}^n}{\partial y}dydx + \int_s^n \int_w^e Q_m^n\,dxdy\right] \tag{A.9}$$

Or:

$$z_P \Delta x_c \Delta y_c + \frac{\theta \Delta t}{\rho_o}[(A_x^n z)_e - (A_x^n z)_w]\Delta y_c$$

$$= \frac{\Delta t}{\rho_o}\{-[(\mathrm{F}^n)_e - (\mathrm{F}^n)_w]\Delta y_c - [(\mathrm{G}^n)_n - (\mathrm{G}^n)_s]\Delta x_c + Q_m^n \Delta x_c \Delta y_c\} \tag{A.10}$$

Dividing both sides by $V_c = \Delta x_c.\Delta y_c$ (cell volume):

$$z_P + \frac{\theta \Delta t}{V_c \rho_o}[(A_x^n z)_e - (A_x^n z)_w]\Delta y_c$$
$$= \frac{\Delta t}{V_c \rho_o}\{-[(\mathrm{F}^n)_e - (\mathrm{F}^n)_w]\Delta x_c - [(\mathrm{G}^n)_n - (\mathrm{G}^n)_s]\Delta y_c + Q_m^n\} \tag{A.11}$$

Δx_c and Δy_c are the cell dimensions in x and y direction, respectively. Approximation of A_x and the intermediate solution z at the east and west faces of the control volume leads to a system of algebraic equations:

$$a_P z_P = a_E z_E + a_W z_W + RHS_P \tag{A.12}$$

where RHS_P is the right hand side of equation A.19 computed at point P, and:

$$a_P = 1 - (a_P + a_W)$$

RHS_P contains the fixed value of the source term. Expressions for a_P and a_W depend on the scheme used to approximate A_x and z at the cell faces, especially the convective terms.

Reconstruction of equations showed that the upwind scheme is used in hygIRC (see below for the computation of the coefficients). After computing the coefficients and solving for z, the following system is integrated, discretized and solved for the final solution Δu:

$$\left(I + \frac{\theta \Delta t}{\rho_o} \frac{\partial}{\partial y} A_y^n\right) \Delta u^{n+1} = z \tag{A.13}$$

Heat equation

The heat transport equation can be rewritten in the case of a 2D problem as:

$$\rho_o c \frac{\partial T}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} + Q_i = Q_h \tag{A.14}$$

Where *F* and *G* are the total heat fluxes in x and y direction:

$$F = \rho_a c_a V_x T + \lambda_x \frac{\partial T}{\partial x} + \rho_o L_v \delta_{P,x} \frac{\partial p_v}{\partial x}$$

$$G = \rho_a c_a V_y T + \lambda_y \frac{\partial T}{\partial y} + \rho_o L_v \delta_{P,y} \frac{\partial p_v}{\partial y}$$

Q_i if the latent heat due the freezing of water or melting of ice:

$$Q_i = L_{ice}\left(\rho_o u \frac{\partial f_l}{\partial t}\right)$$

After time integration and factorization, we get:

$$\left[\left(I + \frac{\theta \Delta t}{\rho_o c} \frac{\partial}{\partial x} A_x^n + H^n\right)\left(I + \frac{\theta \Delta t}{\rho_o c} \frac{\partial}{\partial y} A_y^n\right)\right] \Delta T^{n+1} = \frac{\Delta t}{\rho_o c}\left[-\frac{\partial \mathrm{F}^n}{\partial x} - \frac{\partial \mathrm{G}^n}{\partial y} - Q_i^n + Q_h^n\right] \tag{A.15}$$

where

$$A_x^n = \frac{\partial F}{\partial T} = \frac{\partial}{\partial T}\left(\rho_a c_a V_x \frac{\partial T}{\partial x} + \lambda_x \frac{\partial T}{\partial x} + \rho_o L_v \delta_{P,x} \frac{\partial p_v}{\partial x}\right)$$

$$A_y^n = \frac{\partial G}{\partial T} = \frac{\partial}{\partial T}\left(\rho_a c_a V_y \frac{\partial T}{\partial y} + \lambda_y \frac{\partial T}{\partial y} + \rho_o L_v \delta_{P,y} \frac{\partial p_v}{\partial y}\right) \tag{A.16}$$

$$H^n = \frac{\partial Q_i}{\partial T}$$

As for moisture equation, heat equation is solved in two steps:

Step 1: Solve for intermediate solution z

$$\left(I + \frac{\theta \Delta t}{\rho_o c}\frac{\partial}{\partial x}A_x^n + H^n\right) z = \frac{\Delta t}{\rho_o c}\left[-\frac{\partial \mathrm{F}^n}{\partial x} - \frac{\partial \mathrm{G}^n}{\partial y} - Q_i^n + Q_h^n\right] \tag{A.17}$$

Spatial integration over the control volume leads to:

$$\begin{aligned} z_P(1 + H^n) + \frac{\theta \Delta t}{\rho_o}[(A_x^n z)_e - (A_x^n z)_w]\Delta x \\ = \frac{\Delta t}{\rho_o}\{-[(\mathrm{F}^n)_e - (\mathrm{F}^n)_w]\Delta x - [(\mathrm{G}^n)_n - (\mathrm{G}^n)_s]\Delta y - Q_i^n + Q_h^n\} \end{aligned} \tag{A.18}$$

Approximations of A_x and z at the cell faces lead in this case to the following system of algebraic equations to be solved for x-sweep:

$$a_P z_P = a_E z_E + a_W z_W + RHS_P \tag{A.19}$$

where:

$$a_P = 1 - (a_P + a_W) + H$$

Then for y-sweep, the final solution is found by solving:

$$\left(I + \frac{\theta \Delta t}{\rho_o c}\frac{\partial}{\partial y}A_y^n\right)\Delta T^{n+1} = z \tag{A.20}$$

The cell face fluxes, in the right hand side (RHS) of Eqs. (A.11) and (A.18), are calculated in subroutines QM and QT, respectively, and the RHS for both equations are calculated in subroutine RIGHT. The derivatives of the flux with respect to the potential u or T, A_x, at the east and west faces of the control volume, appearing in the left hand side (LHS) of Eqs. (A.11) and (A.18) are computed in the subroutines COEFST while A_y at the south and north faces are computed in the subroutine COEFRL. The coefficients a_P, a_E, and a_W are formed in the subroutine DUSTAR.

Computation of cell face fluxes: subroutines QM and QT

Subroutine QM

It computes moisture and vapour fluxes between node points in x (Figure A.1) and y directions. In x direction we can write:

$$q_m = -\rho_o D_{w,x}\frac{\partial u}{\partial x} + k_{w,x}\rho_w g - \delta_P \frac{\partial p_v}{\partial x} + \rho_v V_{a,x} \tag{A.21}$$

Case the two adjacent cells belong to the same material

For the case where the two adjacent cells, i and i+1, belong to the same material, the total moisture flux at the interface of the two cells is calculated as:

$$q_m = q_l + q_v$$

where

q_m: total moisture flux at the interface of cell i and cell i+1

q_l: liquid flux at the interface of cell i and cell i+1

q_v: vapour flux at the interface of cell i and cell i+1

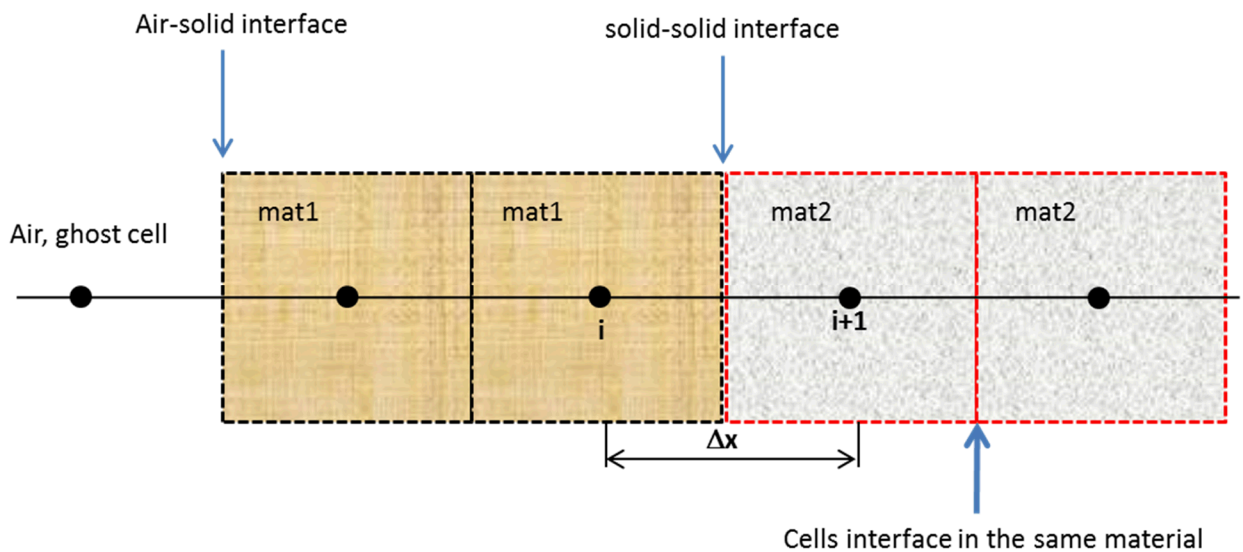

Figure A.1. Arrangement of cells in x direction.

Total moisture flux

Diffusive term

The liquid flux is the sum of flux due to diffusion under the gradient of u, q_{lu}, and the liquid flux due to gravity, q_{lg}.

$$q_{lu} = \rho_o D_w \frac{u_i - u_{i+1}}{\Delta x} \tag{A.22}$$

The implementation is:

$$q_{lu} = \frac{\rho_o}{DX(i)} f\,[DWPOT(i) - DWPOT(i+1)]$$

where *DWPOT* is the array containing the product of moisture diffusivity D_w and the moisture content u established in the subroutine CALCPOT, f is a factor computed as:

$$f = f_1(\bar{T}) \times f_2$$

where $\bar{T}$ the average temperature at node i and i+1, f_1 is a correction factor equal to 0 if $\bar{T} < -5°C$ or equal to 1 if $\bar{T} > 0°C$, and varying from 0 to 1 when $\bar{T}$ is between -5 and 0°C. f_2 seems to be the water density at the average temperature. The formula to compute what seems to be the ratio between the dynamic viscosity of water at reference temperature and the viscosity of water at $\bar{T}$ in subroutine FVISC is:

$$f_2 = \frac{1.004}{C0 + C1T + C2T^2 + C3T^3 + C4T^4}$$

where C0 = 1.76632, C1 = -0.0538082, C2 = 0.000942267, C3 = -8.52254 10^{-6}, and C4 = 3.01283 10^{-8}. For example, at T = 20°C, the result for f_2 is 1.0003.

Gravity term

The flux due to gravity is:

$$q_{lg} = \rho_w k_w g \tag{A.23}$$

where k_w is the water permeability, computed by equating the flux of water under the gradient of moisture content and the flux of water under the gradient of suction pressure:

$$\rho_o D_L \frac{\partial u}{\partial x} = k_w \frac{\partial s}{\partial x} \rightarrow k_w = \rho_o D_L \frac{\partial u}{\partial s}$$

The implementation accounts for the wall inclination, β, so that the gravity flux is:

$$q_{lg} = \rho_w \rho_o \bar{D}_L \frac{\partial u}{\partial s} g \cos\left(\frac{\beta\pi}{180}\right) \tag{A.24}$$

$\frac{\partial u}{\partial s} = 1/DPSUCDU(im, \bar{u}, \bar{T})$ where *im* is the material number, and DPSUCDU is the function that computes the derivative $\frac{\partial s}{\partial u}$. $\bar{u}$, $\bar{T}$ and $\bar{D}_L$ are computed as:

$$\bar{u} = (1-c)u_i + cu_{i+1}$$

$$\bar{T} = (1-c)T_i + cT_{i+1}$$

$$\bar{D}_L = (1-c)D_{L,i} + cD_{L,i+1}$$

where $\begin{cases} c = 1 \; if \; \beta \; greater \; than \; 90° \\ c = 0 \; otherwise \end{cases}$

D_L, the liquid diffusivity, is calculated in subroutine DIFL and is roughly equal to $D_w - \delta_p$. A correction is made on the value of q_{lg}, depending on the actual moisture content u at position i+1, i.e.: if $u_{i+1} \geq 0.99u_{max}$ then $q_{lg} = \frac{\max(u_{max} - u_{i+1},\ 0)}{0.01\, u_{max}} . q_{lg}$ where u_{max} is the maximum moisture content of the material. Thi means that if $u_{i+1} = u_{max}$, the gravity term at position i is cancelled.

For the gravity flow in y direction, the relation is:

$$q_{lg} = -1 * \rho_w \rho_o \overline{D}_L \frac{\partial u}{\partial s} g \sin\left(\frac{\beta\pi}{180}\right)$$

The total liquid flux at cell face is stored in the array QMOIX(N1-1,N2-1) for x direction and in QMOIY(N1-1, N2-1).

Total vapour flux

The total vapour flux, q_v, is the sum of the diffusive and the convective fluxes of vapour:

$$q_v = -\delta_{p,x} \frac{\partial p_v}{\partial x} + \rho_v V_{a,x}$$

Since

$$\rho_v = \frac{M_w}{RT} p_v$$

$$q_v = -\delta_{p,x} \frac{\partial p_v}{\partial x} + \frac{M_w}{RT} V_{a,x}\, p_v$$

Due to the convection term, a particular scheme is used to estimate q_v at the interface:

$$q_v = -\delta_{p,x}\beta \frac{p_{v,i+1} - p_{v,i}}{\Delta x} + \frac{M_w}{R(\overline{T} + 273)} V_{a,x} \left[(0.5 + \alpha)p_{v,i} + (0.5 - \alpha)p_{v,i+1}\right] \qquad \text{(A.25)}$$

R is the gas constant, M_w is the molecular mass of water and $\overline{T}$ is the average temperature at nodes i and i+1, Δx is the distance between nodes i and i+1, obtained from the vector DX, $\delta_{p,x}$ is the vapour permeability at the interface obtained from the array VAPV, p_v is the vapour pressure from the vector PVN, and α and β are computed as:

$$\alpha = \frac{1 + 0.005{P_e}^2}{1 + 0.05P_e^2}$$
$$\beta = \frac{P_e \times |P_e|}{10 + 2{P_e}^2} \qquad \text{(A.26)}$$

where P_e seems to be the Peclet number, computed as:

$$P_e = \frac{\rho_v V_{a,x}}{\frac{\delta_{p,x}}{\Delta x}} = \frac{V_{a,x}\, M_w\, \Delta x}{\delta_{p,x} R(\overline{T} + 273)}$$

The total vapour flux is stored in the array QMOVX(N1-1, N2-1) for x direction and in QMOVY(N1-1, N2-1) for y direction.

Case the two adjacent cells belong to different materials: subroutine QMOLDI

When the two adjacent cells belong to different materials as shown in Figure A.1, QM calls the subroutine QMOLDI to compute moisture fluxes at material interfaces. The algorithm to compute the flux at the interface of two materials (air-solid, solid-solid, or solid-air), going from the exterior to the interior, is:

1) Get or calculate the temperature at the interface of material 1 and 2
2) Start with a given *RH*, calculate the vapour pressure at the interface, $p_{v,f}$.
3) Calculate the total moisture flux QM1, from point i to the interface in material 1
4) Calculate the total moisture flux QM2, from the interface to point i+1 in material 2
5) Calculate F = QM1 - QM2
6) Change *RH* and return to step 2 and repeat until the absolute value of F is less than 10^{-13}

Before starting the loop, the temperature at the interface is retrieved from the array TSURF if the interface is at the boundary of the domain, or calculated if the interface is inside the wall, using the following formula:

$$T_f = \frac{a_i T_i + a_{i+1} T_{i+1} + L_v (q_l - q_v)}{a_i + a_{i+1}}$$

with: $a_i = \frac{\lambda_i}{\frac{\Delta x}{2}} + \max(V_a, 0)$ and $a_{i+1} = \frac{\lambda_{i+1}}{\frac{\Delta x}{2}} - \max(V_a, 0)$

where λ is the thermal conductivity, q_l and q_v are the liquid and vapour flux values obtained at previous time step. If material i is air, the thermal conductivity is replaced by the heat transfer coefficient. *Ts* is then used to calculate the saturated vapour pressure at the interface, from which the vapour pressure at the interface is calculated from the relative humidity value tested at each iteration.

Case the material1 is solid

The liquid flux q_{1l} due to diffusion is computed in the same manner as above (Eqs. (A.22) to (A.24), excepted that i represents the node i in material1 and i+1 represents the interface, implying that the distance is now Δx/2. u at the face is calculated as the average between the moisture content u of the material and the moisture content derived from the sorption curve using the value of the relative humidity that is being tested. It should also be noted that when working in the y direction, the gravity term is computed as:

$$q_{lg} = \rho_w \rho_o \bar{D}_w \frac{\partial u}{\partial s} g * -\sin\left(\frac{\beta \pi}{180}\right) \tag{A.27}$$

The vapour flux is computed as:

$$q_{1v} = \delta_{p,f} \frac{p_{v,i} - p_{v,f}}{0.5\ \Delta x} + \frac{V_f\ M_w p_{v,i}}{R\ (T_f + 273)}$$

where subscript *f* means the value at the interface. $\delta_{p,f}$ is interpolated from the vapour permeability and relative humidity table using T_f and RH_f (subroutine VPER). In case V_f is negative, the convective term is calculated using $p_{v,f}$ rather than $p_{v,i}$.

Case material1 is air (outdoor boundary)

$$q_{1l} = 0$$

$$q_{1v} = \beta_e \left(p_{v,a} - p_{v,f}\right) + \frac{V_f\ M_w p_{v,a}}{R\ (T_a + 273)}$$

The first term on the right hand side is computed by the function QMAIR where βe is the external moisture transfer coefficient. Here also, if V_f is negative, the convective term is calculated using $p_{v,f}$ and T_f.

Case material2 is solid

q_{2l} is computed using Eqs. (A.22) to (A.24) and (A.27), excepted the that the position I is now the interface and the i+1 is the node within material 2 and that the distance is Δx/2.

$$q_{2v} = \delta_{p,f} \frac{p_{v,f} - p_{v,i+1}}{0.5\ \Delta x} + \frac{V_f\ M_w p_{v,f}}{R\ (T_f + 273)}$$

If V_f is negative, the convective term is computed using p_{vi+1} and T_{i+1}

Case material2 is air (e.g.: indoor boundary)

$$q_{2l} = 0$$

$$q_{2v} = \beta_i \left(p_{v,f} - p_{v,air}\right) + \frac{V_f\ M_w p_{v,f}}{R\ (T_f + 273)}$$

where β_i is the indoor moisture transfer coefficient. If V_f is negative, the convective term is computed using p_{vi+1} and T_{i+1}

<u>Note 1</u>:

- Outflow of gravity

  In case the interface is at the left or bottom boundary of the domain, Q1l is corrected as:

  $$q_{1l} = q_{1l} + \Delta q_g$$

  where $\Delta q_g = q_{lg2} - q_{lg1}$.

- For the external boundary and for the nodes below the top node, the rain flux and rain rundown are taken into account by adding their values to F. The rain run down is stored in the vector QRD.

The values of the vapour flux from the interface to i+1 in the material2, q_{2v}, is stored in the array XV(2,N1,N2). The value of vapour flux that is returned to QM is the vapour flux from i to the interface in material1, q_{1v}.

Subroutine QMOLDI also computes the derivatives $\partial p_v/\partial u_1$, $\partial p_v/\partial u_2$, and $\partial p_v/\partial T_f$ and store in array DPV(6, N1, N2). DPV(1:3, :, :) stores values for the x direction while DPV(4:6, :, :) stores values for the y direction. In the case of 1D and of a wall with two layers having 3 and 4 internal nodes, respectively:

$$DPV(1,:,2) = \left[\left(\frac{\partial p_v}{\partial u_1}\right)_{air/solid} \quad 0 \quad 0 \quad 0 \quad \left(\frac{\partial p_v}{\partial u_1}\right)_{soli/solid} \quad 0 \quad 0 \quad 0 \quad 0 \quad \left(\frac{\partial p_v}{\partial u_1}\right)_{solid/air}\right]$$

$$DPV(2,:,2) = \left[\left(\frac{\partial p_v}{\partial u_2}\right)_{air/solid} \quad 0 \quad 0 \quad 0 \quad \left(\frac{\partial p_v}{\partial u_2}\right)_{soli/solid} \quad 0 \quad 0 \quad 0 \quad 0 \quad \left(\frac{\partial p_v}{\partial u_2}\right)_{solid/air}\right]$$

$$DPV(3,:,2) = \left[\left(\frac{\partial p_v}{\partial T_s}\right)_{air/solid} \quad 0 \quad 0 \quad 0 \quad \left(\frac{\partial p_v}{\partial T_f}\right)_{soli/solid} \quad 0 \quad 0 \quad 0 \quad 0 \quad \left(\frac{\partial p_v}{\partial T_f}\right)_{solid/air}\right]$$

u_1 and u_2 refer to moisture content of cell1 and cell2, from exterior to interior, and T_f refers to the interface temperature.

Handling of wind-driven rain

As mentioned above, when there is a rain event, hygIRC computes the amount of rain that is absorbed at the outer boundary. The amount that is not absorbed is considered as rundown and is kept in the array QRD. At the next time step, this amount is considered as extra rain load for the node underneath the current node (applicable for 2D version only) and added to the new rain event if there is any.

Assuming that at the current time step t, there is a rain event $rain_t$ and that there was a rundown at previous step, $rain_{rd}$, the manner in which the rain absorption ($rain_{abs}$) and rundown are computed is:

1) Compute total rain event arriving on node i: $g_{wdr} = rain_t + rain_{rd}$
2) Compute $F = QM1V - (QM2L + QM2V) + g_{wdr}$
3) After finding the vapour pressure or the relative humidity at the interface that makes $|F| < 10^{-13}$ (as described above):
   - IF $rain_t < 10^{-9}$, <u>there is no rundown</u>
   - IF $rain_t > 10^{-9}$, <u>there is possible rundown</u>

     $rain_{rd} = \max(F, 0)$

     $rain_{abs} = QM2L + QM2V - QM1V$

QM2L and QM2V are the total liquid and vapour fluxes in the material close to the surface while QM1V is the total vapour fluxes arriving or leaving the surface.

Subroutine QT

The subroutine QT computes heat fluxes at the cell faces. For the x direction, the heat flux is expressed as:

$$q_{h,x} = -\lambda_x \frac{\partial T}{\partial x} - L_v \rho_o \delta_{P,x} \frac{\partial p_v}{\partial x} + L_v \rho_v V_a + \rho_a c_a V_{a,x} T \tag{A.28}$$

Implementation:

$$q_h = q_{h,\lambda} + q_{h,l} + q_{h,a}$$

Case cell1 and cell2 belong to the same material or solid-solid interface

Conduction

$$q_{h,\lambda} = \beta \frac{\lambda_f (T_i - T_{i+1})}{\Delta x}$$

where λ_f, the thermal conductivity at the interface between the two cells, is obtained from array CNDX (in the case the two solid materials are different, λ_f is the harmonic mean of the two thermal conductivities); Δx, the distance between the two nodes, is obtained from DX, and β, a correction factor, is the same as in eq. A.26, but with P_e now defined as:

$$P_e = \frac{V_{x,f}\, \rho_a\, c_a \Delta x}{\lambda_f}$$

Latent heat

$$q_{h,l} = L_v q_v \tag{A.29}$$

where q_v is the total vapour flux at the interface, obtained from array QMOV and L_v is the latent heat of evaporation/condensation.

Advection

$$q_{h,a} = \rho_a c_a V_{x,f} [(0.5 + \alpha) T_i + (0.5 - \alpha) T_{i+1}]$$

with α the same as in equation A.26.

The total heat flux at the cell interfaces is stored in the array QTX(N1-1, N2-1).

Case air-solid (outdoor wall surface)

The heat radiation is taken into account at the air-solid interface so that the total heat flux is:

$$q_h = q_{h,e} + q_{h,\varepsilon} + q_{h,a} + q_{h,l}$$

where $q_{h,e}$ is the external heat transfer by convection, $q_{h,\varepsilon}$ the radiation term, $q_{h,a}$ is the advection term due to air movement, and $q_{h,l}$ is the latent heat transport term.

External (outdoor) heat transfer by convection/radiation

$$q_{h,e} + q_{h,\varepsilon} = h_e(T_e - T_S) - \varepsilon\sigma\left({T_S}^4 - {T_{sky}}^4\right)$$

where

h_e: equivalent heat transfer coefficient (Eq. A.4)

T_e: sol-air or effective outdoor temperature (Eq. A.2)

T_S: effective surface temperature (K)

ε: emissivity of the wall

σ: Stefan-Boltzmann constant

T_{sky}: sky temperature (K)

T_{sky} is the value of the sky temperature (SKYTEMP) read from input file when ISKY(j) = 1, j being the bottom, top, left or right side of the wall and in which case it was stored in the array SKYT, or the value of the sky temperature calculated in subroutine WEATH using weather data. T_S, the equivalent surface temperature, is computed as:

$$T_S = \frac{a_i T_e + a_{i+1} T_{i+1} + L_v\left(Q_{v,i} - Q_{v,i+1}\right) - \rho_a c_a V_f \Delta T_o}{a_i + a_{i+1}}$$

with:

$$a_i = h_e + \rho_a c_a \max(V_f, 0)$$

$$a_{i+1} = \frac{\lambda_{i+1}(u_{i+1}, T_{i+1})}{\Delta x_{1/2}} - \rho_a c_a \min(V_f, 0)$$

$$\begin{cases} \Delta T_o = T_e - T_a \; if \; V_f > 0 \\ \Delta T_o = 0 \; otherwise \end{cases}$$

where:

T_{i+1}: temperature at node i+1

$Q_{V,i}$: vapour flux arriving at the surface

$Q_{V,i+1}$: vapour flux leaving the surface and entering the wall

T_a: ambient air temperature

$\lambda_{i+1}(u_{i+1}, T_{i+1})$: thermal conductivity at node i+1, calculated by COND

In the program, a_i is stored in the variable CND1 and a_{i+1} is stored in the variable CND2.

Advection term

$$q_{h,a} = \rho_a c_a \left[V_f T_X - \max(V_f, 0)\, \Delta T_o\right]$$

$$\begin{cases} T_X = T_{S,e} \; if \; V_f \; < 0 \\ T_X = T_{e,o} \; if \; V_f \; > 0 \end{cases}$$

Latent heat term

Computed as in Eq. (A.29).

Case solid-air (indoor wall surface)

The heat radiation is taken into account at the solid-air interface so that the total heat flux is:

$$q_h = q_{h,c} + q_{h,\varepsilon} + q_{h,a} + q_{h,l}$$

where $q_{h,c}$ is the heat transfer by conduction, $q_{h,\varepsilon}$ is the radiation term, $q_{h,a}$ is the advection term due to air movement, $q_{h,l}$ is the latent heat transport term.

Heat transfer by conduction/radiation

$$q_{h,i} + q_{h,\varepsilon} = \frac{\lambda_i}{\Delta x_{1/2}}\left(T_i - T_{S,in}\right) + \varepsilon\sigma\left({T_{S,in}}^4 - {T_{sky}}^4\right)$$

where λ_i and T_i are the thermal conductivity and temperature at the node i, respectively, and $T_{S,in}$ is the indoor equivalent surface temperature computed as:

$$T_{S,in} = \frac{a_i T_i + a_{i+1} T_{e,in} + L_v\left(Q_{v,i} - Q_{v,i+1}\right) - \rho_a c_a V_f \Delta T_{in}}{a_i + a_{i+1}}$$

with:

$$a_i = \frac{\lambda_m(u_i, T_i)}{\Delta x_{1/2}} + \rho_a c_a \max(V_f, 0)$$

$$a_{i+1} = h_{e,in} - \rho_a c_a \min(V_f, 0)$$

$$\begin{cases} \Delta T_{in} = -\left(T_{e,in} - T_{a,in}\right) \; if \; V_f < 0 \\ \Delta T_{in} = 0 \; otherwise \end{cases}$$

where:

T_{i+1}: temperature at node near the surface

$Q_{V,i}$: vapour flux from node i to the surface, obtained from array QMOVX

$Q_{V,i+1}$: vapour flux leaving the surface and entering the wall, obtained from array XV

$T_{e,in}$: effective or equivalent indoor temperature

$h_{e,in}$: equivalent heat transfer coefficient ($h_c + h_r$)

$\lambda_m(u_i, T_i)$: thermal conductivity computed by subroutine COND using u and T at node i

The radiation component for the indoor is taken into account the computation of the effective surface temperature $T_{e,in}$ and in the equivalent heat transfer coefficient, $h_{e,in}$.

$$T_{e,in} = T_{a,in} + \frac{\alpha_{in} I_{T,in}}{h_{e,in}}$$

$$h_{e,in} = h_{c,in} + \varepsilon_{in}\sigma(\bar{T}_{in} + 273.15)^3$$

$I_{T,in}$ accounts only for long-wave radiation. $\bar{T}_{in}$ is some sort of average between effective surface temperature and the indoor air temperature:

$$\bar{T}_{in} = \frac{h_{c,in}T_{in} + \frac{\lambda_i T_i}{\Delta x_{1/2}} + L_v\left(Q_{v,i} - Q_{v,i+1}\right)}{h_{c,in} + \frac{\lambda_i}{\Delta x_{1/2}}} + T_{a,in}$$

Advection term

$$q_{h,a} = \rho_a c_a\left[V_f T_X - \Delta T_{in} * \max(V_f, 0)\right]$$

in which: $\begin{cases} T_X = T_{S,in} \; if \; V_f \; < 0 \\ T_X = T_{e,in} \; if \; V_f \; > 0 \end{cases}$

Latent heat

Computed as in Eq. (A.29).

Storage of surface temperature

The effective surface temperature computed for each boundary node is stored in array TSURF(4, NMAX):

$$TSURF = \begin{bmatrix} TS_{b1} & TS_{b2} & \cdots & TS_{bn} \\ TS_{t1} & TS_{t1} & \cdots & TS_{t1} \\ TS_{l1} & TS_{l1} & \cdots & TS_{l1} \\ TS_{r1} & TS_{r1} & \cdots & TS_{r1} \end{bmatrix}$$

Where b, t, l, and r stands for bottom, top, left, and right nodes, respectively.

Computation of the right hand side: subroutine RIGHT

The subroutine RIGHT computes the RHS of Eqs. (A.11) and (A.18) at current iteration, using the results obtained in subroutine QM and QT. The results are stores in the 3D array RHS(2, N1,N2) where RHS(1, :, :) stores results for moisture equation and RHS(2, :, :) stores results for heat equation. The nodal point (i,j) represents the point P as shown in Fig. 1.

Case external nodes

Moisture equation

$$RHS(1,i,j) = p_v - {p_v}^n$$

p_v is the value of boundary vapour pressure at current iteration and p_v^n is the value of the boundary vapour pressure at previous time. The result is nil since boundary p_v remains unchanged throughout iterations.

Heat equation

$$RHS(2,i,j) = T - T^n$$

T is the equivalent outdoor temperature at current iteration. The result is nil since boundary T remains unchanged throughout iterations.

Case internal nodes

Moisture equation

$$RHS(1,i,j) = \frac{\Delta t}{\rho_o}\left\{\left(\alpha_x q_{m,w}\Delta x_w - \beta_x q_{m,e}\Delta x_e\right) + \left(\alpha_y q_{m,s}\Delta y_s - \beta_y q_{m,n}\Delta y_n\right) + Q_m\right\} - (u - u^n)$$

u is the moisture content at current iteration, u^n is the moisture content at previous time step, the subscripts w, e, s, and n represent the cell west, east, south and north face, respectively, and α and β are defined as:

$$\alpha_x = \frac{2}{\Delta x_w(\Delta x_w + \Delta x_e)}$$

$$\beta_x = \frac{2}{\Delta x_e(\Delta x_w + \Delta x_e)}$$

$$\alpha_y = \frac{2}{\Delta y_s(\Delta y_s + \Delta y_n)}$$

$$\beta_y = \frac{2}{\Delta y_n(\Delta y_s + \Delta y_n)}$$

Heat equation

$$\begin{aligned} RHS(2,i,j) = \frac{\Delta t}{\rho_o c}\Big\{&\left(\alpha_x q_{h,w}\Delta x_w - \beta_x q_{h,e}\Delta x_e - q_{\varepsilon,x}\right) \\ &+ \left(\alpha_y q_{h,s}\Delta y_s - \beta_y q_{h,n}\Delta y_n - q_{\varepsilon,y}\right) + Q_h - Q_L\Big\} - (T - T^n) \end{aligned} \tag{A.30}$$

with:

$$Q_L = \rho_o L_i u^n \frac{(f_l - {f_l}^n)}{\Delta t}$$

T is the temperature at point P at current iteration, T^n is the temperature at previous time step at node point P. q_ε represents the internal surface to surface radiation and is calculated as:

$$q_{\varepsilon,x} = 5.67 F_x F_{\varepsilon,x} \left[\left(\frac{T_i + 273}{100} \right)^4 - \left(\frac{T_{i+ICR} + 273}{100} \right)^4 \right]$$

where T_i is the temperature at node P, T_{i+ICR} is the temperature at node at position i+ICR, icr being read in array ICON, f_l is the liquid fraction at current iteration, f_l^n is the liquid fraction at previous time step, F and F_ε are geometric and the emissivity factors calculated as:

$$F_\varepsilon = \frac{1}{\frac{100}{\varepsilon_i} + \frac{100}{\varepsilon_{i+ICR}} - 1}$$

$$\begin{cases} F_x = \beta_x \Delta x_e \; if \; ICR > 0 \\ F_x = \alpha_x \Delta x_w \; if \; ICR < 0 \end{cases}$$

The internal radiation term is computed if ICR = ICON(i,j) is not equal to zero. ε_i and ε_{i+ICR} are read in array IEM. Both arrays ICON and IEM are created in subroutine hygIRC when the file rad.dat exists. The same calculations are performed for $q_{\varepsilon,y}$.

Note: the term 1/100 in bracket is to take into account the 10^{-8} of the Stefan-Bolzmann constant, since (1/100)**4 = 10^{-8}.

In the case of air layer ($\rho < 1.5$), here is how RHS is calculated:

$$RHS(2,i,j) = RHS1 - \frac{\Delta t}{\rho_o c} \{ [\alpha_x q_{v,w} L_v \Delta x_w - \beta_x q_{v,e} L_v \Delta x_e] + [\alpha_y q_{v,s} L_v \Delta y_s - \beta_y q_{v,n} L_v \Delta y_n] \}$$

Where RHS1 is the one computed in Eq. A.30. Basically, it removes from q_h the contribution of latent heat transfer due to evaporation/condensation of vapour. The terms qv,w and qv,e are calculated as:

$$q_{v,w} = \alpha QMOVX(i-1,j) - (1-\alpha) QMVOLX(i-1,j)$$

$$q_{v,e} = \alpha QMOVX(i,j) - (1-\alpha) QMVOLX(i,j)$$

$$q_{v,s} = \alpha QMOVY(i,j-1) - (1-\alpha) QMVOLY(i,j-1)$$

$$q_{v,n} = \alpha QMOVY(i,j) - (1-\alpha) QMVOLY(i,j)$$

In case the material at position i-1 is not equal to material at position i, or material at position j-1 not equal to material at position j:

$$q_{v,w} = \alpha XV(1,i-1,j) - (1-\alpha) XVOLD(1,i-1,j)$$

$$q_{v,s} = \alpha XV(2,i,j-1) - (1-\alpha) XVOLD(2,i,j-1)$$

Computation of matrix elements (left hand side)

For each row (j = 1, N2), DUSTAR calls COEFST to compute the jacobians at the interface ***f***, between node i and node i+1 and store in A(2, 2, NMAX), B(2, 2, NMAX), and C(2, 2, NMAX), then uses the results to form the arrays AAA(2, 2, NMAX), BBB(2, 2, NMAX) and CCC(2, 2, NMAX) and finally the matrix coefficients AU (10, N1, N2) and AT(10, N1, N2). For the arrays A, B, C, and AAA, BBB, and CCC, the first line and first column stores values for moisture equation while the second line and second column stores value for heat equation

To illustrate, we will use the case of 1D where j = 3, and i = 5 (2 boundary nodes and 3 internal nodes in x direction). The max nodes in this case is 5.

For j = 1 (bottom nodes)

Subroutine COEFST

A(1,:,1) A(2,:,1)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	B(1,:,1) B(2,:,1)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	C(1,:,1) C(2,:,1)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
A(1,:,2) A(2,:,2)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	B(1,:,2) B(2,:,2)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	C(1,:,2) C(2,:,2)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
A(1,:,3) A(2,:,3)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	B(1,:,3) B(2,:,3)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	C(1,:,3) C(2,:,3)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
A(1,:,4) A(2,:,4)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	B(1,:,4) B(2,:,4)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	C(1,:,4) C(2,:,4)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
A(1,:,5) A(2,:,5)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	B(1,:,5) B(2,:,5)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	C(1,:,5) C(2,:,5)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$

Subroutine DUSTAR

AAA(1,:,1) AAA(2,:,1)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	BBB(1,:,1) BBB(2,:,1)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	CCC(1,:,1) CCC(2,:,1)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
AAA(1,:,2) AAA(2,:,2)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	BBB(1,:,2) BBB(2,:,2)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	CCC(1,:,2) CCC(2,:,2)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
AAA(1,:,3) AAA(2,:,3)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	BBB(1,:,3) BBB(2,:,3)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	CCC(1,:,3) CCC(2,:,3)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
AAA(1,:,4) AAA(2,:,4)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	BBB(1,:,4) BBB(2,:,4)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	CCC(1,:,4) CCC(2,:,4)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
AAA(1,:,5)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	BBB(1,:,5)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	CCC(1,:,5)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$

AAA(2,:,5)	BBB(2,:,5)		CCC(2,:,5)	
AU(1,:,1)	0	0	0	0
AU(2,:,1)	0	0	0	0
AU(3,:,1)	0	0	0	0
AU(4,:,1)	0	0	0	0
AU(5,:,1)	0	0	0	0
AU(6,:,1)	**0**	**0**	**0**	**0**
AU(7,:,1)	**0**	**0**	**0**	**0**
AU(8,:,1)	0	0	0	0
AU(9,:,1)	0	0	0	0
AU(10,:,1)	0	0	0	0
AT(1,:,1)	**0**	**0**	**0**	**0**
AT(2,:,1)	**0**	**0**	**0**	**0**
AT(3,:,1)	0	0	0	0
AT(4,:,1)	0	0	0	0
AT(5,:,1)	0	0	0	0
AT(6,:,1)	0	0	0	0
AT(7,:,1)	0	0	0	0
AT(8,:,1)	0	0	0	0
AT(9,:,1)	0	0	0	0
AT(10,:,1)	0	0	0	0

Note: the bold values are the ones that have been modified since all arrays are initialized to 0.

For j = 2

Subroutine COEFST

A(1,:,1) A(2,:,1)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$	B(1,:,1) B(2,:,1)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	C(1,:,1) C(2,:,1)	$\begin{bmatrix} A_{e,u} & 0 \\ 0 & A_{e,T} \end{bmatrix}$
A(1,:,2) A(2,:,2)	$\begin{bmatrix} A_{w,u} & 0 \\ 0 & A_{w,T} \end{bmatrix}$	B(1,:,2) B(2,:,2)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	C(1,:,2) C(2,:,2)	$\begin{bmatrix} A_{e,u} & 0 \\ 0 & A_{e,T} \end{bmatrix}$
A(1,:,3) A(2,:,3)	$\begin{bmatrix} A_{w,u} & 0 \\ 0 & A_{w,T} \end{bmatrix}$	B(1,:,3) B(2,:,3)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	C(1,:,3) C(2,:,3)	$\begin{bmatrix} A_{e,u} & 0 \\ 0 & A_{e,T} \end{bmatrix}$
A(1,:,4) A(2,:,4)	$\begin{bmatrix} A_{w,u} & 0 \\ 0 & A_{w,T} \end{bmatrix}$	B(1,:,4) B(2,:,4)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	C(1,:,4) C(2,:,4)	$\begin{bmatrix} A_{e,u} & 0 \\ 0 & A_{e,T} \end{bmatrix}$
A(1,:,5) A(2,:,5)	$\begin{bmatrix} A_{w,u} & 0 \\ 0 & A_{w,T} \end{bmatrix}$	B(1,:,5) B(2,:,5)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	C(1,:,5) C(2,:,5)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

$A_{w,u}$ and $A_{e,u}$, and $A_{w,T}$ and $A_{e,T}$ are the intermediate coefficients for moisture content and temperature at the west of node i+1 and the east of node i and are computed as:

Case cells i and i+1 are in the same material

$$A_{w,u} = \left\{\frac{-\rho_0 \overline{D}_L}{\Delta x_i} - \left[\left(\frac{\delta_{p1}}{\Delta x_i} + \max\left(V_{a,f}, 0\right) \rho_{v1}\right) \frac{\partial p_v}{\partial u_1}\right]\right\} \left(\frac{\Delta y_1 + \Delta y_2}{2}\right)$$

$$A_{e,u}(i) = \left\{\frac{-\rho_0 \overline{D}_L}{\Delta x_i} - \left[\left(\frac{\delta_{p1}}{\Delta x_i} - \min\left(V_{a,f}, 0\right) \rho_{v2}\right) \frac{\partial p_v}{\partial u_2}\right]\right\} \left(\frac{\Delta y_1 + \Delta y_2}{2}\right)$$

where $\overline{D}_L$ is the average liquid diffusivity at node i and i+1 obtained from the array AMUX; δ_{p1} is the vapour permeability at node i obtained from the array VAPV; $V_{a,f}$ is the air velocity at the interface between the two cells, obtained from the array VELO; $\frac{\partial p_v}{\partial u_1}$ is the derivative computed by the function DP1U using temperature and moisture content at current iteration from UITER at node i; ρ_{v1} is the vapour density computed using temperature at node i at previous time step (from UOLD); Δx_1 is the distance between node i and i+1.

$$A_{w,T} = -\left(\frac{\lambda_1}{\Delta x_i} + \max(V_{a,f}, 0)\rho_a c_{pa}\right)\left(\frac{\Delta y_1 + \Delta y_2}{2}\right)$$

$$A_{e,T} = -\left(\frac{\lambda_1}{\Delta x_i} - \min(V_{a,f}, 0)\rho_a c_{pa}\right)\left(\frac{\Delta y_1 + \Delta y_2}{2}\right)$$

It should be noted that λ_1 is the harmonic mean of the thermal conductivity at nodes i and i+1.

Note: in the program, at the very last step of the computation of Aw in subroutine COEFST, Δx_{i+1} is used? I am also not sure why δ_{p1} is used for both coefficients, since it is a nodal rather than interface value.

Case cells i and i+1 are in different materials

- **Air/solid interface or solid/air interface: subroutine DQMDPV**

The subroutine DQMDPV is called to compute the coefficients C1, C2, C3, and C4. C1 and C2 are relative to moisture equation and are calculated, for the case of **air/solid interface** as:

$$C1 = \beta - \beta\left(\frac{\partial p_v}{\partial u}\right)_f \left(\frac{\beta}{D1}\right)$$

$$C2 = -\beta\left(\frac{\partial p_v}{\partial u}\right)_f \frac{\left(\frac{\rho_{o2} D_{L2}}{0.5\,\Delta x_i} + \frac{\delta_{p2}}{0.5\,\Delta x_i} + \rho_{v1}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_2}\right)}{D1}$$

in which:

$$\beta = \beta_e + \rho_{v1}\left|V_{a,f}\right|$$

$$D1 = \beta\left(\frac{\partial p_v}{\partial u}\right)_f + \left[\frac{\rho_{o2} D_{L2}}{0.5\,\Delta x_i} + \frac{\delta_{p2}}{0.5\,\Delta x_i} + \rho_{v1}\left|V_{a,f}\right|\left(\frac{\partial p_v}{\partial u}\right)_f\right]$$

$\left(\frac{\partial p_v}{\partial u}\right)_f$ means the derivative at the interface and it is obtained from the array DPV (computed in subroutine QMOLDI); $\frac{\partial p_v}{\partial u_2}$ is the derivative computed using temperature and moisture content at node i+1 in material 2; $\left|V_{a,f}\right|$ means the absolute value of the air velocity at the interface and is obtained from the array VELO; ρ_{o2} is the density of material 2; D_L is the liquid water diffusivity obtained from the array AMUX; β_e is the mass transfer coefficient obtained from AMUX; and ρ_v is the vapour density and is calculated using the air temperature.

For the case solid/air interface:

$$C1 = \beta\left(\frac{\partial p_v}{\partial u}\right)_f \frac{\left(\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_i} + \frac{\delta_{p1}}{0.5\,\Delta x_i} + \rho_{v2}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_1}\right)}{D2}$$

$$C2 = -\beta + \beta\left(\frac{\partial p_v}{\partial u}\right)_f \left(\frac{\beta}{D2}\right)$$

$$\beta = \beta_i + \rho_{v2}\left|V_{a,f}\right|$$

$$D2 = \beta\left(\frac{\partial p_v}{\partial u}\right)_f + \left[\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_i} + \frac{\delta_{p1}}{0.5\,\Delta x_i} + \rho_{v2}\left|V_{a,f}\right|\left(\frac{\partial p_v}{\partial u}\right)_f\right]$$

C3 and *C4* are relative to heat equation and are calculated, **in the case of air/solid** interface as:

$$C3 = \left[-\beta\left(\frac{\partial p_v}{\partial u}\right)_f\left(\frac{-\beta\frac{\partial p_v}{\partial T_S}A - \left[\left(\frac{\delta_{p2}}{0.5\,\Delta x_i} + \rho_{v1}\left|V_{a,f}\right|\right)A\right]}{D1}\right)\right] - \left[\left(\beta\frac{\partial p_v}{\partial T_S}\right)A\right]$$

$$C4 = \left[-\beta\left(\frac{\partial p_v}{\partial u}\right)_f\left(\frac{-\beta\frac{\partial p_v}{\partial T_S}B - \left[\left(\frac{\delta_{p2}}{0.5\,\Delta x_i} + \rho_{v1}\left|V_{a,f}\right|\right)B\right]}{D2}\right)\right] - \left[\left(\beta\frac{\partial p_v}{\partial T_S}\right)B\right]$$

where:

$$A = \frac{h_e + \rho_a c_{pa}\left|V_{a,f}\right|}{\left(h_e + \rho_a c_{pa}\left|V_{a,f}\right|\right) + \left(\frac{\lambda_2}{0.5\Delta x_i} + \rho_a c_{pa}\left|V_{a,f}\right|\right)}$$

$$B = \frac{\frac{\lambda_2}{0.5\Delta x_i} + \rho_a c_{pa}\left|V_{a,f}\right|}{\left(h_e + \rho_a c_{pa}\left|V_{a,f}\right|\right) + \left(\frac{\lambda_2}{0.5\Delta x_i} + \rho_a c_{pa}\left|V_{a,f}\right|\right)}$$

$\frac{\partial p_v}{\partial T_S}$ is the derivative at the interface; β, and D are the same as above for moisture equation; h_e is the equivalent heat transfer coefficient obtained from the array CNDX (array CNDX contains $he*\Delta x_{1/2}$); λ_2 if the thermal conductivity calculated by the subroutine COND using temperature and moisture content of at node i+1 in material 2.

For the case solid/air interface:

$$C3 = \left[\beta\left(\frac{\partial p_v}{\partial u}\right)_f\left(\frac{-\beta\frac{\partial p_v}{\partial T_S}B - \left[\left(\frac{\delta_{p1}}{0.5\,\Delta x_i} + \rho_{v2}\left|V_{a,f}\right|\right)B\right]}{D2}\right)\right] + \left[\left(\beta\frac{\partial p_v}{\partial T_S}\right)B\right]$$

$$C4 = \left[\beta\left(\frac{\partial p_v}{\partial u}\right)_f\left(\frac{-\beta\frac{\partial p_v}{\partial T_S}A - \left[\left(\frac{\delta_{p1}}{0.5\,\Delta x_i} + \rho_{v2}\left|V_{a,f}\right|\right)A\right]}{D2}\right)\right] + \left[\left(\beta\frac{\partial p_v}{\partial T_S}\right)A\right]$$

- **Solid/solid interface: subroutine DQMDU**

In the case of interface between cells belonging to different solid materials, the subroutine DQMDU is call to compute the coefficient *C1* and *C2* for moisture equation:

$$C1 = \left(\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_{c1}} + \frac{\delta_{p1}}{0.5\,\Delta x_{c1}} + \rho_{v1}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_1}\right)\left(1 - R_{pv}\frac{\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_{c1}} + \frac{\delta_{p1}}{0.5\,\Delta x_{c1}} + \rho_{v1}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_1}}{D}\right)$$

$$C2 = -\left(\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_{c1}} + \frac{\delta_{p1}}{0.5\,\Delta x_{c1}} + \rho_{v1}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_1}\right) R_{pv}\left(\frac{\frac{\rho_{o2} D_{L2}}{0.5\,\Delta x_{c2}} + \frac{\delta_{p2}}{0.5\,\Delta x_{c2}} + \rho_{v2}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_2}}{D}\right)$$

$$C3 = 0$$

$$C4 = 0$$

where:

$$D = \left(\frac{\rho_{o1} D_{L1}}{0.5\,\Delta x_{c1}} + \frac{\delta_{p1}}{0.5\,\Delta x_{c1}} + \rho_{v1}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_1}\right) R_{pv} + \left(\frac{\rho_{o2} D_{L2}}{0.5\,\Delta x_{c2}} + \frac{\delta_{p2}}{0.5\,\Delta x_{c2}} + \rho_{v2}\left|V_{a,f}\right|\frac{\partial p_v}{\partial u_2}\right)$$

$$R_{pv} = \frac{\frac{\partial p_v}{\partial u_2}}{\frac{\partial p_v}{\partial u_1}}$$

Δx_c is the cell size and is retrieved from array DXM, subscript 1 refers to value at node 1 (i) and 2 refers to value at node 2 (i+1). R_{pv}, the ratio between the derivatives, is obtained directly from the array KFAC which stores these ratios at solid/solid interfaces (in subroutine QMOLDI). $\frac{\partial p_v}{\partial u}$ at position 1 and 2 are computed by the function DP1U.

After computing *C1, C2, C3* and *C4*, A_{wu}, A_{eu}, A_{wT}, and A_{eT} are computed in COEFST as:

$$A_{wu} = \frac{-C1\,\Delta x_1}{\Delta x_1}\left(\frac{\Delta y_{i1} + \Delta y_2}{2}\right)$$

$$A_{eu} = \frac{C2\,\Delta x_1}{\Delta x_1}\left(\frac{\Delta y_1 + \Delta y_2}{2}\right)$$

$$A_{wT} = \left\{\frac{-\bar{\lambda}_h}{\Delta x_1} - \max\left(V_{a,f}, 0\right)\rho_a c_{pa} - l\,L_{v1}\,C3\,\Delta x_i\right\}\left(\frac{\Delta y_1 + \Delta y_2}{2}\right)$$

$$A_{eT} = \left\{\frac{-\bar{\lambda}_h}{\Delta x_1} + \min\left(V_{a,f}, 0\right)\rho_a c_{pa}\Delta x_1 + l\,L_{v1}\,C4\right\}\left(\frac{\Delta y_1 + \Delta y_2}{2}\right)$$

For air/solid interface:

$$\bar{\lambda}_h = \frac{\Delta x_{c1} + \Delta x_{c2}}{\frac{\Delta x_{c1}}{\Delta x_{1/2} h_e} + \frac{\Delta x_{c2}}{\lambda_2}}$$

For solid/solid interface:

$$\bar{\lambda}_h = \frac{\Delta x_{c1} + \Delta x_{c2}}{\frac{\Delta x_{c1}}{\lambda_1} + \frac{\Delta x_{c2}}{\lambda_2}}$$

For solid/air interface:

$$\bar{\lambda}_h = \frac{\Delta x_{c1} + \Delta x_{c2}}{\frac{\Delta x_{c1}}{\lambda_1} + \frac{\Delta x_{c2}}{\Delta x_{1/2} h_i}}$$

where Δx_c is the cell size in x direction obtained from the array DXM, Δx is the distance between nodes i and i+1 obtained from the array DX; the subscript 1 and 2 to refers to values at node i and i+1, respectively; h_e and h_i are the equivalent heat transfer coefficients for outdoor and indoor, respectively, obtained from array CNDX which contains the product $\Delta x_{1/2} h$; and λ is the thermal conductivity. l is equal to 1 when the logical variable *TUINTE*R is true. However, *TUINTER* is set to false in DUSTAR and remains unchanged so that l is always zero. Also, C3 and C4 are both nil. That means the connection between moisture and temperature fields through latent heat no longer exists in the LHS but still in effect in the RHS.

Subroutine DUSTAR

AAA(1,:,1)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	BBB(1,:,1)	$\begin{bmatrix} \boldsymbol{Aeu} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{AeT} \end{bmatrix}$	CCC(1,:,1)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$
AAA(2,:,1)		BBB(2,:,1)		CCC(2,:,1)	
AAA(1,:,2)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	BBB(1,:,2)	$\begin{bmatrix} \boldsymbol{Aeu} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{AeT} \end{bmatrix}$	CCC(1,:,2)	$\begin{bmatrix} \boldsymbol{Awu} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{AwT} \end{bmatrix}$
AAA(2,:,2)		BBB(2,:,2)		CCC(2,:,2)	
AAA(1,:,3)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	BBB(1,:,3)	$\begin{bmatrix} \boldsymbol{Aeu} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{AeT} \end{bmatrix}$	CCC(1,:,3)	$\begin{bmatrix} \boldsymbol{Awu} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{AwT} \end{bmatrix}$
AAA(2,:,3)		BBB(2,:,3)		CCC(2,:,3)	
AAA(1,:,4)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	BBB(1,:,4)	$\begin{bmatrix} \boldsymbol{Aeu} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{AeT} \end{bmatrix}$	CCC(1,:,4)	$\begin{bmatrix} \boldsymbol{Awu} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{AwT} \end{bmatrix}$
AAA(2,:,4)		BBB(2,:,4)		CCC(2,:,4)	
AAA(1,:,5)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	BBB(1,:,5)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$	CCC(1,:,5)	$\begin{bmatrix} \boldsymbol{Awu} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{AwT} \end{bmatrix}$
AAA(2,:,5)		BBB(2,:,5)		CCC(2,:,5)	

Contains of BBB are transferred to AU (moisture equation) and AT (heat equation) as follows: subscript u and T are removed for simplicity; column i contains parameters relative to node i.

AU(1,:,2)	0	0	0	0	0
AU(2,:,2)	0	0	0	0	0
AU(3,:,2)	0	0	0	0	0
AU(4,:,2)	0	0	0	0	0
AU(5,:,2)	0	0	0	0	0
AU(6,:,2)	**0**	$\boldsymbol{A_W}$	$\boldsymbol{A_W}$	$\boldsymbol{A_W}$	$\boldsymbol{A_W}$
AU(7,:,2)	$\boldsymbol{A_E}$	$\boldsymbol{A_E}$	$\boldsymbol{A_E}$	$\boldsymbol{A_E}$	$\boldsymbol{0}$
AU(8,:,2)	0	0	0	0	0
AU(9,:,2)	0	0	0	0	0
AU(10,:,2)	0	0	0	0	0
AT(1,:,2)	**0**	$\boldsymbol{A_W}$	$\boldsymbol{A_W}$	$\boldsymbol{A_W}$	$\boldsymbol{A_W}$

AT(2,:,2)	$\boldsymbol{A_E}$	$\boldsymbol{A_E}$	$\boldsymbol{A_E}$	$\boldsymbol{A_E}$	**0**
AT(3,:,2)	0	0	0	0	0
AT(4,:,2)	0	0	0	0	0
AT(5,:,2)	0	0	0	0	0
AT(6,:,2)	0	0	0	0	0
AT(7,:,2)	0	0	0	0	0
AT(8,:,2)	0	0	0	0	0
AT(9,:,2)	0	0	0	0	0
AT(10,:,2)	0	0	0	0	0

For j = 3 (top nodes)

Subroutine COEFST

A(1,:,1) A(2,:,1)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	B(1,:,1) B(2,:,1)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	C(1,:,1) C(2,:,1)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
A(1,:,2) A(2,:,2)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	B(1,:,2) B(2,:,2)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	C(1,:,2) C(2,:,2)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
A(1,:,3) A(2,:,3)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	B(1,:,3) B(2,:,3)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	C(1,:,3) C(2,:,3)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
A(1,:,4) A(2,:,4)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	B(1,:,4) B(2,:,4)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	C(1,:,4) C(2,:,4)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
A(1,:,5) A(2,:,5)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	B(1,:,5) B(2,:,5)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	C(1,:,5) C(2,:,5)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$

Subroutine DUSTAR

AAA(1,:,1)	$\begin{bmatrix}\mathbf{1} & \mathbf{0}\\\mathbf{0} & \mathbf{1}\end{bmatrix}$	BBB(1,:,1)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$	CCC(1,:,1)	$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$

AAA(2,:,1)		BBB(2,:,1)		CCC(2,:,1)	
AAA(1,:,2)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	BBB(1,:,2)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$	CCC(1,:,2)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$
AAA(2,:,2)		BBB(2,:,2)		CCC(2,:,2)	
AAA(1,:,3)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	BBB(1,:,3)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$	CCC(1,:,3)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$
AAA(2,:,3)		BBB(2,:,3)		CCC(2,:,3)	
AAA(1,:,4)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	BBB(1,:,4)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$	CCC(1,:,4)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$
AAA(2,:,4)		BBB(2,:,4)		CCC(2,:,4)	
AAA(1,:,5)	$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$	BBB(1,:,5)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$	CCC(1,:,5)	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$
AAA(2,:,5)		BBB(2,:,5)		CCC(2,:,5)	

AU(1,:,3)	0	0	0	0	0
AU(2,:,3)	0	0	0	0	0
AU(3,:,3)	0	0	0	0	0
AU(4,:,3)	0	0	0	0	0
AU(5,:,3)	0	0	0	0	0
AU(6,:,3)	**0**	**0**	**0**	**0**	**0**
AU(7,:,3)	**0**	**0**	**0**	**0**	**0**
AU(8,:,3)	0	0	0	0	0
AU(9,:,3)	0	0	0	0	0
AU(10,:,3)	0	0	0	0	0

AT(1,:,3)	**0**	**0**	**0**	**0**	**0**
AT(2,:,3)	**0**	**0**	**0**	**0**	**0**
AT(3,:,3)	0	0	0	0	0
AT(4,:,3)	0	0	0	0	0
AT(5,:,3)	0	0	0	0	0
AT(6,:,3)	0	0	0	0	0
AT(7,:,3)	0	0	0	0	0
AT(8,:,3)	0	0	0	0	0
AT(9,:,3)	0	0	0	0	0
AT(10,:,3)	0	0	0	0	0

Subroutine SOLVE

The subroutine SOLVE is called by DUSTAR to set the elements of AU(8, 1:5, 1:3), AU(9, 1:5, 1:3), AT(3, 1:5, 1:3) and AT(4, 1:5, 1:3) to zero. Its initial purpose is to compute the LHS coefficients for y-sweep, but because we are in 1D, the y direction will not be solved.

Matrix coefficients

The final matrix coefficients, a_E, a_W, and a_P, are calculated in DUSTAR using A_E and A_W *previously stored in AU and AT.*

Case internal nodes (j = 2)

For each internal node i (i=2, N1-1):

Moisture equation:

$$AU(1,i,j) = AU(6,i,j) * \frac{\Delta t}{\rho_o V_i} = a_W$$

$$AU(2,i,j) = AU(7,i,j) * \frac{\Delta t}{\rho_o V_i} = a_E$$

$$AU(5,i,j) = 1 - [AU(7,i-1,j) + AU(6,i+1,j)] * \frac{\Delta t}{\rho_o V_i} = a_P$$

Heat equation:

$$AT(6,i,j) = AT(1,i,j) * \frac{\Delta t}{\rho_o c\, V_i} = a_W$$

$$AT(7,i,j) = AT(2,i,j) * \frac{\Delta t}{\rho_o c\, V_i} = a_E$$

$$AT(10,i,j) = 1 - [AT(2,i-1,j) + AT(1,i+1,j)] * \frac{\Delta t}{\rho_o c\, V_i} + \frac{\rho_o u_i L_i}{\rho_o c}\frac{\partial f_l}{\partial T} = a_P$$

V_i is the volume of cell i computed as:

$$V_i = \left(\frac{1}{2}\Delta x_{i-1} + \frac{1}{2}\Delta x_i\right)\left(\frac{1}{2}\Delta y_{i-1} + \frac{1}{2}\Delta y_i\right)$$

$\rho_o c$ is the volumetric heat capacity, u_i is the moisture content at current iteration (from UITER), L_i is the latent heat of melting/thawing, f_l is fraction of liquid water.

Note: In computing the derivative $\frac{\partial f_l}{\partial T}$, T_i, the temperature at node i at current iteration, is used twice: $\frac{\partial f_l}{\partial T} = FLDER(T_i,\ T_i)$ which signifies the derivative of f_l against temperature computed by the function FLDER will always be zero. Not sure if it was the intention.

The arrays AU and AT are thus modified as follows (each column contains parameters for each node):

AU(1,:,2)	0	$\boldsymbol{a_W}$	$\boldsymbol{a_W}$	$\boldsymbol{a_W}$	0
AU(2,:,2)	0	$\boldsymbol{a_E}$	$\boldsymbol{a_E}$	$\boldsymbol{a_E}$	0
AU(3,:,2)	0	**0**	**0**	**0**	0
AU(4,:,2)	0	**0**	**0**	**0**	0
AU(5,:,2)	0	$\boldsymbol{a_P}$	$\boldsymbol{a_P}$	$\boldsymbol{a_P}$	0
AU(6,:,2)	0	*Aw*	*Aw*	*Aw*	*Aw*
AU(7,:,2)	*Ae*	*Ae*	*Ae*	*Ae*	*0*
AU(8,:,2)	0	0	0	0	0
AU(9,:,2)	0	0	0	0	0

AU(10,:,2)	0	0	0	0	0

AT(1,:,2)	0	AwT	AwT	AwT	AwT
AT(2,:,2)	AeT	AeT	AeT	AeT	0
AT(3,:,2)	0	0	0	0	0
AT(4,:,2)	0	0	0	0	0
AT(5,:,2)	0	0	0	0	0
AT(6,:,2)	0	$\boldsymbol{a_W}$	$\boldsymbol{a_W}$	$\boldsymbol{a_W}$	0
AT(7,:,2)	0	$\boldsymbol{a_E}$	$\boldsymbol{a_E}$	$\boldsymbol{a_E}$	0
AT(8,:,2)	0	**0**	**0**	**0**	0
AT(9,:,2)	0	**0**	**0**	**0**	0
AT(10,:,2)	0	$\boldsymbol{a_P}$	$\boldsymbol{a_P}$	$\boldsymbol{a_P}$	0

Case external nodes (j = 1 or j = 3)

<u>For j = 1:</u>

AU(1,:,1)	**0**	**0**	**0**	**0**	**0**
AU(2,:,1)	**0**	**0**	**0**	**0**	**0**
AU(3,:,1)	**0**	**0**	**0**	**0**	**0**
AU(4,:,1)	**0**	**0**	**0**	**0**	**0**
AU(5,:,1)	**1**	**1**	**1**	**1**	**1**
AU(6,:,1)	0	0	0	0	0
AU(7,:,1)	0	0	0	0	0

AU(8,:,1)	0	0	0	0	0
AU(9,:,1)	0	0	0	0	0
AU(10,:,1)	0	0	0	0	0

AT(1,:,1)	0	0	0	0	0
AT(2,:,1)	0	0	0	0	0
AT(3,:,1)	0	0	0	0	0
AT(4,:,1)	0	0	0	0	0
AT(5,:,1)	0	0	0	0	0
AT(6,:,1)	**0**	**0**	**0**	**0**	**0**
AT(7,:,1)	**0**	**0**	**0**	**0**	**0**
AT(8,:,1)	**0**	**0**	**0**	**0**	**0**
AT(9,:,1)	**0**	**0**	**0**	**0**	**0**
AT(10,:,1)	**1**	**1**	**1**	**1**	**1**

<u>For j = 2:</u>

AU(1,:,2)	**0**	0	0	0	**0**
AU(2,:,2)	**0**	0	0	0	**0**
AU(3,:,2)	**0**	0	0	0	**0**
AU(4,:,2)	**0**	0	0	0	**0**
AU(5,:,2)	**1**	0	0	0	**1**

AU(6,:,2)	0	0	0	0	0
AU(7,:,2)	0	0	0	0	0
AU(8,:,2)	0	0	0	0	0
AU(9,:,2)	0	0	0	0	0
AU(10,:,2)	0	0	0	0	0

AT(1,:,2)	0	0	0	0	0
AT(2,:,2)	0	0	0	0	0
AT(3,:,2)	0	0	0	0	0
AT(4,:,2)	0	0	0	0	0
AT(5,:,2)	0	0	0	0	0
AT(6,:,2)	**0**	0	0	0	**0**
AT(7,:,2)	**0**	0	0	0	**0**
AT(8,:,2)	**0**	0	0	0	**0**
AT(9,:,2)	**0**	0	0	0	**0**
AT(10,:,2)	**1**	0	0	0	**1**

<u>For j = 3:</u>

AU(1,:,3)	**0**	**0**	**0**	**0**	**0**
AU(2,:,3)	**0**	**0**	**0**	**0**	**0**
AU(3,:,3)	**0**	**0**	**0**	**0**	**0**

AU(4,:,3)	**0**	**0**	**0**	**0**	**0**
AU(5,:,3)	**1**	**1**	**1**	**1**	**1**
AU(6,:,3)	0	0	0	0	0
AU(7,:,3)	0	0	0	0	0
AU(8,:,3)	0	0	0	0	0
AU(9,:,3)	0	0	0	0	0
AU(10,:,3)	0	0	0	0	0

AT(1,:,3)	0	0	0	0	0
AT(2,:,3)	0	0	0	0	0
AT(3,:,3)	0	0	0	0	0
AT(4,:,3)	0	0	0	0	0
AT(5,:,3)	0	0	0	0	0
AT(6,:,3)	**0**	**0**	**0**	**0**	**0**
AT(7,:,3)	**0**	**0**	**0**	**0**	**0**
AT(8,:,3)	**0**	**0**	**0**	**0**	**0**
AT(9,:,3)	**0**	**0**	**0**	**0**	**0**
AT(10,:,3)	**1**	**1**	**1**	**1**	**1**

Solving for z: subroutine UMATSOL

DUSTAR called UMATSOL with the arrays AU, AT, and the RHS solve for z, which in the case of 1D is the final solution, Δu_i and ΔT_i at the current iteration i. They are stored in the array DU.

Relaxation of the solution

At the end of the iteration i, the solution arrays are updated as:

$$U_i = UITER_{i-1} + \Delta U_i$$

$$UITER_i = (1-\alpha)UITER_{i-1} + \alpha(UITER_{i-1} + \Delta U_i) = UITER_{i-1} + \alpha \Delta U_i$$

where α is the under-relaxation coefficient. The minimum is set to 0.2 for moisture equation and to 0.3 for heat equation and they varies with the time step and iteration number.

Checking the convergence of the solution

To check the convergence, the limits are set for Δu, ΔT and ΔRH:

$$\begin{cases} \Delta u_{max} = 0.001 \\ \Delta T_{max} = 0.1 \\ \Delta RH_{max} = 0.01 \end{cases}$$

As long as any of the incremental solution Δu or ΔT or ΔRH is greater than their respective maximum values and the number of iteration is less than the max iteration (set to 15 for the first two time steps and 21after), the iteration continue.

The incremental solutions are the maximum value among nodal incremental solutions:

$$\begin{cases} \Delta u = \max\{|u_i - u_{i-1}|_{n=2}, |u_i - u_{i-1}|_{n=3}, \dots, |u_i - u_{i-1}|_{n=n-1}\} \\ \Delta T = \max\{|T_i - T_{i-1}|_{n=2}, |T_i - T_{i-1}|_{n=3}, \dots, |T_i - T_{i-1}|_{n=n-1}\} \\ \Delta RH = \max\{|RH_i - RH_{i-1}|_{n=2}, |RH_i - RH_{i-1}|_{n=3}, \dots, |RH_i - RH_{i-1}|_{n=n-1}\} \end{cases}$$

where *n* is the number of nodes. *u* and *T* are obtained from the solution array U, and RH is derived from the sorption curve using u and T by the function RHSOR.

At each iteration, the new value of RH is computed and stored in the array RHU.

A1.17.9 Modification of time step

Within each initial time step, there is possibility, if the incremental solution is too large, to reduce the time step and then cycle within the initial time step using new time step until satisfactory solution is found. This is done in the main routine after call to DUSTAR when DU(1,1,1) is less than -9990.

DU(1,1,1) is set to -9999:

- in UMATSOL when the incremental solution during an iteration is large (> 5 for Δu, or >1000 for ΔT)
- or in DUSTAR if the maximum number of iterations is attained without convergence and Δu > 1, or ΔT > 2, and the number of steps (ILOP) of changing time steps il less or equal than two.

In case the time step need to be changed for, the new time step is computed as:

$$\Delta t_{short} = \Delta t_i \times \frac{0.1}{ILOP}$$

where Δt_{short} is the new time step, ILOP is the number of times within initial time step $\Delta t_i$where the time step has been shortened. Note that the maximum ILOP is 2. The maximum number of cycles using shortened time steps within the initial time step is n_{steps} = 10 * ILOP.

The default time step is 3600s. If during a particular time step n, there is a need to shorten the time step, for the first time (ILOP = 1), the new time step will be 360s and the number of time steps will be 10. If there is again the need within the same time step to reduce the time step again, the new time step will be 180s with a number of steps of 20.

A1.17.10 Update solutions: subroutine INITMOIS

The INITMOIS update solutions at each time step. There are two options:

1) In case there is no modification of time step

- Assign U to UOLD
- Assign U to UITER
- Update PVN and RHU

Note: Before moving to new time step, UOLD is assigned to UOOO in the main program.

2) In case there is modification of time step within a particular time step

- Assign UOOO to UOLD, U, and UITER
- Update PVN and RHU using u and T of the previous time step.

Basically, before start cycling within the current time step using shortened time step, reinitialize all arrays to their values at the beginning of the current time step.

A1.17.11 Estimation of mold risk: subroutine MOLD

The subroutine MOLD estimates mold growth risk. The index calculated is stores in the vector RMOLD(N1*N2). The model is the one developed by Hukka and Viitanen (1999) based on the work of Viitanen (1997). The algorithm used to assess mold risk at node i and at time step n is as follow, given the temperature T_i and the relative humidity RH_i at node i:

$$M_n(i) = min\{max[(M_{n-1}(i) + \Delta M_n \times \Delta t_d), \quad 0], \ 10\}$$

Where:

t_d: the continuous time duration (in days) where RH_i stays less than RH_c.

ΔR_n : incremental mold index at time step n

M_{n-1}: cumulative mold index up to previous time step

M_n: cumulative mold index up to time step n

Δt_d: time step (**days**)

RH_c is computed as:

$$\begin{cases} RH_c = 1 - 3.13T_i + 0.16{T_i}^2 - 0.00267{T_i}^3 \\ \quad if\ T\ \leq 0°C, \qquad RH_c = 1 \\ \quad if\ T\ > 20°C, \qquad RH_c = 0.8 \end{cases}$$

The formula to calculate the incremental risk ΔM_n depends on the temperature T_i, and relative humidity RH_i.

<u>If $RH_i \leq RH_c$ or if $Ti < 0$ or $Ti > 50$ (unfavorable conditions)</u>

$$\begin{cases} \Delta M_n = -0.032 \quad if\ t_d\ \leq 0.25 \\ \Delta M_n = -0.016 \quad if\ t_d\ > 1 \\ \Delta M_n = 0 \qquad if\ 0.25 < t_d \leq 1 \end{cases}$$

The cumulative time duration under dry conditions is stored in TDRY and is reinitialized to 0 when RH_i greater than RH_c.

<u>If $RH_i > RH_c$ and Ti between 0 and 50 (favorable conditions)</u>

$$\Delta M_n = \frac{R1 \times R2}{7t_m}$$

in which t_m is the response time (in days) needed for the initiation of mold growth, defined as:

$$T_m = e^{[-0.68ln(T_i) - 13.9ln(100*RH_i) + 0.14I_{sp} - 0.33I_{sq} + 66.02]}$$

and R1 and R2 are correction factors defined as:

$$\begin{cases} R1 = \dfrac{2}{\dfrac{T_v}{T_m} - 1} \quad if\ M_{n-1}\ > 1 \\ R1 = 1 \quad if \quad M_{n-1}\ < 1 \quad (no\ growth) \end{cases}$$

$$R2 = \max\left[0,\ \ 1 - e^{[2.3 \times \min(M_{n-1} - M_{max},\ \ 0)]}\right]$$

where I_{sp} is the wood species index (0 for pine – default value set in subroutine MOLD, and 1 for spruce), and I_{sq} is the index for the wood quality (1 for kiln dry surface - default value set in subroutine MOLD, and 0 for resawn kiln dry wood surface). There is no option to modify these values in the GUI. T_v (time needed for the first visual appearance of mold) and M_{max} (maximum mold index) are defined as:

$$T_v = e^{[-0.74ln(T_i) - 12.72ln(100*RH_i) + 0.06I_{sp} + 61.5]}$$

$$M_{max} = 1 + 9.4 \times \left(\frac{RH_c - RH_i}{RH_c - 1}\right) - 4.4 \times \left(\frac{RH_c - RH_i}{RH_c - 1}\right)^2$$

The subroutine MOLD compute also the array IWET(N1*N2) and IWETSAT(N1*N2) as:

$$\begin{cases} if\ T_i \geq 0°C\ and\ RH_i \geq 0.8,\ \ IWET_i = \sum_{n=1}^{n-1} IWET_i + 1 \\ if\ T_i \geq 0°C\ and\ RH_i \geq 1,\ \ IWETSAT_i = \sum_{n=1}^{n-1} IWETSAT_i + 1 \end{cases}$$

where n is the time step. IWET and IWETSAT are initialized to 0s at the beginning of the simulation.

A1.17.12 RHT analysis

RHT

The number of RHT analysis is read in the file Control.in, and stored in the variable RHT_NUMBER_ANALYSIS. For each analysis, the minimums RH and T are read, as well as the starting hours for analysis. Values are stored in RHT_MINIMUM_RH (1:RHT_NUMBER_ANALYSIS), RHT_MINIMUM_T (1:RHT_NUMBER_ANALYSIS), and RHT_STARTING_HOUR_FOR_SUM.

At each time step and for each analysis (RHT80, RHT90, etc.) and for each internal node I, RHT is calculated as:

$$RHT = (RH - RH_{min}) \times (T - T_{min}) \times \Delta t/3600$$

where Δt is the time step (s), RH and T are relative humidity and temperature of the node, RH_{min} and T_{min} are minimums relative humidity and temperature. For RHT80, RH_{min} = 80 and T_{min} = 0. These limits are set in the file Control.in. If T > T_{min} or RH < RH_{min}, RHT = 0. The cumulative sum of RHT is also calculated:

$$RHT_{sum} = \sum_{i=istart}^{n} RHT_i$$

where n is the number of steps, *istart* is the starting hour set in the file *Control.in* to start computing the RHT's sum. At the end of the simulation, RHT_{sum} is written in the file *rht_index.out*.

For each output cycle (set to 1 at the beginning of the main file or read in file *Control.in*), the sum and the average RHT are calculated and written in binary file:

$$RHT_{sumOutput} = \sum_{i=1}^{n_c} RHT_i$$

$$RHT_{sumOutputAverage} = \frac{\sum_{i=1}^{n_c} RHT_i}{n_c}$$

where n_c is the total number of steps per cycle. At the end of each cycle, $RHT_{sumOutput}$ is reinitialized to 0.

Time of wetness

The time of wetness for each analysis (RHT80, RHT95) and for each internal node i is computed following two steps:

7) Sum all the steps for which *RH(i)* > *RHmin* and *T(i)* > *Tmin,* i.e.:

   Start: n_w (i) = 0

   For each step n ≥ n_{start}:

   If *RH(i)* > *RHmin* and *T(i)* > *Tmin*

   n_w (i) = n_w (i) + 1

8) When the simulation is completed, compute the time of wetness for each analysis as:

$$t_w(i) = \frac{n_w(i)}{n - n_{start}} \times RHT_{sum}(i)$$

where t_w is the time of wetness, n is the total number of steps, n_{start} is the starting hour (step) for summing RHT. The time of wetness is stored in the array RHTT. RHTT is written in the file *time_of_wetness.out*.

Freeze/thaw cycles

Assuming *Tf* is the critical temperature for freezing, *Tt* if the temperature for thawing and *RHc* is the critical RH, one freeze/thaw cycle occurs when *T* drops below *Tf* and then rises above *Tt*. The algorithm to calculate the number of free/thaw cycles at node i is:

isFrozen = false

FreezeThawCycles = 0

If at step n-1, the water was not in a frozen state (isFrozen(i) = false)

If at step n, *RH* ≥ *RHc* and *T* < *Tf*

Set isFrozen(i) to true

Else (at time step n-1, isFrozen was true)

If at step n, *T* > *Tt*

FreezeThawCycles(i) = FreezeThawCycle(i) + 1

Set IsFrozen(i) to false

The freeze/thaw cycles is written in the file *freeze_thaw_cycles.out*.

ANNEX II
Example of input file

Input file for hygIRC Model

Version 1.2 (BETA) - May 1995

Number of Nodes

5 3

Input hygIRC filename

C:\Users\defom\Documents\hygIRC-NEW\New folder\TESTNODES\Project No. 1\Case No. 1\hygirc.in

Output Final Name

hygIRC.out

Mean Values Output for RH and T

Meanrh-t.out

Output for TECPLOT Graphics

hygIRC.tec

Temporary Screen Files

Screen.dat

Temperature Post-Processing Contour file

Output Temperature File

Temper.dat

Output Moisture Content File

Moiscon.dat

Output Relative Humidity File

Relhud.dat

TILT Angle (Inclination) Vertical=90 horizontal=0 ...etc

90

Azimuth (north=0, east=90, south=180)

0

Cut --Vertical=0 / Horizontal=1

0

K0 (Mass Operator) (FD=3)

3

KP (Convective Operatot) (FV=3 corrected)

3

Theta coefficient (time blending coefficient)

1.0d0

KG (Grid Flag) (Given Dx,Dy, KG=1)

1

Dx --> From Left to Right

0.002 0.002 0.002

Dy --> From Top to Bottom

1

layer definition- define how many lines fortran should omit

4

1

1 3 0 0 0.006

1

1 1 0 0 1

Numerical Simulation Parameters

Inumer=0 (Steady) 1 (General Transient) 2 (weather Datafile)

2

case 0: number of iterations - tolarance

0 0 0

case 1: timestep totaltime outputtime

1 1 0 1

case 2: date start end

"2 ",#1-01-01#,#2-01-01#

case 2: time start end

"2 " 1/1/2001 2/1/2001

case 2: yearhour start

2 0

case 2: timestep totaltime outputtime

2 1 8760 1

Moisture Potential: (IMOIST) (IMOIST=0 RH) (IMOIST=1 U)

1

InitialConditions (INC=0 Read from file)(INC=1 BELOW)

1

Intial Conditions file

initial.con

Initial Moisture Contents/RH

1 0.15 0.15 0.15

Initial Temperatures

1 20 20 20

Definition of boundary conditions for b-t-l-r

((IBOUND)) Bottom Top Left Right 0=external 1=internal 2=constant

2 2 0 1

Bottom

Top

Left

Exterior

Right

Interior

Heat Transfer Coefficient (0=constant 1=file)

Bottom

0 0 0

Top

0 0 0

Left

30

Right

8

Mass Transfer Coefficient (0=constant 1=file)

Bottom

7.400005E-40 7.400005E-40 7.400005E-40

Top

7.400005E-40 7.400005E-40 7.400005E-40

Left

2.1E-07

Right

5.9E-08

Absorptivity (0=constant 1=file)

Bottom

0 0 0

Top

0 0 0

Left

0.5

Right

0

Emissivity (0=constant 1=file)

Bottom

0 0 0

Top

0 0 0

Left

0.9

Right

0

Heat Flux (0=constant 1=file)

0 0 1 1

Bottom

0 0 0

Top

0 0 0

Left

Exterior

Right

Interior

Temperature (0=constant 1=file)

0 0 1 1

Bottom

0 0 0

Top

0 0 0

Left

Exterior

Right

Interior

Moisture Boundary Conditions

Potential for moisture boundary conditions (Imoisp=0 RH) (Impoisp=1 U)

0

Relative Humidity / Moisture Content (0=constant 1=file)

0 0 1 1

Bottom

0 0 0

Top

0 0 0

Left

Exterior

Right

Interior

Wind Effect (IWind=0 not used, 1=constant, 2=file)

0 0 2 0

Wind Speed

1

Wind Orientation

45

Bottom

Top

Left

Exterior

Right

Interior

Number of Material Blocks

1

Material Property (MATRIX)

0 -1 -1 -1 -1 -1

1 -1 1 1 1 -1

2 -1 -1 -1 -1 -1

Sky Effect (ISky=0 not used, 1=constant, 2=file)

0 0 2 0

Sky Temperature

0

Bottom

Top

Left

Exterior

Right

Rain Condition (Irain=0 not used, 1=constant, 2=file)

0 0 2 0

Rain Fluxes (bottom--top--left--right)

0 0 0 0

Bottom

Top

Left

Exterior

Right

Heat/Moisture Sources

C:\Users\defom\Documents\hygIRC-NEW\TEST2-PPTY-DATA\cond.dat

Heat Source

1 0 0 0

Moisture Source

1 0 0 0

Use Air Flow Module? (1=Yes)

1

Pressule Level Parameters

Neutral Pressure Level

-3

Prescribe Pressure Difference (0=No, 1=Yes)

1

Pressure Difference Value (Pa)

10

Include Stack Effect (0=NO, 1=Yes)

1

Include Wind Pressure Effect (0=NO, 1 Yes, eg 0 0 1 0)

0 0 1 0

Definition of Air FLow Boundary resistance for b-t-l-r

Bottom

1E+20 1E+20 1E+20

Top

1E+20 1E+20 1E+20

Left

1E-20

Right

1E-20

Definition of Boundary Type for b-t-l-r

Bottom

1 1 1

Top

1 1 1

Left

1

Right

1

Pressure/Velocities for b-t-l-r

Bottom

0 0 0

Top

0 0 0

Left

0

Right

0

ANNEX III
APPROXIMATE FACTORIZATION METHOD

In order to understand the implementation of moisture and heat equations, let develop general case of a diffusion-convection problem using the approximate factorization method.

Let us consider a general diffusion-convection problem described by the following equation:

$$c\frac{\partial\varphi}{\partial t}+\nabla.(k\nabla\varphi)+\nabla.(\rho V)+S=0 \tag{A.31}$$

Where C is the capacity, φ is a potential such as u or T, k the conductivity, V the motion speed and S a source term, which can be or not a function of φ. Integrating equation (A.31) over the time interval n and n+1 yields:

$$c\int_{n}^{n+1}\frac{\partial\varphi}{\partial t}dt+\int_{n}^{n+1}\nabla.(k\nabla\varphi+\rho V)dt+\int_{n}^{n+1}Sdt=0 \tag{A.32}$$

Using the general formulation of time discretization, i.e.:

$$\int_{t}^{t+\Delta t}fdt=\Delta t[\theta f^{t+\Delta t}+(1-\theta)f^{t}]$$

where θ is the time approximation weighting factor, Eq. A.32 can be rewritten in 2D as:

$$c(\varphi^{n+1}-\varphi^{n})+\Delta t\theta\left[\frac{\partial \mathrm{F}}{\partial x}+\frac{\partial \mathrm{G}}{\partial y}\right]^{n+1}+\Delta t(1-\theta)\left[\frac{\partial \mathrm{F}}{\partial x}+\frac{\partial \mathrm{G}}{\partial y}\right]^{n}+\Delta t\theta S^{n+1}$$

$$+(1-\theta)S^{n}=0 \tag{A.33}$$

where F and G are the flux terms expressed as: $F=\frac{\partial}{\partial x}(k\varphi+\rho V)$ and $G=\frac{\partial}{\partial y}(k\varphi+\rho V)$. φ(t) = φ^n = φ(nΔt). F, G and S are non-linear with respect to φ (e.g. k is a function of φ). They need to be approximated by linearization at time step n+1. Using the Newton-Raphson method (first order Taylor series), they can be approximated by:

$$F^{n+1}\approx F^{n}+\left(\frac{\partial F}{\partial\varphi}\right)^{n}(\varphi^{n+1}-\varphi^{n})=F^{n}+A_{x}^{n}(\varphi^{n+1}-\varphi^{n})$$

$$G^{n+1}\approx G^{n}+\left(\frac{\partial G}{\partial\varphi}\right)^{n}(\varphi^{n+1}-\varphi^{n})=G^{n}+A_{y}^{n}(\varphi^{n+1}-\varphi^{n}) \tag{A.34}$$

$$S^{n+1} = S^n + \left(\frac{\partial S}{\partial \varphi}\right)^n (\varphi^{n+1} - \varphi^n) = H^n(\varphi^{n+1} - \varphi^n)$$

where $A = {\partial F}/{\partial \varphi}$, $B = {\partial G}/{\partial \varphi}$ and $H = {\partial S}/{\partial \varphi}$

By setting $\Delta\varphi^{n+1} = \varphi^{n+1} - \varphi^n$ and substituting A.34 into A.33, we obtain:

$$c\Delta\varphi^{n+1} + \Delta t\theta \frac{\partial}{\partial x}[\mathrm{F}^n + A_x^n \Delta\varphi^{n+1}] + \Delta t\theta \frac{\partial}{\partial y}\left[\mathrm{G}^n + A_y^n \Delta\varphi^{n+1}\right] + \Delta t\left[\frac{\partial \mathrm{F}}{\partial x} + \frac{\partial \mathrm{G}}{\partial y}\right]^n$$
$$- \Delta t\theta \left[\frac{\partial \mathrm{F}}{\partial x} + \frac{\partial \mathrm{G}}{\partial y}\right]^n + \Delta t\theta H^n \Delta\varphi^{n+1} + \Delta t\theta S^n + \Delta t\, S^n - \Delta t\, \theta S^n = 0 \qquad \text{(A.35)}$$

Equation A.35 can be rewritten as:

$$\left[I + \frac{\theta\Delta t}{c}\frac{\partial}{\partial x}A_x^n + \frac{\theta\Delta t}{c}\frac{\partial}{\partial y}A_y^n + \frac{\theta\Delta t}{c}H^n\right]\Delta\varphi^{n+1} = -\frac{\Delta t}{c}\left[\frac{\partial \mathrm{F}}{\partial x} + \frac{\partial \mathrm{G}}{\partial y} + S\right]^n \qquad \text{(A.36)}$$

Eq. A.36 is the delta-formulation of the time discretization of Eq. A.31 at any time step where I is the identity matrix (1 in our case). As stated by Salonvaara and Karagiosis (1994), this incremental form increases the accuracy of the formulation, since the solution will be a progression of $\Delta\varphi$ instead of φ. The spatial variables are still continuous in Eq. A.36. The discretization of the spatial derivatives by any mean (finite-difference, finite-volume, etc.) would result in large matrices, difficult to solve. This difficulty has led to the so-called factored scheme (Warming and Beam 1978), which consists of splitting a multidimensional problem into a series of one dimensional ones, easy to solve. In fact, Eq. A.36 can be rewritten in the following form, assuming that the source term is independent of φ (H = 0):

$$\left[\left(I + \frac{\theta\Delta t}{c}\frac{\partial}{\partial x}A_x^n\right)\left(I + \frac{\theta\Delta t}{c}\frac{\partial}{\partial y}A_y^n\right) - \left(\frac{\theta\Delta t}{C}\right)^2 \frac{\partial}{\partial x}A_x^n \frac{\partial}{\partial y}A_y^n\right]\Delta\varphi^{n+1}$$
$$= -\frac{\Delta t}{c}\left[\frac{\partial \mathrm{F}}{\partial x} + \frac{\partial \mathrm{G}}{\partial y} + S\right]^n \qquad \text{(A.37)}$$

The cross-term (in red) can be neglected without any loss in the order of accuracy (Beam and Warming 1978) and equation A.37 reduced to:

$$\left[\left(I + \frac{\theta\Delta t}{c}\frac{\partial}{\partial x}A_x^n\right)\left(I + \frac{\theta\Delta t}{c}\frac{\partial}{\partial y}A_y^n\right)\right]\Delta\varphi^{n+1} = RHS \qquad \text{(A.38)}$$

RHS is the right hand side of the system. It is known from previous time step. Eq. A.38 is the approximate factored form of Eq. A.36, which is easier and cost effective to implement. By setting:

$$z = \left[I + \frac{\theta \Delta t}{c}\frac{\partial}{\partial y}A_y^n\right]\Delta\varphi^{n+1} \tag{A.39}$$

The first step is to solve the following one dimensional equation for z:

$$\left(I + \frac{\theta \Delta t}{c}\frac{\partial}{\partial x}A_x^n\right)z = RHS$$

Then, the second step is to solve Eq. A.39 for $\Delta\varphi^{n+1}$:

$$\left[I + \frac{\theta \Delta t}{c}\frac{\partial}{\partial y}A_y^n\right]\Delta\varphi^{n+1} = z$$

When the source term H (Eq. A.36) is nonzero, there are several alternatives to treat it (Shih and Chyu 1991). In hygIRC, method 1 given by Shih and Chyu (1991) is used:

$$\left[\left(I + \frac{\theta \Delta t}{c}\frac{\partial}{\partial x}A_x^n + \frac{\theta \Delta t}{c}H^n\right)\left(I + \frac{\theta \Delta t}{C}\frac{\partial}{\partial y}A_y^n\right)\right]\Delta\varphi^{n+1} = RHS \tag{A.40}$$

of global and some important variables

ype	Description/Contents	Storage	Comments
22)	Absorptivity for left and right side nodes	COMMON/PROP1/ in Common.for	Read in input file
11)	Absorptivity forn bottom and top	COMMON/PROP1/ in Common.for	Read in input file
1,N2)	Harmonic mean of air permeability at the faces of cells	Local in hygIRC	Set in subroutine GENAKV
N2)	Contains hm for l-r nodes and nodal liquid water diffusivities, DL, in xdir	COMMON/AMUS/ in common.for	Subroutine INPUT for hm and MaterP for DL
N2)	Contains hm for b-t nodes and nodal liquid water diffusivities, DL, in y dir	COMMON/AMUS/ in common.for	Subroutine INPUT for hm and MaterP for DL
	Wall inclination	COMMON/WIND/ in Common.for	Read in input file
	Wall orientation	COMMON/WIND/ in Common.for	Read in input file
22)	Air moisture or vapour pressure for left and right (IMOIS = 1, read from file) nodes	COMMON/THERMAL/ in Common.for	Read in input file or calculated using weather data
11)	Air moisture or vapour pressure for bottom and top nodes	COMMON/THERMAL/ in Common.for	Read in input file or calculated using weather data
22)	Mass transfer coefs for left and right side	COMMON/PROP1/ in Common.for	Read in input file but later modified to account for wind effects
11)	Mass transfer coefs for bottom and top nodes	COMMON/PROP1/ in Common.for	Read in input file but later modified to account for wind effects
N2)	Contains hc/(Dx/2) for l an r nodes and thermal conductivity in x dir for internal nodes	COMMON/CONDC2/ in common.for	Subroutine INPUT / used in many places
N2)	Contains hc/(Dx/2) for b an t nodes and thermal conductivity in y dir for internal nodes	COMMON/CONDC2/ in common.for	Subroutine INPUT
M)	specific heat capacity at dry state for materials	COMMON/RHOCP/ in common.for	Read from heat capacity file
	0= vertical wall (default); otherwise, horizontal wall	COMMON/WIND/ in Common.for	Read in input file
N2)	Latent heat of evaporation/condensation of water in x dir	COMMON/CHPHASE/ in hygIRC	Compute by MATERP as HDX
N2)	Latent heat of evaporation/condensation of water in y dir	COMMON/CHPHASE/ in hygIRC	Compute by MATERP as HDY
	Ventilation pressure	COMMON/RAIN/ in common.for	Read in input file

Variable name	Type	Description/Contents	Storage	Comments
		during each iteration	common.for	
DTIME	Real	Time step in s	COMMON/INPUT1/ in hygirc	Read in input file
DTP	Array(N11,N22)	Time step at each node, all equal Δt (3600 s by default)	COMMON/CONSTZ/ in common.for	
DTPRINT	Integer	Time step for output in s	COMMON/INPUT1/ in hygirc	Read in input file
DU	Array(2,N1,N2)	Incremental solutions: DU(1, :, :) contains moisture solution and DU(2, :, :) contains temperature		set in UMATSOL and used in DUSTAR
DWPOT	3D array(200, 2, 200)	Product of Dw and u at x and y dir as function of u	COMMON/DWP/ in hygIRC	Computed in subroutine CALCPOT
DX	Array(N1)	Distance between nodes in x dir	COMMON/GRID1/ in Common.for	Set in subroutine REALDXDY
DXM	Array(N1)	Cell size in x dir with DXM(1) = DXM(2) = first dx read and DXM(N1-1)= DXM(N1) = last dx read	COMMON/NEWGRID/ in common.for	Read in input file. DXM is later modified in subrpoutine CORRGRID depending on the layer thicknesses and nodal schemes
DY	Array(N2)	Distance between nodes in y dir	COMMON/GRID1/ in Common.for	Set in subroutine REALDXDY
DYM	Array(N2)	Cell size in y dir	COMMON/NEWGRID/ in common.for	Read in input file. DXM is later modified in subrpoutine CORRGRID depending on the layer thicknesses and nodal schemes
EMISSX	Array(2, N22)	Emissivity for left and right side nodes	COMMON/PROP1/ in Common.for	Read in input file
EMISSY	Array(2, N11)	Emissivity for bottom and top nodes	COMMON/PROP1/ in Common.for	Read in input file
FILEN	Array of strings (15)	Contains namse of various input and ouput files	Local var in main	Set in subroutine INPUT
FLAT	Real	Latitude	COMMON/WEAT/	Read in input file or in weather file
FLON	Real	Longitude	COMMON/WEAT/	Read in input file or in weather file
HDX	Array(N1,N2)	Latent heat of evaporation/condensation of water in x dir	Local in MATERP	Values transferred to DHX in hygIRC
HDY	Array(N1,N2)	Latent heat of evaporation/condensation of water in y dir	Local in MATERP	Values transferred to DHY in hygIRC
HTIM	Real	day time (o to 24)	Local in hygIRC	set in hygIRC, used by Solar and AFSUN
HTX	Array(2, N22)	Heat transfer coefs for left and right side nodes	COMMON/PROP1/ in Common.for	Read in input file but later modified to account for wind effects
HTY	Array(2, N11)	Heat transfer coefs for bottom and top nodes	COMMON/PROP1/ in Common.for	Read in input file but later modified to account for wind effects
IBC	Array (4,N1)	Boundary type for b-t-l-r (1 = pressure, 2 = velocity)	COMMON/RAIN/ in common.for	Read in input file. It is set to 1 automatically and there is no way to change in GUI.
IBCP	Array(N1, 4)	New index for boundary conditions, computed using IBOUND	COMMON/INPUT2/ in Common.for	Computed in subroutine input. Can also be read in a boundary condition file.

Variable name	Type	Description/Contents	Storage	Comments
IBOUND	Array(4)	Index boundary conditions for b-t-l-r (0 = external 1 = internal, 2 = constant)	COMMON/INPUT2/ in Common.for	Read in input file
ICON	Array(N11,N22,2)	Index for internal radiation condition in x and y dir	COMMON/INTRAD/ in hygIRC	Read from Rad.dat if exist, used in RIGHT to compute internal radiation
ICYCLE	Integer	Time step number		
IEM	Array(N11,N22,2)	Emissivities at x and y dir for internal nodes	COMMON/INTRAD/ in hygIRC	Read from Rad.dat if exist, used in RIGHT to compute internal radiation
IHFLX	Array(4)	Boundary heat flux index for bottom, top left and right (0 = constant, 1 = file: weather data or interior data)	COMMON/INPUT2/ in Common.for	Read in input file
IHOUR	Real	Current hour	Common/WEAT/ in common.for	Set in subroutine WEATH called byhygIRCp
IINC	Integer	Initial condition index (0 = read from file, 1 = read from input file)	COMMON/INPUT1/ in COMMON.for	Read in input file
IMAT	Array(N1, N2)	Index of materials	COMMON/MATNUM/ in Common.for	Read in input file
IMOIS	Array(4)	Index for boundary moisture potential (0 = constant, 1 = file)	COMMON/INPUT2/ in Common.for	Read in input file
IMOISP	Integer	Index moisture potential for moisture (0 = RH and 1 = U)	COMMON/INPUT1/ in COMMON.for	Read in input file
IMOIST	Integer	Index for moisture potential (0 = RH, 1 = u)	COMMON/INPUT1/ in COMMON.for	Read in input file
IPCNTRL	Integer	0 = don't use air flow module; 1 = use air flow module	COMMON/RAIN/ in Common.for	Read in input file. The default value is 1 and there is no option in GUI to modify
IPRESD	Integer	Indoor ventilation index (0 = NO; 1 = YES)	COMMON/RAIN/ in common.for	Read in input file
IRAIN	Array(4)	Index for rain effect (0 = not used, 1 = constant, 2 = file)	COMMON/RAIN/ in common.for	Read in input file
ISATCK	Integer	index for stack effect (0 = NO, 1 = YES)	COMMON/RAIN/ in common.for	Read in input file
ISKY	Array(4)	Index for sky temperature (0 = not used, 1 = constant, 2 = file)	COMMON/TSKY/ in input.for	Read in input file
ISNUMER	Integer	0 = steady, 1 = general transient, 2 = use wheather file	COMMON/INPUT1/ in hygirc	Read in input file
ITEMP	Array(4)	Index for air T for bottom, top, left and right (0 = constant, 1 = read from file)	COMMON/INPUT2/ in Common.for	Read in input file
IWIND	Array(4)	Index for wind effect for btlr (0 = don't use, 1 = constant, 2 = read file) heat and mass transfer coefficients	COMMON/INPUT2/ in Common.for	Read in input file. In case 1 or 2, wind speed is used to modify the transfer coefficients
IWINDPR	Array (4)	Wind pressure effect index for b-t-l-r (0 = NO, 1 = YES) for pressure difference across the wall	COMMON/RAIN/ in common.for	Read in input file
KFAC	Array(2,NMAX,NMAX)	Ratio dPv/du2 over dPv/du1 at the	COMMON/KF/	set in QMOLDI and used in DQMDU

Variable name	Type	Description/Contents	Storage	Comments
		solid/solid interface		
KG	Real	Grid flag (= 1 given Dx and Dy)	COMMON/INPUT1/ in Common.for	Read in input file
KO	Real	Mass operator ??	COMMON/INPUT1/ in Common.for	Read in input file
KP	Real	Convective operator ??	COMMON/INPUT1/ in Common.for	Read in input file
LAYERX	Array(0:NMAX)	Index position for different layers in x dir	COMMON/ILAYERD/	Subroutine CORRGRID in call by hygIRC
LAYERY	Array(0:NMAX)	Index position for different layers in x dir	COMMON/ILAYERD/	Subroutine CORRGRID in call by hygIRC
MATBLK	Integer	Number of material blocks. Fpr 1D, it is the number of layers. For 2D it may be different	COMMON/INPUT1/ in COMMON.for	Read in input file
MAXCYC	Integer	Maximum # of cycles (MAXTIME/DTIME)	Local in hygIRC	
MAXTIME	Real	Max simulation time in s	COMMON/INPUT1/ in hygirc	Read in input file
N1	Integer	Actual nb nodes in x dir (nb of nodes defined in GUI + 2)	COMMON/N1N2/ in Common.for	Read in input file
N11	constant = 200	Nb max nodes in x dir	Constant defined in Param.for	
N2	Integer	Actual nb nodes in y dir (nb of nodes defined in GUI + 2)	COMMON/N1N2/ in Common.for	Read in input file
N22	constant = 200	Nb max nodes in y dir	Constant defined in Param.for	
NITER	Integer	Number of iterations	COMMON/INPUT1/ in hygirc	Read in input file
NLAYERX	Integer	Nb of layers in x direction	COMMON/ILAYERD/ in hygIRC	Read in input file
NLAYERY	Integer	Nb of layers in y direction	COMMON/ILAYERD/ in hygIRC	Read in input file
NM	constant = 229	Number max of material?	Constant defined in Param.for	One can have 2 or more layers with the same material, at different position of the wall
NMAX	constant = 200	Nb max of nodes occuring in x or y directions	Constant defined in Param.for	
P	Array(N11, N22)	Pressure at nodal points	COMMON/RAIN/ in common.for	Boundary values read from input or calculated using wind, stack and ventilation pressure. Internal node values obtained from solution of pressure equation.
PARAMX	Array(nlayerx,3)	Contains layer parameters (nb of nodes, unknown var, stretching factor) in x dir	COMMON/RLAYERD/ in hygIRC	Read in input file. The maxsize is (20,3) set by DIMENSION in hygIRC
PARAMY	Array(nlayery,3)	Contains layer parameters (nb of nodes, unknown var, stretching factor) in y dir	COMMON/RLAYERD/ in hygIRC	Read in input file
PILVI	Real	Cloudiness index	Common/WEAT/ in common.for	Set in subroutine WEATH called byhygIRCp
PRES	Array (4, N1)	Flow boundary resistance for b-t-l-r nodes	COMMON/RAIN/ in common.for	Read in input file
PRESN	Integer	Distance above neutral pressure level	COMMON/RAIN/ in common.for	Read in input file. In the input file, the value is negative of the value in GUI
PVN	Array(N11,N22)	Nodal vapour pressure	COMMON/VAP2/	Initialize in hygIRC at 10^{-10} if IMAT = 99 (air?), updated at each time step in call of

Variable name	Type	Description/Contents	Storage	Comments
				subroutine MATERP
QFLX	Array(2,N22)	Radiation heat flux for left and right sides	COMMON/PROP1/ in Common.for	Read in input file or computed using radiation data from weather file
QFLY	Array(2,N11)	Radiation heat flux for bottom (line 1) and top (line 2) nodes	COMMON/PROP1/ in Common.for	Read in input file or computed using radiation data from weather file
QMOIX	Array(N1-1,N2-1)	Liquid moisture flux at the interface of cells in x-dir	Common/MTFLX/ in common.for	Set in subroutine QM called by DUSTAR
QMOIY	Array(N1-1,N2-1)	Liquid moisture flux at the interface of cells in y-dir	Common/MTFLX/ in common.for	Set in subroutine QM called by DUSTAR
QMOVX	Array(N1-1,N2-1)	Vapour flux at the interface of cells in x-dir	Common/MTFLX/ in common.for	Set in subroutine QM called by DUSTAR
QMOVY	Array(N1-1,N2-1)	Vapour flux at the interface of cells in y-dir	Common/MTFLX/ in common.for	Set in subroutine QM called by DUSTAR
QMVOLX	Array(N1-1,N2-1)	Vapour flux at cell interfaces in x-dir at previous time step	COMMON/QMLATE/ in DUSTAR	Referenced in DUSTAR, used in QM, DUSTAR, RIGHT
QMVOLY	Array(N1-1,N2-1)	Vapour flux at cell interfaces in x-dir at previous time step	COMMON/QMLATE/ in DUSTAR	set in DUSTAR, used in QM, DUSTAR, RIGHT
QRD	Array(N2)	Rundown rain at external boundary	COMMON/QDOWN/	used in QMOLDI
QSRC	Array(2,N1,N2)	Heat and moisture sources	COMMON/SRC/ in input.for	Read in input file (QSRC = QSRCI) or read from sourfe file data
QSRCI	Array(2,N1,N2)	Heat and mc sources flags per node. In GUI, index 0, 1 or 2 can be entered as constant for all node, per layer or per node.	COMMON/SRC/ in input.for	Read in input file. For constant, QSRC(i) = 0; for 1, QSRC(i) = -5005 and for 2, QSRC(i) = -6006 for node. Case 1 or 2, a source filename should be indicated
QTX	Array(N1-1,N2-1)	Heat flux at the cell interfaces in x-dir	COMMON/HTFLX/ in common.for	Set in subroutine QT called by DUSTAR
QTY	Array(N1-1,N2-1)	Heat flux at the cell interfaces in y-dir	COMMON/HTFLX/ in common.for	Set in subroutine QT called by DUSTAR
RADDIF	Real	Diffuse radiation	Common/WEAT/ in common.for	Set in subroutine WEATH called byhygIRCp
RADDIR	Real	Direct radiation = RADTOT-RADDIF	Local in hygIRC	
RADREF	Real	Reflected radiation	Common/WEAT/ in common.for	Set in subroutine WEATH called byhygIRCp
RADTOT	Real	Total radiation	Common/WEAT/ in common.for	Set in subroutine WEATH called byhygIRCp
RAIN	Real	Wind driven rain	local in hygIRC	hygIRC, used by sub. QM in QMOLDI.for
RAINFLX	Array(4)	Rain flux value for b-t-l-r in case constant rain	COMMON/RAIN/ in common.for	Read in input file or computed using weather data. For 1D case, only left side is considered.
RAINMM	Real	Rain	Common/WEAT/ in common.for	Set in subroutine WEATH called byhygIRCp
RAINX	Array(2, N2)	Rain fluxes at left and right boundary nodes	COMMON/WEAT/ in hygIRC	hygIRC
RAINY	Array(2, N2)	Rain fluxes at b and t boundary nodes	COMMON/WEAT/ in hygIRC	hygIRC
RH	Array(N1,N2)	Nodal relative humidity obtained from sorption curve using u	COMMON/HISTR/ in common.for	Calculated in subroutine UP

Variable name	Type	Description/Contents	Storage	Comments
RHLIM1	Real	indoor RH upper limit for 24 hours	COMMON/TINDOOR/ in sub. hygIRC	Read in file tindoor.bcf if it exists.
RHLIM2	Real	indoor RH lower limit for 24 hours	COMMON/TINDOOR/ in sub. hygIRC	Read in file tindoor.bcf if it exists
RHMAT	2D array (200, 200)	Materials' relative humidity (sorption)	COMMON/SMATR/ in hygIRC	Read from file interp.dat in hygIRC
RHO	Array(-1:NM)	density at dry state for materials	COMMON/RHOCP/ in common.for	Read from density file
RHU	Array(N11,N22)	Nodal relative humdity	COMMON/VAP2/	Initilaize in hygIRC at 10^{-10} if IMAT = 99 (air), updated at each time step
RMOLD	Array(N1,N2)	Mold index at each node	COMMON/CMOLD/ in hygIRC	Set in subroutine MODEL. Not used nor printed.
SF	Array(4,N11,N22)	Random value at each node for stochastic properties. If SF = 1, deterministic properties	COMMON/SEED/ in hygIRC	
SKYT	Array(4)	Sky T for b, t, l and r sides. Each is assigned the value of SKYTEMP	COMMON/TSKY2/ in input	Subroutine INPUT and WEATH in file full99w
SKYTEMP	Real	Sky temperature	COMMON/TSKY/ in input.for	Read in input file or computed using weather data
TEM	Array(N1,N2)	Nodal temperature	COMMON/HISTR/ in common.for	
TEMPX	Array(2,N22)	Air T for left and right nodes	COMMON/THERMAL/ in Common.for	Read in input file if constant or obtained from weather data
TEMPY	Array(2,N11)	Air T for bottom and top nodes	COMMON/THERMAL/ in Common.for	Read in input file if constant or obtained from weather data
TETOTU	Array(NM)	Total tem./Vol for each material	local in hygIRC	Calculated in subroutine TMASS
THETA	Real	Time blending coefficient??	COMMON/INPUT1/ in Common.for	Read in input file
THICKX	Array(nlayerx)	Contains layer thickness in x dir	COMMON/RLAYERD/ in hygIRC	Read in input file
THICKY	Array(nlayery)	Contains layer thickness in y dir	COMMON/RLAYERD/ in hygIRC	Read in input file
TINDAV	Real	< -900 if indoor file exists. Otherwise it does not exist.	COMMON/TINDOOR/ in sub. hygIRC	Read in file tindoor.bcf if it exists. A default value of -999 is set in subroutine input if file does not exist
TMAS	Array(NM)	Mass of water in each material	local in hygIRC	Calculated in subroutine MASS
total_moisture_content	Array(number of steps)	total moisture content of the wall at each time step	local in hygIRC	The default size is set to 17520 in runhygIRC (main)
TOTEMP	Real	Total temperature	local in hygIRC	Calculated in subroutine TMASS
TOTMAS	Real	Total mass of water in the wall	local in hygIRC	Calculated in subroutine MASS
TOTU	Array(NM)	Moisture content kg/m^3 for each material	local in hygIRC	Calculated in subroutine MASS
TSURF	Array(4,NMAX)	Surface T at l (3, :) , r(4, :), t (1, :), and b (2, :) boundary nodes	COMMON/TSURFAC/	Set in subroutine QT called by DUSTAR
TTEM	Array(NM)	total temp. in each material	local in hygIRC	Calculated in subroutine TMASS
TZO	Real	Timezone	COMMON/WEAT/	Read in input file or in weather file
U	Array(2, N1, N2)	u and T for each node, U in line (1, 2, …, N1-	COMMON/ING1/ in COMMON.for	

Variable name	Type	Description/Contents	Storage	Comments
		1) and T in line (2,2, …, N1-1) and for boundary conditions. For boundary nodes, u is replaced by pv.		
UHMAT	2D array (200, 200)	Materials' sorption moisture content	COMMON/SMATR/ in hygIRC	Read from file interp.dat in hygIRC
UITER	Array(2,N11,N22)	Storage of solution at current iteration	COMMON/UITER1/ in common.for	
UOLD	Array(2,N11,N22)	Storage of solution at time t-1	COMMON/ING1/ in common.for	
UOOO	Array(2,N11,N22)	Storage of solution at time t-1	COMMON/UOLOL/ in hygIRC	UOOO is useful in the case there is a reduction of time step within a particular time step . UOOO keeps the solution before the new cycling starts.
VAPV	Array(2, N11,N22)	Vapour permeability at nodal points	COMMON/VAP2/	Set in MaterP, used in various subroutine
VELO	3D array(2, N11, N22)	Velocity Vx and Vy at the cell interfaces	COMMON/AIRFLOW/ in Velo.f9	Initialize with 0s in hygIRC, calculated in DARCY
WINDIR	Real	wind direction	Common/WEAT/ in common.for	Set in subroutine WEATH called byhygIRCp
WINSPD	Real	wind speed	Common/WEAT/ in common.for	Set in subroutine WEATH called byhygIRCp
X	Array(N11)	Local in hygIRC computed by XYCAL, contains x positions for cell boundaries	Local in hygIRC	
XADD	vector(24)	indoor air moisture content at every hour of the day, read from indoor file if it exists	COMMON/TINDOOR/ in sub. hygIRC	Elements read in tindoor.bcf if it exists
XMAT1	2D Array (200, 200)	Moisture content (u) for air permeability	COMMON/APERC/ in hygIRC	Read from file aper.dat in hygIRC
XMAT2	2D array (200, 200)	Moisture content for thermal conductivity	COMMON/CONDC/ in hygIRC	Read from file cond.dat in hygIRC
XMAT3	2D array (200, 200)	u for density	COMMON/DENSC/ in hygIRC	Read from file dens.dat in hygIRC
XMAT4	2D array (200, 200)	u for moisture diffusivity	COMMON/DIFLC/ in hygIRC	Read from file difl.dat in hygIRC
XMAT5	2D array (200, 200)	u for heat capacity	COMMON/HCAPC/ in hygIRC	Read from file hcap.dat in hygIRC
XMAT6	2D array (200, 200)	u for vapour permeability	COMMON/VPERC/ in hygIRC	Read from file vper.dat in hygIRC
XMAT7	2D array (200, 200)	u for suction pressure	COMMON/SUCCION/ in hygIRC	Read from file psuc.dat in hygIRC
XV	Array(2, N1-1,N2-1)	Vapour flux from the interface to i+1 in mat2 in x-dir and y-dir	Local in DUSTAR	set in QMOLDI and used in DUSTAR
XVOLD	Array(2, N1-1,N2-1)	Vapour flux from interface to i+1 in mat2 in x-dir and y-dir and at previous time step	COMMON/QMLATE/ in DUSTAR	Referenced and used in DUSTAR, RIGHT, and QM
Y	Array(N22)	Local in hygIRC computed by XYCAL, contains y positions for cell boundaries	Local in hygIRC	
YERHOUR	Real	Starting hour when using weather file	COMMON/WIND/ in hygirc	Read in input file
YMAT1	3D array(200, 200, 2)	Air permeability in x and y dir	COMMON/APERC/ in hygIRC	Read from file aper.dat in hygIRC
YMAT2	3D array(200, 200, 2)	Thermal conductivity for x and y dir	COMMON/CONDC/ in hygIRC	Read from file cond.dat in hygIRC
YMAT3	2D array (200, 200)	Density values	COMMON/DENSC/ in hygIRC	Read from file dens.dat in hygIRC

Variable name	Type	Description/Contents	Storage	Comments
YMAT4	3D array(200, 2, 200)	Moisture diffusivity (Dw) in x and y dir	COMMON/DIFLC/ in hygIRC	Read from file difl.dat in hygIRC
YMAT5	2D array (200, 200)	Heat capacity	COMMON/HCAPC/ in hygIRC	Read from file hcap.dat in hygIRC
YMAT6	3D array(200, 200, 2)	Vapour permeability values	COMMON/VPERC/ in hygIRC	Read from file vper.dat in hygIRC
YMAT7	2D array (200, 200)	Suction (s) values	COMMON/SUCCION/ in hygIRC	Read from file psuc.dat in hygIRC
YMAT7B	2D array (200, 200)	Derivative du/ds	COMMON/SUCCION/ in hygIRC	Computed in subroutine INDER
YMAT7C	2D array (200, 200)	Derivative $d^2u/duds$ where s is suction pressure	COMMON/SUCCION/ in hygIRC	Computed in subroutine INDER