**Introductions for Using TST2M Model Software**

1. **Preparation for the using the software**

TST2M model software is used to simulate land surface temperature and near surface air temperature. To run it , we should installed python2.7 runtime environment, installed numpy, pandas, and matplotlib package. It was tested in win7 system.

1. **Software files introductions**

**modelMain.py** is the main program used to prepare for input data, controlling the model and output data for the model.

**TST2MClass.py** includes the class of TST2M Model, the surface temperature and near surface air temperature simulation procedures are implemented here.

**constAndSetting.py** includes the setting of constant and program running setting.

**dealfile.py** includes the basic functions to deal with files in the model.

**drawpic.py** includes the functions to draw the picture productions.

**dealSurfaceType.py** includes the functions to deal with the surface type.

**getH.py** includes the function to get the air block height.

**interpolater.py** includes the interpolation method.

**ReadAwsData.py** includes the class for read the auto weather station (AWS) data.

**readDiamondData.py** includes the function to read the ECMWF data which is saved as diamond four data type（one of CMA Micaps data file）

**ReadFileData.py** includes the based class for reading AWS file

**shortwaveDownwardCompute.py** includes the function to compute the downward short wave flux.

**computeQs.py** includes the methods to compute net wave radiation flux

1. **The parameter files introductions**

The parameter files are saved in the “conf” directory which is at the same level of the modelMain.py.

**land2011.txt** is the surface type data that derived from MODIS.

**mapData2ModelMapData.csv** is the reference table for surface type index from “land2011.txt” to the model surface type index, which is set by ‘physicalParam.csv’.

**listStationForOutput.txt** is the list of the stations that need to output serial data of the result.

**output-lucy.csv** is the anthropogenic heat table.

**physicalParam.csv** is the table of physical parameter for different surface types.

1. **Using of TST2MClass**

The model class is named TST2MModel and defined in TST2MClass.py. We use the model class as following step：

**Step 1: Create the model object**

**l = TST2MModel(’physicalParam.csv’)**

’physicalParam.csv’ is the table of physical parameter for different surface types.

**Step 2:set surface type area array**

**l.setSurfaceArea(surfAreaArr)**

surfAreaArr should be numpy array with shape=(l.latGridNum,l.lonGridNum,l.surfaceTypeNum),value is the proportion of each surface type area in the grid.

**step 3:set initiated time**

**l.setInitiateTime(dt)**

dt is the beginning time of simulation, dt is datetime type of python.

**step 4:set initiated air temperature and surface temperaure field**

**l.initiateMetData(ts0,ta0)**

ts0 and ta0 should be numpy float array with shape=(l.latGridNum,l.lonGridNum)

**step 5:set backgroud meteorological data field of each step**

**l.setBackgroundMetDataOfEachTime(u,v,td,p,cl,ch,tb,tUpLevel,mlb)**

meteorological data should be numpy float array with shape=(l.stepNum,l.latGridNum,l.lonGridNum). u,v is the wind u and wind v in 10m, td is the drew point in 2m, p is the sea level pressure, cl is low/middle cloud cover, tb is deep soil temperature, tUpLevel is air block up level temperature, mlb is air block height.

**step 6:run model**

**l.run()**

1. **the input and output of the model**

What the main program should do is setting the parameters, preparing the input data, creating and controlling the model object. We should prepare the following parameter: model run time parameters, surface physical parameters, the surface type percent of each grid etc. And we should prepare initiate surface temperature and near surface air temperature fields, and background meteorological data should obey the format that mention in Point 4. Here the example is given to show how to prepare the input data and what output data will get from the model.

**Input data：**

1. **model run time parameter, which is set in ‘constAndSetting.py’.**

The simulate range, for example,

FROMLON=120.7

FROMLAT=32

TOLON=122.2

TOLAT=30.5

The simulating range is 120.7~122.2°E，32~30.5°N

The grid number,

GRIDXNUM=101

GRIDYNUM=101

The simulating step

STEPNUM=25

anthropogenic heat(True) or not(False)

ADDQF=True

need to save grid data(True) or not(False)

SaveGridData = True

need to save serial data(True) or not(False)

SaveSerialData = True

need to save image(True) or not(False)

SaveImage = True

1. **Physical parameter setting. the setting file is ‘physicalParam.csv’ in ‘conf’ directory.**

In the parameter file, the first line is name of physical variable, the second is the explain of the variable, the third line is the unit of variable, and the following lines are the physical value of each surface type, each one surface type occupies one line. In the value lines, the first raw is surface type index which will corresponding to ‘mapData2ModelMapData.csv’ file’s model surface type index. We can add arbitrarily number types(lines) to the file, but the more types we set, the more time will be spend to run the model.

1. **Area proportion of each surface type in the grid.**

The file to deal with the surface type proportion is ‘dealSurfaceType.py’. In the example, the land2011.txt”is the MODIS land cover type, which has 17 types, while the model number of surface type is 6. The MODIS land cover type map to model surface type is set by ‘mapData2ModelMapData.csv’ file. In the file, each MODIS land cover type has one line, the first column is the type index, the following column is proportion of each model type correspond to the MODIS land cover type index, and the sum of them should equal 1. For example,

Line “13,0,0,0.17,0.58,0.25,0”, means when the grid in ‘land2011.txt’ index number is 13, the grid equivalent 17% of index 2, 58% of index 3, 25% of index 4 of model surface type.

Line “8,0,0,1,0,0,0” means when the grid in ‘land2011.txt’ index number is 8, the grid equivalent 100% of index 2 of model surface type.

If there is a more accurate underlying surface coverage type for a grid，we can adjust it by ’conf\ replaceMapPointSurType.csv’.In ‘replaceMapPointSurType.csv’, each line includes the grid index and the model surface type proportion.

For example, line ‘63,52,7,20,11,44,18,0’ means the type index 0 ~ 5 of the grid(i=63,j=52) is 7%,20%,11%,44%,18%,0% respectively, which will replace the data calculated from ‘land2011.txt’.

1. **initiate surface temperature and near surface air temperature field**

In the example the initiate field is get by interpolating the AWS data. The AWS data is scatter data with position and 2m air temperature and land surface temperature. We can input other data source as initiate data, and the data should be numpy array with shape of (GRIDYNUM,GRIDXNUM) .

1. **background meteorological data**

In the example, the background field data of low (in) high cloud, cloud cover, 10 meter height U, 10 meter height V, 1000hPa temperature, 2 meter height temperature, 2 meter height dew point temperature are get from the ECMWF data formatted as diamond 4 (one of CMA MICAPS Data type) by resampling to model grid point data. The hourly data is obtained by linear interpolation from 3 hours of source data. The deep soil layer temperature is get from 20cm deep soil AWS data. Those data is saved in ‘datasource’ directory。

**Output：**

The output includes three categories -grid data, timing serial data, and images . The output file is stored in the ‘result’ directory, each of category is stored in a directory according to the date of the simulation.

**1) grid data**: It includes the average sensible heat, latent heat, heat storage, net radiation, temperature and surface temperature of each grid. The data are saved in directory ‘result\grid’. If you don’t want output it, please set ‘SaveGridData = False’ in ‘constAndSetting.py’.

**2) time serial data**: They are the hourly average data of the specified position, including air temperature, surface temperature, net radiation flux, heat storage and latent heat, sensible heat, the downward shortwave radiation element, The specify the type of surface sites under different temperature, surface temperature, net radiation flux, heat storage and latent heat, sensible heat, surface temperature, the type of time in the lattice in the proportion of different type of surface in the specified position. The position need to be output can be set by the ‘listStationForOutput.txt’ file. The data are saved in directory ‘result\serial’.If you don’t want output it, please set ‘SaveSerialData = False’ in ‘constAndSetting.py’.

**3) images:** It can output hourly air temperature distribution, surface temperature distribution, sensible heat flux distribution, latent heat flux distribution, heat storage flux distribution, net radiant flux distribution etc. These files are saved in directory ‘result\img’. If you don’t want output it, please set ‘SaveImage = False’ in ‘constAndSetting.py’.