Matlab script for WRF-QUIC one-way coupling

Matthieu Renault

November 2018

Contents

1 Introduction

The WRF-QUIC-COUPLING script aims at generating the necessary input files for QUIC-URB v6.1 from WRF netCDF output. More precisely, it will extract and process wind components, roughness length, terrain topography and land-use categories.

This user guide will give details on which parameters must be defined and how functions are constructed. First, here is the global procedure to follow :

1. If not already done, generate a QUIC_URB_RELIEF.exe from Fortran code (it is a modified version accounting for topography).
2. Set necessary parameters (see next section) and run coupling script from WRF netCDF output.
3. Move generated project directory and executable in QUIC MCODE directory.
4. Run the executable QUIC_URB_RELIEF.exe.
5. Start QUIC-URB from Matlab (MCODE) and select the project of interest.
6. Wait until it generates MAT-files (for example: velocity.mat, containing the velocity components and magnitude at each grid point).
7. In order to use QUIC GUI for topography, one must replace cityplot.m (in MCODE directory) by its new version provided along with the script files.
8. Delete existing .mat files before running this simulation again, exit QUIC-URB and go back to step 2. QUIC will not generate new MAT-files if there are already any.

2 Simulation settings

Parameters of interest are defined in the first lines of the script.

1. Open Main.m.
2. Pick a project name.
3. Select the WRF file used for this simulation (netCDF format).
4. Choose at which minimum and maximum altitude WRF data will be extracted (in meters above ground level). It is very important to choose a minimum altitude above the canopy layer maximum height.
5. Select the maximum number of topography blocks allowed to construct the domain.
6. Select a number of WRF horizontal wind data points. To achieve an even distribution, the final number may be modified.
7. Select simulation time moments from WRF.
8. Enable or disable different flags (see below for more information).

Flags

1. **TerrainFlag** enables terrain generation. Can be used to compare the impact of adding topography.
2. **BdFlag** and **VegFlag** respectively enable buildings and vegetation addition.
3. **Z0Flag** indicates from which kind of data roughness length will be computed.
   (a) If set to 1, its value is directly extracted from WRF restart file. **Z0DataSource** must be defined as WRF restart file name.
   (b) When set to 2, land use categories in the WRF source file supply the information. Pay attention to the scheme used in WRF (MODIS WINTER and USGS LAND COVER are already implemented in the **RoughnessLengthFunc.m**. **Z0DataSource** must be set to WRF output file name (defined above in step 3).
   (c) If this flag is set to 3, roughness length will be constant all over the domain and the user can define its value in **Z0DataSource**.

3 Main functions

3.1 Reading WRF data - *ReadDomainInfo.m*

This function extracts the following variables :

- Horizontal dimensions nx and ny (vertical dimension will be computed later, see part).
- Time moments (in units provided by WRF).
- Horizontal resolution dx and dy (vertical resolution is set to 1 in this version).

It is possible to crop domain borders by adding a variable while calling the function. This variable should be an array with the following format:

$$\begin{matrix} Xstart & Xend \\ Ystart & Yend \end{matrix}$$

Finally, this function also extracts:

- Terrain topography.
- Wind data (more details in next subsection).
- Land-use categories.
- Roughness length.

3.2 Reading wind data - *WindFunc.m*

In a few steps, wind data vertical position, velocity and direction is computed:

- WRF wind data altitude (in meters) is obtained from geopotential height (dividing the sum of **PH** and **PHB** by the constant of gravitational acceleration $g = 9.81 m/s^2$).
- Wind components U and V components are extracted.

- These three variables are interpolated at each WRF cell center (simple arithmetic mean).
- Velocity magnitude is computed as $\sqrt{U^2 + V^2}$.
- Velocity direction is computed from U and V vector components.
- In the last step, velocity magnitude, direction and vertical coordinates are permuted to be recorded in format [row,col] = [ny,nx]. So are every other layers in this script (they are stored as [nx,ny] in WRF).

3.3 Smoothing domain - *Smooth.m*

If the total number of horizontal cells exceeds the limit defined in the first part with **MaxTerrainSize**, this function will generate a new domain respecting the constraint.

It is based on the matlab function **imresize**, which by default relies on a bicubic interpolation method (it is possible to change the method, see **imresize** other options: bilinear, nearest point, Lanczos-2 and Lanczos-3, which have not been tested yet).

- First, new horizontal dimensions and resolutions are computed.
- Terrain topography is interpolated to fit the new domain dimensions.
- Idem for the wind velocity magnitude, direction and altitude values.
- Idem for roughness length.
- Land-use categories are interpolated to the "nearest point" as we must not change their values while averaging them.

3.4 Minimizing cell number - *MinDomainHeight.m*

This small function aims at reducing the number of cells by lowering the minimum domain elevation to 0. For the same reason, it also sets every block below 50 cm to 0, recording the corresponding number of blocks in **NbTerrain**. Wind vertical coordinate is modified accordingly.

3.5 Selecting WRF data stations - *SetWRFdataPt.m*

If **MaxNbStat** is large enough, every WRF wind data points will be used. If this is not the case, a number of points close to **MaxNbStat** are distributed evenly across the domain. In order to do so:

- WRF wind data row and column (horizontal) indices are computed to assert a even distribution.
- These horizontal coordinates are associated to each stations.
- For each station, a vertical index is computed. It corresponds to wind data altitude contained between defined boundaries (**MinWRFAlt** and **MaxWRFAlt**).
- The corresponding vertical coordinates (in meters) are associated to each stations, as well as the number of these coordinates.
- Wind speed and direction at these altitudes are associated to each stations.
- Finally, maximal vertical coordinate is stored, it will be used to define the domain height (this value multiplied by 1.2).

3.6 Reading building information - *ReadBDInfo.m*

If the BDFlag is set on, it means you will have to provide the script with a list of building data.

As building information sources can be very different from one case to another, this function simply assumes that the necessary data is stored in a csv file "BdFile". It should consist in an array where each line code for a building, and each column contains the following:

- Geometry type number (between 1 and 6 for Rectangular, Elliptical, Rectangular Stadium, Elliptical Stadium, Polygon)
- Height (in meters AGL).
- X center coordinate
- Y center coordinate
- Length
- Width
- Rotation with respect to the North-South axis.
- Base wall thickness (in case of stadiums)
- Roof flag (in case of stadiums)

Note that the script have not been tested for polygon type buildings yet.

3.7 Reading vegetation information - *ReadVegInfo.m*

Before running the *ReadVegInfo.m* function, one has to define the following parameters (in Main.m) for each canopy type:

- Corresponding land-use category number (as set in WRF output).
- Attenuation coefficient (cf Cionco work, among others).
- Mean height.

VegFlag has to be turned on to process the canopy layer data.

4 Writing output

Last part of the code generates QUIC-URB input files.

The first ones are not of particular interest for QUIC-URB simulation (they may be used by other QUIC features:

- WritePROJ creates a ".proj" file with basic grid information.
- QU_fileoptions.inp contains some information about QUIC-URB output type.
- WriteINFO creates a ".info" file with various information, such as display options.
- QP_elevation.inp is left blank (technically speaking, a certain amount of blank spaces are written), and is just present to allow QUIC-URB code to run without problems.

- Idem for QU_landuse.inp.

The four other functions code for critical QUIC-URB input files:

- *WriteQU_BUILDINGS.m* generates QU_buildings.inp, which contains information about topology, buildings and vegetation.
- *WriteQU_SIMPARAMS.m* generates QU_simparams.inp, which contains the simulation dimensions, resolutions, time moments and flags.
- *WriteQU_METPARAMS.m* creates QU_metparams.inp, which indicates the number of measuring sites and maximum size of data points profiles.
- *WriteSENSOR.m* creates as many sensors.inp files as there are wind data points. Each of which includes wind data location, velocity magnitude and direction.

See QUIC's documentation for more details.
Simulation settings and WRF data are saved in a "*ProjectName*Data.mat" file. It can be used for validation or debugging.