Development of RANS boundary condition for AMR-Wind

Document all notes here.

1 RANS ABL grid stretching

The physical domain coordinates x and the computational domain ξ are related by the mapping function $x(\xi)$:

$$x(\xi) = x_{geom}(\xi)(1 - h(\xi)) + h(\xi)x_{const}(\xi)$$

where the blend function $h(\xi)$ is defined as

$$h(\xi, \xi_T, W) = \frac{1}{2}\left(1 + \tanh\left(\frac{\xi - \xi_T}{W}\right)\right)$$

and depends on the transition location ξ_T and width W.

$$\frac{dx}{d\xi} = (1 - h)x'_{geom} - h'x_{geom} + hx'_{const} + h'x_{const}$$

1.1 Geometry growth mapping function

The geometric growth function x_{geom} is defined as :

$$x_{geom}(\xi) = \frac{\Delta_0}{r}\frac{1 - r^{\xi+1}}{1 - r} - \frac{\Delta_0}{r}$$

With the first cell height Δ_0 and stretch rate $r > 1$ provided by the user. The inverse of this function is

$$\xi_{inv}(x) = \frac{\ln\left(1 - (x - \Delta_0/r)\frac{r(1-r)}{\Delta_0}\right)}{\ln(r)}$$

The derivative of this function is:

$$\frac{dx_{geom}}{d\xi} = \frac{\Delta_0}{r(r-1)}r^{\xi-1}\ln(r)$$

1.2 Constant cell size mapping function

$$x_{const} = s(\xi - \xi_0) + L_{match}$$

slope s, offset ξ_0, and match height L_{match}.
To set up the constants in x_{const}, there are two options:
1. We can compute the ξ that would correspond to L input by the user:

$$\xi_L = \text{int}\{x_{inv}(L, \delta_0, r)\}$$

2. We can compute the ξ that would correspond to Δ_{max} input by the user:

$$\xi_L = \text{floor}\left[\frac{\ln(\Delta_{max} r^2/\Delta_0)}{\ln r}\right] - 1$$

We'll set the offset ξ_0=ξ_L. Because ξ_L has been rounded to the nearest integer, we'll compute L_{match} at ξ_L

$$L_{match} = x_{geom}(\xi_L, \delta_0, r)$$

And the slope will be the cell size at ξ_L:

$$s = x_{geom}(\xi_L, \delta_0, r) - x_{geom}(\xi_L - 1, \delta_0, r)$$

The derivative of x_{const} is

$$\frac{dx_{const}}{d\xi} = s$$

2 RANS k-ϵ ABL boundary conditions

Implemented in `MOData.cpp` as `calc_phi_m_alinot()`:

$$\phi_m(\zeta) = \begin{cases} (1 - \beta_m \zeta)^{-1/4}, & \zeta < 0 \\ 1 + \gamma_m \zeta & \zeta > 0 \end{cases} \tag{1}$$

where $\zeta = z/L$ (see [1])
Implemented in `MOData.cpp` as `calc_phi_eps_alinot()`:

$$\phi_\epsilon(\zeta) = \begin{cases} 1 - \zeta & \zeta < 0 \\ \phi_m(\zeta) - \zeta & \zeta > 0 \end{cases} \tag{2}$$

The following calculations are implemented in `ShearStress.H`.
As `ShearStressAlinot.calc_mu()`

$$\mu_{t0}(z) = \frac{\rho \kappa u_* z}{\phi_m(\zeta)} \tag{3}$$

As `ShearStressAlinot.calc_eps()`

$$\epsilon_0(z) = \frac{u_*^3}{\kappa z}\phi_\epsilon(\zeta) \tag{4}$$

As `ShearStressAlinot.calc_tke()`

$$k_0(z) = \sqrt{\frac{\kappa u_* z \epsilon_0(z)}{C_\mu \phi_m}} \tag{5}$$

$$k_0(z) = \sqrt{\frac{\mu_{t0}(z)\epsilon_0(z)}{\rho C_\mu}} \tag{6}$$

As `ShearStressAlinot.calc_omega()`

$$\omega_0(z) = \frac{\epsilon_0(z)}{C_\mu k_0(z)} \tag{7}$$

References

[1] Alinot, C., and Masson, C. k-ϵ model for the atmospheric boundary layer under various thermal stratifications. *Journal of Solar Energy Engineering 127*, 4 (06 2005), 438–443.