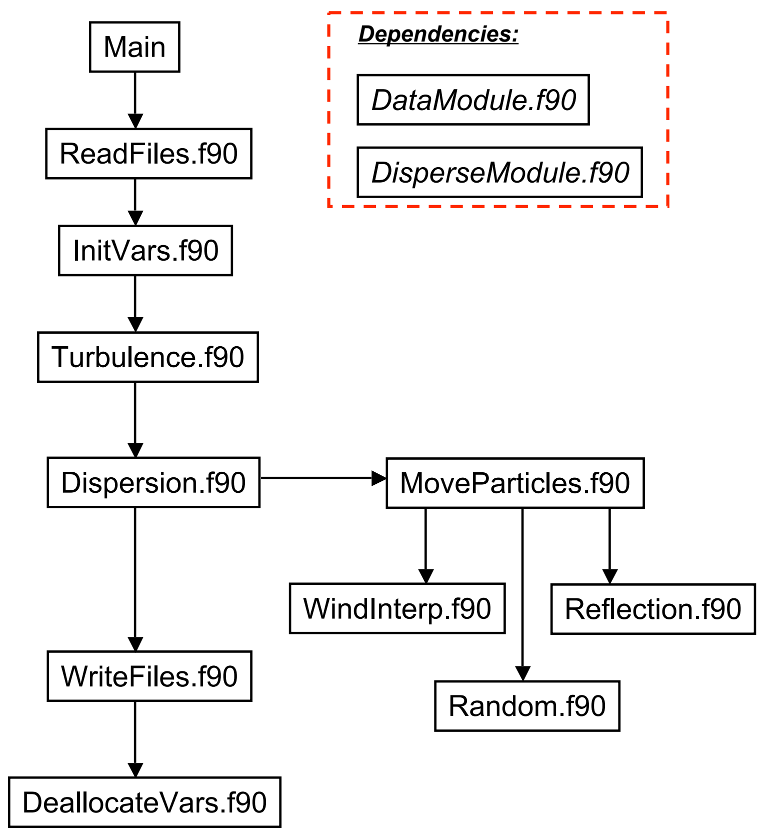
73 Term Urban Lagrangian Dispersion Model

- Equations and Background on Lag. model
- Assumptions for Equations – consider Eularian vs. Lagrangian quantities

- Term by term analysis –
  - 1st bulk/general
  - 2nd look at actual terms broken to see for example what different terms (ie. τ_{ij}'s mean)
- Turbulence Model



Breakdown of Each Subroutine

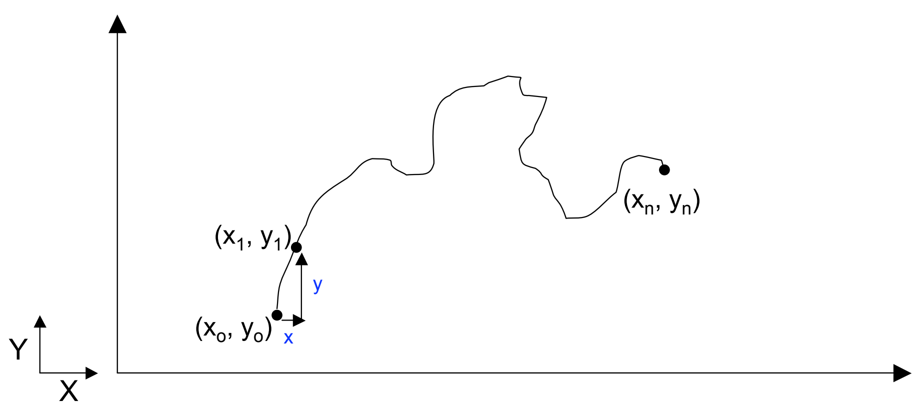
- Proposed Turbulence models??

Plan for validation

- Proposed Test Problems
- Comparison to QP
- Bug’s Experimental data

The following 2 slides will cover basic Lagrangian model description and equations (Its incomplete, I just want to know whether I am going in the right direction or not)

Lagrangian Particle Motion



❖Particles move under the influence of:

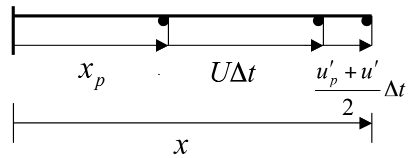
- ❑ Mean wind components ($\overline{U}, \overline{V}$ and $\overline{W}$).
- ❑ Fluctuating wind components (u', v' and w').
  - ➢Drift forcing due to turbulent stresses.
  - ➢Random forcing due to molecular bombardments.

Mathematical Formulation of Particle Translation

$$x = x_p + U\Delta t + \frac{u'_p + u'}{2}\Delta t$$

$$y = y_p + V\Delta t + \frac{v'_p + v'}{2}\Delta t$$

$$z = z_p + W\Delta t + \frac{w'_p + w'}{2}\Delta t$$



where *U,V* and *W* are mean interpolated velocities, Subscript '*p*' refers to "previous" positions, & Δt is the time step.

❖ The Fluctuating components of the velocities are calculated as:

$$u' = u'_p + du$$
$$v' = v'_p + dv$$
$$w' = w'_p + dw$$

[Williams, M.D. and M.J. Brown, (2002) "Description of the QUIC-PLUME model", Los Alamos National Laboratory report LA-UR-02-1246.).]

Basic Langevin equation for stochastic processes

$$du/dt = \underbrace{-a_1 u}_{\text{memory term}} + \underbrace{a_o}_{\text{Drift term}} + \underbrace{b\xi(t)}_{\text{Random forcing}} \quad \text{....(1)}$$

memory term [Damping effect associated with the viscous drag.]

Drift term [Accounts for the acceleration of particles at velocity gradients.]

Random forcing [Irregular and unsymmetrical molecular bombardment.]

Memory term is proportional to Lagrangian time scale (τ_L).

Drift term is proportional to the standard deviation of winds (σ_U, σ_V and σ_W).

Random forcing is proportional to the random pressure fluctuation.

Assumptions:

- Three dimensional stationary and inhomogeneous turbulence.
- High Reynold's number flow.
- Valid for inertial sub range only.
- Velocities of tracer and fluid are identical.

Langevin Equation in one-dimension

$$du = \underbrace{-\frac{u}{\tau_L}dt}_{\text{Memory term}} + \underbrace{\frac{1}{2}\left(1+\frac{u^2}{\sigma_u^{\ 2}}\right)\frac{\partial \sigma_u^{\ 2}}{\partial z}dt}_{\text{Drift term}} + \underbrace{\left(\frac{2\sigma_u^{\ 2}}{\tau_L}\right)^{1/2}\xi(t)dt}_{\text{Random forcing}} \quad \text{...... (2)}$$

Where,

u = Total wind speed in streamwise direction.

τ_L = Lagrangian time scale.

σ_u = Standard deviation of total wind in streamwise direction.

z = Vertical dimension.

$\xi(t)$ = Random function

dt = Time step

Eulerian Vs. Lagrangian quantities

$$du = -\underbrace{\frac{u}{\tau_L}dt}_{\text{Memory term}} + \underbrace{\frac{1}{2}\left(1+\frac{u^2}{\sigma_u^{\,2}}\right)\frac{\partial \sigma_u^{\,2}}{\partial z}dt}_{\text{Drift term}} + \underbrace{\left(\frac{2\sigma_u^{\,2}}{\tau_L}\right)^{1/2}\xi(t)dt}_{\text{Random forcing}}$$

…… **(2)**

Eulerian quantities:

❖Standard deviation of wind (σ_u).
❖Vertical coordinate (z).

Lagrangian quantities:

❖Total velocity (u).
❖Lagrangian time scale (τ_L).
❖Random function ($\xi(t)$).

In the following slides I added description of various subroutines.

ReadFiles.f90

❑Reads all the input files required for dispersion modeling:

❖Mean wind data from QUIC-URB ($\overline{U}, \overline{V}$ and $\overline{W}$)
❖Information regarding buildings in the domain.
❖Dispersion simulation parameters:
- ➢Release location.
- ➢Release type.
- ➢Number of particles released.
- ➢Collecting box dimensions.

❑**Called by following subroutine:**

❖Main.f90

InitVars.f90

- Initialize variables to their initial values.
- Set values for constants like π, von karman etc.

- **Called by following subroutine:**
  - Main.f90

Turbulence.f90

- Calculates turbulence parameters (Eularian reference) for the whole domain using zeroth order turbulence model.

- **Called by following subroutine:**
  - Main.f90

Dispersion.f90

❑Initialize particles from the source location.
❑Assign initial fluctuating component of velocity to particles based on turbulence parameters.
❑Estimates concentration based on averaging time.

❑**Called by following subroutine:**

❖Main.f90

❑**Calls the following subroutine:**

❖MoveParticles.f90 [For incrementing already released particles]

MoveParticles.f90

❑Estimates the turbulence stresses (τ_{ij}) from the standard deviation of winds (σ_U, σ_V and σ_W).
❑Estimates fluctuating component of winds by solving Langevin equations.
❑Estimates next position of particles from the previous position, interpolated mean wind, fluctuating wind components and time step.

❑**Called by following subroutine:**

❖Dispersion.f90

❑**Calls the following subroutines:**

❖WindInterp.f90 [For calculating interpolated mean wind]
❖Random.f90 [For calculating random forcing components of winds]
❖Reflection.f90 [For estimating surface reflection of particles]

Random.f90

Generates Gaussian (Normal) random number distribution for the random forcing component of Lagevin equations.

❑**Called by following subroutine:**

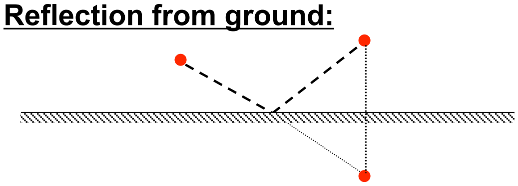
❖MoveParticles.f90

Reflection.f90

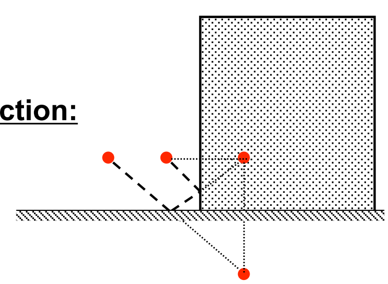
❑Accounts for reflection from all the surfaces encountered by the particle.

❑The approach to reflection is analogous to billiard ball type reflections.

Reflection from ground:



Corner reflection:

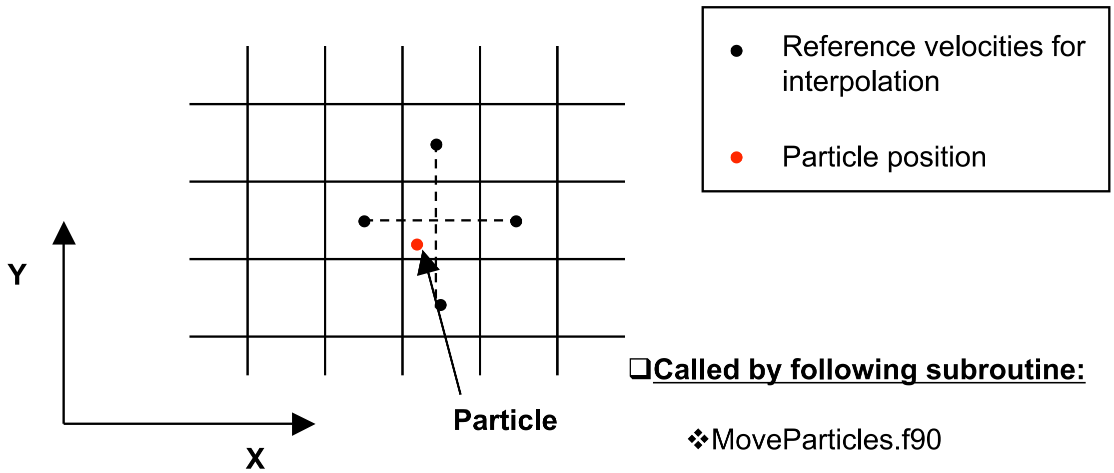


❑**Called by following subroutine:**

❖MoveParticles.f90

WindInterp.f90

❑Estimates velocity at particle position by interpolating between cell center velocities around the particle position.



❑**Called by following subroutine:**

❖MoveParticles.f90

WriteFiles.f90

❑Writes out all the output from the dispersion model.

❑Output includes:

❖Concentration field.

❖Turbulence field.

❑**Called by following subroutine:**

❖Main.f90

DeallocateVars.f90

Deallocate all the arrays to free the computer memory allocated for running the dispersion model.

- **Called by following subroutine:**
  - Main.f90