1. Algorithms’ description

Our code will work based on 3 core algorithms complying to the functional requirements of the specification document.

* 1. FR8 – Determine the quality of air at a given time and location

The goal is to determine the ATMO quality of air at a given time and location, based on the neighbouring’s sensors records. We will proceed by estimating the concentration of the 4 attributes (O3, SO2, NO2 and PM10) at the target location from all measurements made up to 24 hours before of after the target time.

We will then weight these measurements by their relative distance: the closer the sensor is, the more it will count. To proceed, we will take a random sensor and calculate its distance for the target location: it will be our referential and have a weight of 1. The weight of all other sensors will then be inversely proportionate to the distance from the target location: . If the referential is at 100m of the target location, a sensor located at 50m will have a weight of 2, and a sensor at 300m a weight of 1/3.

Finally, we determine the weighted average of measurements for each attribute from our determined weights. We can then apply the ATMO quality of air formula (<https://fr.wikipedia.org/wiki/Indice_de_qualit%C3%A9_de_l%27air>) to return the quality of air of the location at the specified time

algorithm FR8\_quality is

    inputs: Period timePeriod, Position askedPosition

    output: ATMOData predictedATMOScore

    call: FR8\_quality(Period timePeriod, Position askedPosition)

{

    // returns True if the timestamp is inside the period, False otherwise.

    function isGivenTimeInsideTimePeriod(Pediod period, Time timestamp) -> Boolean isInside

        if period.start <= timestamp and period.end >= timestamp

            return true

        else

            return false

    // returns the distance between 2 positions

    function distanceBetweenPositions(Position a, Position b) -> Float distance

    // get a random Sensor, the fastest to get actually

    function getASensor() -> Sensor sensor

    // convert

    function convertValuesAttributesToATMOCoefficient(Map<Attribute, Float> values) -> ATMOData predictedATMOScore

    // get a list of all measurements

    function getAllMeasurements() -> Measurement[] allMeasurements

    // returns predicted values for ATMO attributes for a given position and a considered period of time for the data

    function FR8\_qualityAttributes(Period timePeriod, Position askedPosition) -> Map<Attribute, Float> attributesPredictedValues

        var Attribute[] attributes := getAllAttributes() // an array of all data types (Attibutes)

        var Measurement[] allMeasurements := getAllMeasurements()

        var Map<Attribute, Float> numeratorSums := {

            attributes.O3: 0.0,

            attributes.NO3: 0.0,

            attributes.SO2: 0.0,

            attributes.PM10: 0.0

        }

        var Map<Attribute, Float> denominatorSums := {

            attributes.O3: 0.0,

            attributes.NO3: 0.0,

            attributes.SO2: 0.0,

            attributes.PM10: 0.0

        }

        var Sensor referentiel := getASensor()

        for each measurement in allMeasurements

            if measurement.getSensor().reliable and isGivenTimeInsideTimePeriod(timePeriod, measurement.timestamp)

                var Float coefficient := distanceBetweenPositions(referentiel.position, measurement.getSensor().position)

                numeratorSums[measurement.attribute] := numeratorSums[measurement.attribute] + (coefficient \* measurement.value)

                denominatorSums[measurement.attribute] := denominatorSums[measurement.attribute] + coefficient

        var Map<Attribute, Float> attributesPredictedValues := {

            attributes.O3: 0.0,

            attributes.NO3: 0.0,

            attributes.SO2: 0.0,

            attributes.PM10: 0.0

        }

        attributesPredictedValues := numeratorSum/denominatorSums

        return attributesPredictedValues

    // returns a ATMO coefficient for a given position and a considered period of time for the data

    function FR8\_quality(Period timePeriod, Position askedPosition) -> ATMOData predictedATMOScore

        var Map<Attribute, Float> results := FR8\_qualityAttributes

        return convertValuesAttributesToATMOCoefficient(results)

}

* 1. FR5 – Determine if a sensor is reliable

The goal is to determine if a private sensor is