**Windows instructions:**

1. **Install anaconda with python version >3.0:**

https://www.anaconda.com/distribution/#download-section

1. **Install Cantera from Anaconda Prompt:**

Open “Anaconda prompt” from the start menu, then type (and press enter):

conda install -c cantera cantera

1. **Make a new directory for your simulation. For example, simulationTest1.0**
   1. **Put example\_runfile.py in that directory.**
   2. **Put ceO2\_input\_reactions\_parameters.csv in that directory.**
   3. **Put ceO2\_input\_cti\_top\_info.cti in that directory.**
   4. **ceO2\_input\_simulation\_settings.py**
2. **Get the python modules ready.**
   1. **If you know how, put the following files in your python path. If you don’t know how to do that, just put them in simulationTest1.0.**
      1. **canteraKineticsParametersParser.py**
      2. **canteraSimulate.py**
3. **Run the file example\_runfile.py**
   1. **If you know how to use python, run the file as you normally would. If you don’t know how to run a python file, open Spyder from the start menu (or from Anaconda navigator). Open the .py file, click the play button or press f5, or choose “Run > Run”**
4. **The “Example 5” part has the example with the piecewise coverage dependence modifiers.**
   1. **As can be seen, some coverages and offsets must be provided.**
   2. **There is a helper function called descendingLinearEWithPiecewiseOffsetCheckOneReaction which returns true or false. The intent is that if it returns false, the parameter set is considered unrealistic and should be abandoned.**
      1. **That helper function is called repeatedly using: descendingLinearEWithPiecewiseOffsetCheckOneReactionAllReactions(reactions\_parameters\_array, piecewise\_coverage\_intervals\_all\_reactions, E\_offsets\_array\_all\_reactions, verbose = False).**
   3. **More details are at the end of this file on how to use the piecewise coverage dependence.**

**Making your own model:**

1. **First decide on a model name (in the example it was “ceO2”)**
2. **Create a file called model\_name+“\_cti\_top\_info.cti” like ceO2\_input\_cti\_top\_info.cti**
   1. **It is easiest to copy the example file.**
   2. **Phases Objects:**
      1. **inside the phases, declare the elements and species that may appear in that phase. If you want to make “unknown” species, and don’t care about their masses or heat capacities, you can make them out of elements like an inert gas like Ar, but the elements must be real. Don’t worry if you have real Ar in your simulation – the “unknown” species won’t convert to Ar unless you make a reaction allowing that.**
      2. **Define the site\_density in mol/m^2, unless you have defined the units as cm up above.**
   3. **Species Objects:**
      1. **Must have a name and atoms that they are made of (but can simply set them as being made of one Ar, for example).**
      2. **We are doing a kinetics example, so for each species we are leaving the thermo field as (). However, if your attempts at simulation later give errors due to needing a thermo object, you can use thermo=dummy\_thermo**
   4. **Reactions and Parameters File:**
      1. **When first making a mode for this modulel, you do not normally make the reactions in a cti file. You need to make a csv file like ceO2\_input\_reactions\_parameters.csv**
         1. **The easiest way is to open the existing file and edit it using the same syntax as is already in it. It is usually easiest to make everything irreversible with =>, but you can make things reversible with <=> (don’t make things reversible unless you understand what that means for a cantera simulation).**
         2. **Note: Alternatively, the module has a way to export reactions from an existing cti file into a csv file by the function extractReactionParametersFromFile which takes in an existing cti file and exports the parameters from it with this syntax (it also returns objects instead if somebody wants that, see python code): extractReactionParametersFromFile ("ceO2\_input\_cti\_full\_existing.cti", "ceO2\_input\_reactions\_parameters.csv")**
   5. **(Optional) Reaction Parameters Uncertainties File:**
      1. **Make a file like ceO2\_input\_reactions\_parameters\_errors.csv. Just copy the Reaction Parameters file and then replace numerical values with uncertainties. Each “error” value is an uncertainty, typically 1 standard deviation. This is useful for parameter estimation.**
   6. **Create/Update simulation\_settings.py file. For example, ceO2\_input\_simulation\_settings.py** 
      1. **This must have the initial conditions like surface coverages, which means that if you switch to a different initial species you might have to update that file.**

**How to use the Coverage dependence:**

The basic usage of the coverage dependence as demonstrated in example\_runfile.py

modifiers\_A in Example 5 has modifiers for the pre-exponential.

modifiers\_E in Example 5 has modifiers for the activation Energies.

For both A and E it is quite simple: it is basically just a lookup table with interpolation, and it is 1D.

Note that there are \*\*4\*\* arrays inside modifiers\_A. This is because there are \*\*4\*\*reactions , as definedin ceO2\_input\_reactions\_parameters.csv.

For each of these arrays inside modifiers\_A, there are 7 values. That is because piecewise\_coverage\_intervals has 7 values in this example (the number of coverages can be arbitrary-- 7 values was chosen with the intention of demonstrating that the number of coveragees chosen can be arbitrary).

In the case of A, we apply a coefficient of 10^modifiers\_array\_value (so it is like A0\*(10\*\*modifier). For E, we add the modifiers. That means that for "E", things are currently like E0 + 20000 etc. It is not by "gamma" right now.

canteraKineticsParametersParser.populatePiecewiseCoverageDependence populates simulation\_settings\_module.piecewise\_coverage\_dependences by species name, and (optionally) differently for each reaction based on species name. Note that a different piecewise number of concentrations/coverages can be used for each reaction. As seen inside canteraKineticsParametersParser.descendingLinearEWithPiecewiseOffsetCheckOneReactionAllReactions , #piecewise\_coverage\_intervals\_all\_reactions can either be 1D (defined the same for all reactions) or can be 2D (different intervals for each reaction)

Currently, if a species is listed as having concentration dependence for \*\*any\*\* reaction, that species currently must have a concentration dependence for all reactions. --> maybe in the future a blank list can mean no dependence. Currently, a "0" for "E" means no modification (because E0 + 0), and a "0" for "A" means no modification (because A0\*1)

Inside "getInterpolatedModifiersArray", the specific concentration/coverage at a given time in the simulation is used to assess the correct interval to use. Once the correct interval has been chosen, the interpolated\_modifiers\_array = slopes\_array\*species\_coverage+intercepts\_array

This gets used inside the modifiers\_array inside calculatePiecewiseCoverageDependentModifiedParametersArray

#This means that \*\*all\*\* reactions get modified at one time This uses piecewise\_kinetic\_parameter\_modifier\_arrays which are \*\*not\*\* coefficients yet. The piecewise\_kinetic\_parameter\_modifier\_arrays get calculated in calculatePiecewiseCoverageDependentModifiedParametersArray.

descendingLinearEWithPiecewiseOffsetCheckOneReaction and descendingLinearEWithPiecewiseOffsetCheckOneReactionAllReactions are designed to calculate \*\*all\*\* the rate constants for \*all\* of the coverage intervals according to Cantera’s coverage dependence formula in order to