Make a ‘library’ of the code. See WINDOW\_openMDAO/ README.md for instructions

There are 2 options to link modules:

- Connect (connect(c1.x, c2.x))

- Promote ( G.C1.x

G.x 🡨 promoted x to group level)

The ‘Connect’ option is preferred, because promotion needs the same variable name, plus the connect statements show connections explicitly.

Making variables known to functions:

1. from file import Thrust

2. def towerdesign(Thrust)

3. thrust = read\_csv(…)

Option 2 (passing variables) is preferred. Option 1 is/was often used, but then the parameter is a constant and cannot be changed by/inside the function. This means a lot has to be rewritten when a parameter that was previously a constant is changed into a (design) variable (e.g. rotor diameter could first have been fixed/not optimised, but later someone else might want to optimise it).

Option 3 is possible, but not fast, because it requires reading and writing files.

[From Tanuj – mail 10-10-2018]

**Coding Standards**

Here is the list of coding standards that we had discussed earlier. Feel free to add or modify items in the list.

1. Class and folder names in CamelCase. Function, variable and file names in snake\_case.
2. Use Docstrings to define every function and class.
3. Use comments to explain the logic in the code.
4. Provide **desc** and **units** to all OpenMDAO inputs and outputs. If metadata is used, use **desc** to describe it.
5. The abstract of every component should be in the **src** folder with its own folder named as Abs[ComponentName] and the file named as abs\_component\_name.
6. If a discipline has sub-systems, create sub-folders.
7. Only the **initialize** and **setup** functions should be parts of abstraction. The **compute** function should not be a part of abstraction.
8. The **initialize** ( or **\_\_init\_\_** )function should be used to declare the fixed parameters of a component. Using **import** to source the fixed parameters should be (generally) avoided.
9. [Discuss] If a function has a large number of inputs (maybe over 10), usage of data structure should be preferred over the usage of each input as an argument. – Conclusion: since this is about parameter passing inside a module, the developer may choose themselves what they prefer.