Previous Next

8.5.1 An IVP class hierarchy

MO2.2

The design and implementation of a Python class, or for that matter any computer software, in general does not have one perfect option. More often, there will be multiple options that are reasonably good. Our IVP class as currently implemented is reasonable, and now we will introduce another design. In this case, we will utilize the concept of *inheritance* in which a new class can be built upon an existing class.

In the case of our IVP, we will consider the two-level hierarchy shown in Figure 8.10. This hierarchy has the IVP parent class with skydiver, coffee, predator-prey, and neuron child classes. In addition to the parent and child class terminology, other common wordings include: base and derived class; superclass and subclass. We will use these pairings interchangeably.

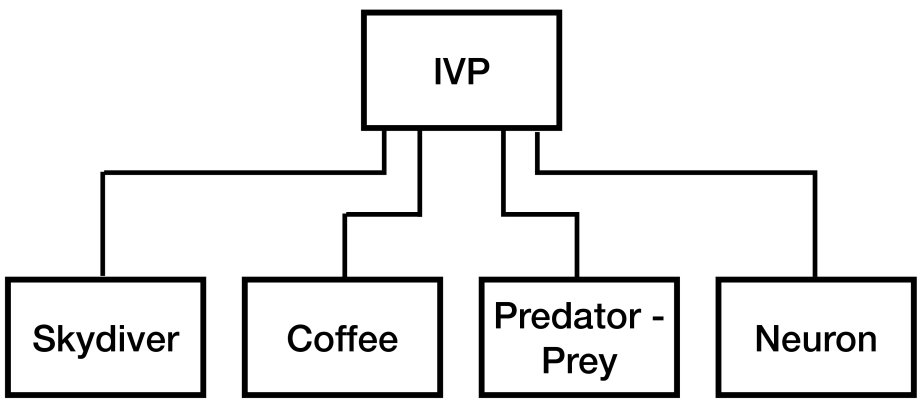

Figure 8.10: IVP two-level class hierarchy

For this current implementation of our IVP class hierarchy, all of the data attributes will be in the IVP superclass (no data will be stored in the subclasses). Further, the only method in the subclass will be `evalf`. Here's the `IVP.__init__` and `IVP.evalf` code for the IVP superclass implementation (the other methods have not changed and so are not included in the text):

```
class IVP():
    def __init__(self, uI, tI, tF, p):
        """
        Args:
            uI (float list): initial condition
of state.
            tI (float): initial time.
            tF (float): final time.
            p (dictionary): set of fixed
parameters.
        """

        self._uI = uI[:]
        self._tI = tI
        self._tF = tF
        self._p  = copy.deepcopy(p)
```

Discussions

All posts sorted by recent activity

```
############ virtual methods for use outside
of class ############

    def evalf(self,u,t):
        """
        Args:
            u (float list): current solution.
            t (float): current time.

        Returns:
            float list: f(u,t).
        """
        raise NotImplementedError("evalf is
not implemented for this object")
```

Summarizing the changes:

- The `__init__` method no longer passes the function reference (since `evalf` will now be in the subclass).

- The more interesting change is that we still have `evalf` in this superclass. However, we have implemented it to raise an exception if it were to be invoked. What will happen then is if a subclass does not implement its own `evalf` method, then this exception will be raised.

A method which is not implemented in the superclass but which must be implemented in the subclass is known as a *virtual method* (another common term is *e*

Previous Next

contains a virtual method is commonly referred to as an *abstract class*. The reason for saving a superclass

- there is no `__init__` method in the `coffeeIVP`. Thus, the initialization of a `coffeeIVP` object