Previous Next

10.3.1 Implementation and application

MO2.3 MO2.8

Next, we further evolve our IVP class hierarchy by introducing `LinSysIVP`, a new IVP subclass specifically for IVPs governed by linear systems. The resulting hierarchy will have the structure shown in Figure 10.1, in which IVPs modeled with linear system will be derived from the `LinSysIVP` class, which in turn is derived from the `IVP` base class.

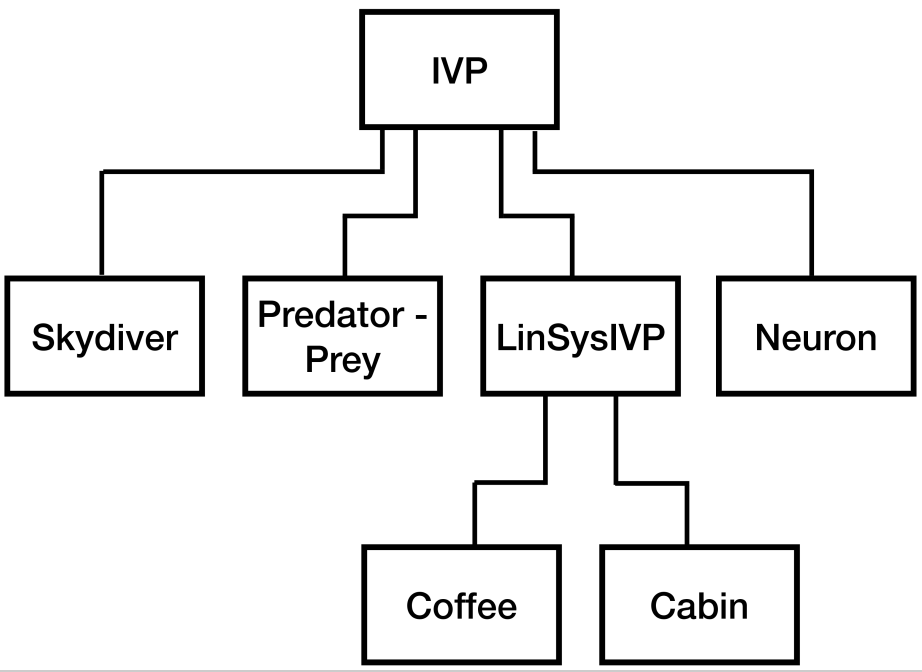

Figure 10.1: IVP class hierarchy including LinSysIVP

In the following video, we discuss the implementation of `LinSysIVP` and illustrate its use with the two-room cabin example described in Section 10.1.4. The following are key points of the video:

- As we've discussed before, a method implemented in a subclass "hides" the method of the same name in the superclass, which means that the superclass method will not be called when the method is invoked. For example, the `LinSysIVP.__init__` method hides `IVP.__init__`.
- `super()` is a builtin Python function which accesses an attribute of the superclass from which the subclass is derived. This function is used in `LinSysIVP.__init__` to execute the constructor of its `IVP` superclass, specifically through the call `super().__init__`
- The abstract function `evalf` of the IVP base class is now defined in `LinSysIVP`.
- `LinSysIVP` introduces two new abstract functions, specifically `LinSysIVP.calcA` and `LinSysIVP.evalb` which must be defined in its subclasses. An example of this is shown in the `Cabin2IVP` in `solve_cabin2_lin.py`

The Python codes discussed in this video are

Discussions

All posts sorted by recent activity

inheriting from IVP vs LinSysIVP Why are Skydiver, Neu
m_powers
3

[STAFF] transcript corrections That said, though, there
m_powers

Previous Next >