Class Internals

Robert Smallshire

COFOUNDER - SIXTY NORTH

@robsmallshire

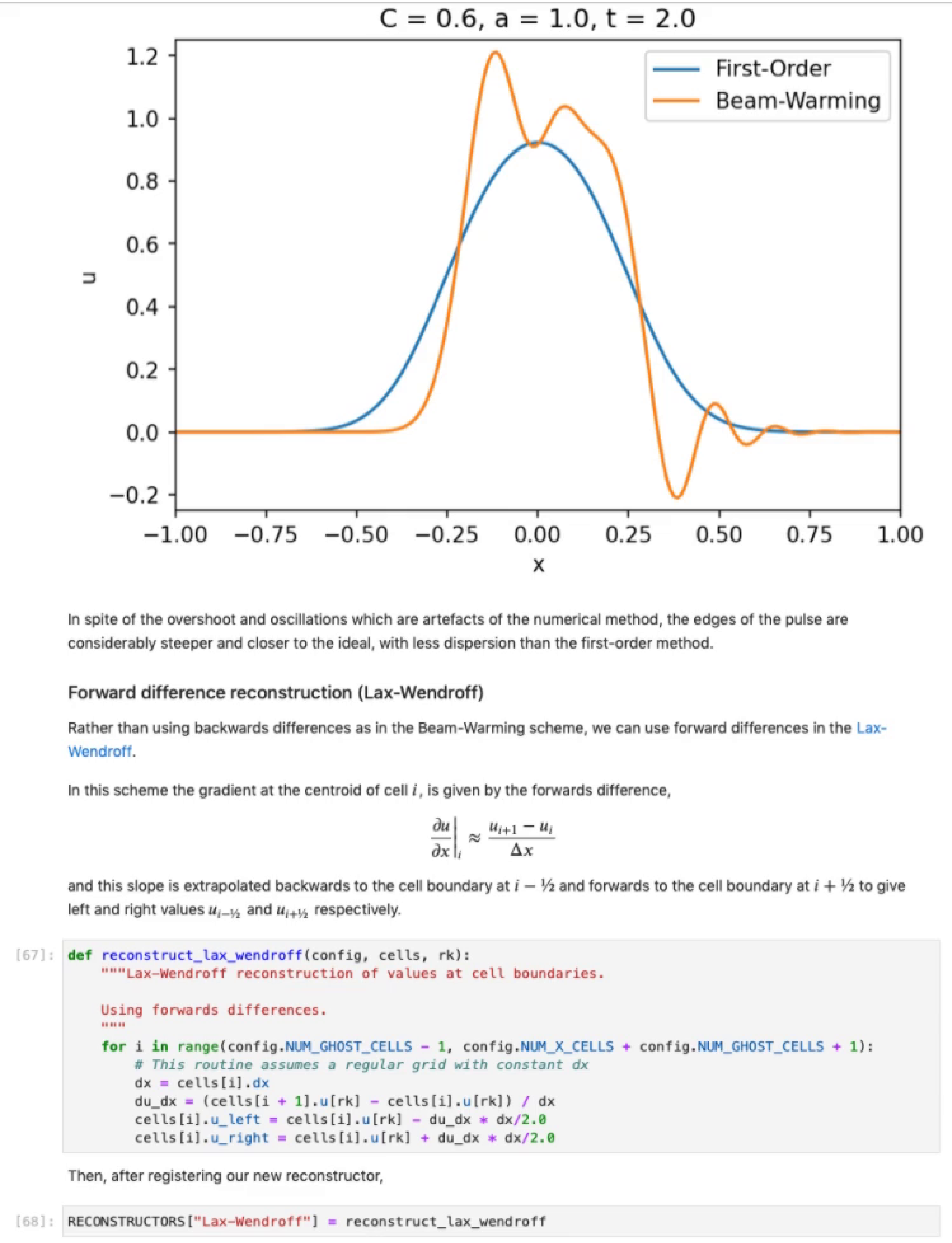
Austin Bingham

COFOUNDER - SIXTY NORTH

@austin_bingham

Class Attribute Lookup

```
>>> vars(type(v))["__repr__"](v)
'Vector(x=3, y=7)'
>>> v.__class__.__dict__['a_vector_class_attribute'] = 5
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'mappingproxy' object does not support item assignment
>>> setattr(v.__class__, 'a_vector_class_attribute', 5)
>>> v.__class__.__dict__
mappingproxy({'__module__': 'vector', '__doc__': 'An n-dimensional vecto
r.', '__init__': <function Vector.__init__ at 0x10eb66160>, '__getattr__
': <function Vector.__getattr__ at 0x10eb661f0>, '__setattr__': <functio
n Vector.__setattr__ at 0x10eb66280>, '__delattr__': <function Vector.__
delattr__ at 0x10eb66310>, '__repr__': <function Vector.__repr__ at 0x10
eb663a0>, '_args': <function Vector._args at 0x10eb66430>, '__dict__': <
attribute '__dict__' of 'Vector' objects>, '__weakref__': <attribute '__
weakref__' of 'Vector' objects>, 'a_vector_class_attribute': 5})
>>>
```



In spite of the overshoot and oscillations which are artefacts of the numerical method, the edges of the pulse are considerably steeper and closer to the ideal, with less dispersion than the first-order method.

Forward difference reconstruction (Lax-Wendroff)

Rather than using backwards differences as in the Beam-Warming scheme, we can use forward differences in the Lax-Wendroff.

In this scheme the gradient at the centroid of cell i, is given by the forwards difference,

$$\left.\frac{\partial u}{\partial x}\right|_i \approx \frac{u_{i+1} - u_i}{\Delta x}$$

and this slope is extrapolated backwards to the cell boundary at $i - ½$ and forwards to the cell boundary at $i + ½$ to give left and right values $u_{i-½}$ and $u_{i+½}$ respectively.

```
[67]: def reconstruct_lax_wendroff(config, cells, rk):
          """Lax-Wendroff reconstruction of values at cell boundaries.

          Using forwards differences.
          """
          for i in range(config.NUM_GHOST_CELLS - 1, config.NUM_X_CELLS + config.NUM_GHOST_CELLS + 1):
              # This routine assumes a regular grid with constant dx
              dx = cells[i].dx
              du_dx = (cells[i + 1].u[rk] - cells[i].u[rk]) / dx
              cells[i].u_left = cells[i].u[rk] - du_dx * dx/2.0
              cells[i].u_right = cells[i].u[rk] + du_dx * dx/2.0
```

Then, after registering our new reconstructor,

```
[68]: RECONSTRUCTORS["Lax-Wendroff"] = reconstruct_lax_wendroff
```

Dynamic class modification rarely needed

Incremental class definition

Useful in notebook environments

Facilitates literate programming

Attribute Lookup

Simplified Attribute Lookup Algorithm

```python
class object:
    """
    This code is illustrative. The real object class and its methods are implemented in C.
    """

    def __getattribute__(obj, name):
        cls = type(obj)
        if name in vars(obj):
            return vars(obj)[name]
        if hasattr(cls, name):
            return getattr(cls, name)
        if hasattr(cls, "__getattr__"):
            return cls.__getattr__(obj, name)
        raise AttributeError(f"{cls.__name__} object has no attribute {name}")
```