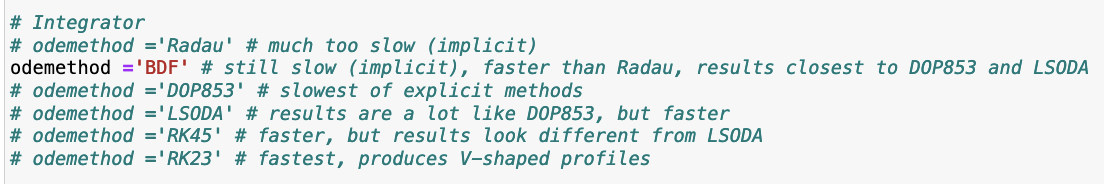
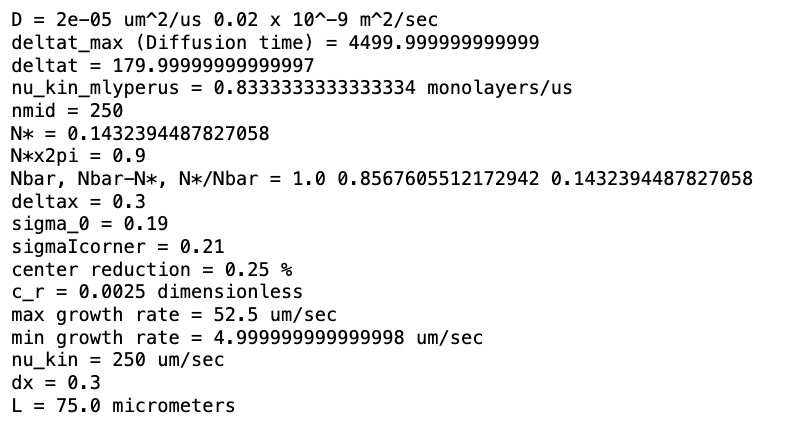
The notebook continuum\_model11\_2var.ipynb now uses Python’s scipy.integrate.solve\_ivp function, which gives access to a suite of algorithms. So, I’ve been comparing them. Here they’re ranked in order of most computationally expensive, to least:



The implicit methods are very slow – Radau doesn’t make any apparent progress at all, and BDF ran once, but other times it has crashed my laptop! The explicit methods readily produce faceted growth in the oscillating pattern we’ve been seeing. Figure 1 shows an example.

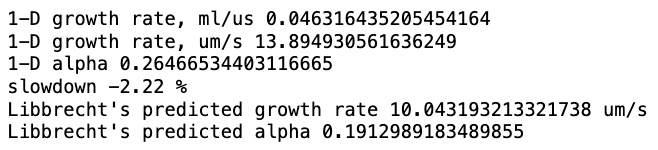
|  |  |
| --- | --- |
|  |  |
|  |  |
| **Figure 1**. Original (two-variable) formulation using LSODA (top panel) and DOP853 (bottom). | |

Other parameters for these runs were as follows:

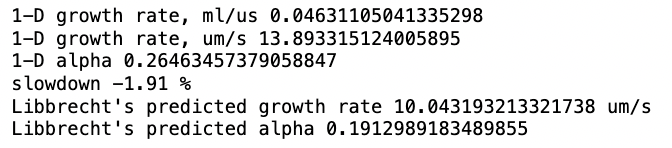


Both algorithms produced a diffusive slowdown of about 2%:

LSODA:

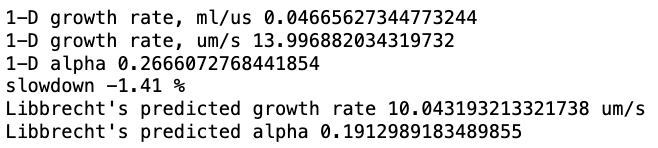


DOP853:



Also, I should note that the above runs were purposely assigned a low value of the diffusion coefficient, , so that a “V” shape would be more likely. When run at the value corresponding (I think) more closely to 260 K, , we get an even flatter steady state, as shown in Fig. 2.

|  |  |
| --- | --- |
|  |  |
| **Figure 2**. LSODA results with parameters corresponding to a temperature of 260 K. | |



And yet … when I compare LSODA 2-variable runs to LSODA 1-variable runs, I get a big difference:

|  |  |
| --- | --- |
|  |  |
|  |  |
| **Figure 3**. Two-variable formulation (top panel) and 1-variable formulation (bottom), both using LSODA. | |

The 1-variable formulation still seems to want to get to steady-state, eventually, but similarly to the behavior we saw previously, with odeint, it also encounters stability problems (Fig. 4).

|  |  |
| --- | --- |
|  |  |
| **Figure 4**. One-variable formulation using DOP853, with an enhanced diffusion coefficient. | |