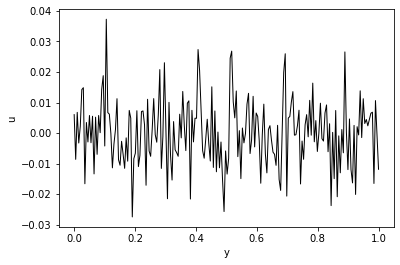
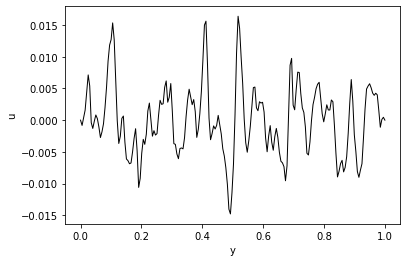
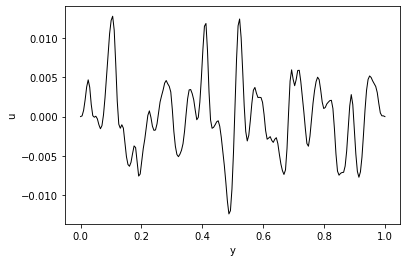
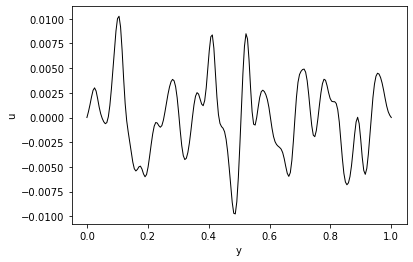
Use 1D Burgers’ equation with **diffusive** and **convective** terms on a domain with **Dirichlet** boundary condition

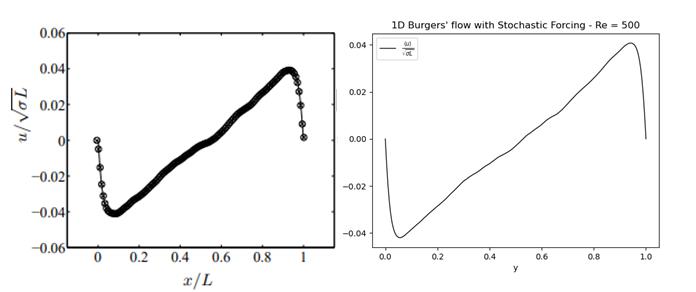


The 1D BDIM code just uses 2nd order finite differencing and is written in Python. Setting the kernel width and domain extension in the code to zero gives DNS and a stochastic forcing is given every , where a set of independent Gaussian variables -with variance - are used to give each grid point in the DNS a ‘kick’, producing a rough, noisy profile. The forcing timescale here is chosen so that the profile has almost decayed by the time the next forcing is given. This seems to give most similar results to the paper.



This profile is then evolved using the 1D Burgers’ equation e.g.:

where with Reynolds number 500 (used in Shoeybi), the diffusion effects are dominating; we could use higher Re to get more convective problems.

Guessing the forcing timescale and variance, the results from Shoeybi could be replicated quite closely.

Notes:

* The setup of this problem already gives us a set of random, rough profiles like we wanted, and the Burgers’ equation allows for some turbulence-like effects at higher Re, making it seem like a pretty ideal test case.
* It will probably be useful to adjust Re and the forcing timescale/variance to get the kind of problem that we want
* Possibly, the complete roughness generated by the independent stochastic forcing might be a problem (since it’s poorly resolved even in the DNS), and might hinder our comparisons later. Could use spatially correlated forcing to try and get around this potential problem.

The next steps:

* Show that this problem can be solved/well approximated by normal, O(2) BDIM with good resolution. This simply involves adding the domain extension and setting finite kernel width in my code.
* Need to choose how to represent the forcing in the BDIM simulations. I think it would make sense to apply the “BDIM operator” to the DNS forcing to get an equivalent forcing in the BDIM.
* Increase kernel width/down-sample the grid points to investigate the effects and increased errors in BDIM. Comparison can be made to the DNS for BDIM. Also, near-wall errors should be compared to the centre-channel errors, perhaps to show the effects of the BDIM wall in this.
* Attempt to calculate the “ImLES” version of the problem. We aim to calculate the wall-filtered field and so comparison should be made to the filtered DNS (as in Shoeybi where they used LES and compared against filtered DNS). **NB**: in Shoeybi they are using the same grid resolution for all cases but varying the LES filter width. This is where looking for models comes in. Hopefully, the terms from IP make some sense here. I will also keep in mind the extension/polynomial approach from Shoeybi which may prove useful.