Homework week 10

Problem 1 Consider a SDE:

$$dX_t = \mu(t, X_t)\, dt + \sigma(t, X_t)\, dB_t, \tag{0.1}$$

where B_t is standard Brownian motion. Show that the corresponding probability flow ODE is written as

$$dY_t = \left[\mu(t, Y_t) - \frac{1}{2}\frac{\partial}{\partial x}\sigma^2(t, Y_t) - \frac{\sigma^2(t, Y_t)}{2}\frac{\partial}{\partial x}\log\left(p(t, Y_t)\right)\right] dt. \tag{0.2}$$

Note that the Fokker-Planck equation for the corresponding SDE (0.1) is given as follows:

$$\partial_t p = -\partial_x(\mu p) + \partial_x^2\left(\frac{\sigma^2}{2}p\right). \tag{0.3}$$

Observe that

$$\partial_x^2\left(\frac{\sigma^2}{2}p\right) = \partial_x\left[\partial_x\left(\frac{\sigma^2}{2}\right)p + \frac{\sigma^2}{2}\partial_x\log(p)p\right]. \tag{0.4}$$

Then we find

$$\begin{aligned}\partial_t p &= -\partial_x\left(\mu p - \partial_x\left(\frac{\sigma^2}{2}\right)p - \frac{\sigma^2}{2}\partial_x\log(p)p\right)\\ &= -\partial_x\left(\left[\mu - \partial_x\left(\frac{\sigma^2}{2}\right) - \frac{\sigma^2}{2}\partial_x\log(p)\right]p\right).\end{aligned} \tag{0.5}$$

Therefore, we find the corresponding probability ODE as follows:

$$\begin{aligned}dY_t &= \tilde{\mu}(t, Y_t)dt\\ &= \left[\mu(t, Y_t) - \partial_x\left(\frac{\sigma^2(t, Y_t)}{2}\right) - \frac{\sigma^2(t, Y_t)}{2}\partial_x\log\left(p(t, Y_t)\right)\right]dt.\end{aligned} \tag{0.6}$$

Problem 2 The present question is motivated by the paper *On the approximation of functions by tanh neural networks*. To the best of our knowledge, there is no explicit and constructive neural-network schemes for approximating Sobolev functions $W^{k,p}$ with $k \geq 1$ and $1 \leq p < \infty$. In basic analysis, a first task is to understand function spaces and their regularity (e.g., smoothness classes). One then proceeds to more advanced topics, such as L^p spaces and convergence theory, where approximation by simpler or well-understood families plays a central role.

A natural question is given as follows:

Can a functions in L^p space, more generally $W^{k,p}$ function be approximated by a simple function (e.g. polynomials function) in a way that is numerically implementable?

From a theoretical perspective, the answer is affirmative: classical approximation results and density theorems guarantee such approximations under suitable norms. In numerical practice, however, we don't know the answer when $kp \leq n$. For instance, *On the approximation of functions by tanh neural networks* treats the class for the $W^{k,\infty}$ function. By Bramble-Hilbert Theorem, a function with k weak derivatives bounded in L^∞ space admits polynomial approximation of degree at most $k-1$.

We recall an embedding suitable for our discussion. Let $\Omega \subset \mathbb{R}^n$ be a bounded domain with smooth boundary. If $f \in W^{k,p}(\Omega)$ with $kp > n$, then

$$W^{k,p}(\Omega) \subset C^{k-[\frac{n}{p}]-1,\gamma}(\Omega) \tag{0.7}$$

for some $\gamma \in (0,1)$. In particular, if we consider $p = \infty$, then the $W^{k,\infty}$ function is just a C^{k-1} function, which is coincide with the Bramble-Hilbert Theorem. Thus, we believe that the method in *On the approximation of functions by tanh neural networks* still work when we consider the case $kp > n$ and $p < \infty$. However, for the case where $kp < n$, we didn't have the estimate (0.7). Therefore, we would like to fulfill this gap.

Note that the above is just a rough and simplified idea, and there is still a long way to go before implementation. Therefore, many things still require rigorous calculation.

Problem 3

Question