Assignment 2

(Dated: June 2, 2021)

In this problem set, we will begin to explore Bilby, which is a package that provides a consistent framework to run different types of MCMC and nested samplers on generic distributions. `Bilby` itself is not a sampler, but wraps popular samplers like `emcee` and `dynesty` with an additional layer to make the analysis as modular as possible. `Bilby` is not specific to GW science, but can be used to sample from any probability distribution with any of the supported samplers. To begin with, you will explore three tutorials (2 from `Bilby` website and 1 from Deep). You should download both examples, run them, and be familiar enough with the code that you could modify the likelihood functions and samplers.

1. The first tutorial fits a 1-D gaussian curve to a series of data drawn from a normal distribution. In this context, the two parameters of the model are the mean (μ) and standard deviation (σ) of the distribution. That is, our model is defined by the likelihood function

$$p(x|\mu,\sigma) = \frac{1}{\sqrt{\sigma^2 2\pi}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}, \tag{1}$$

   where x is our data. The analysis begins by defining a class for the likelihood of interest. This format is the standard way `Bilby` works: you define a class object with the specific members (like the `log_likelihood` function in this tutorial), and that provides enough abstraction for `Bilby` to pass this likelihood on to the sampler of choice. `Bilby` also allows you to set the priors for the parameters as well, which, for this tutorial, are uniform from $[0, 5]$ and $[0, 10]$ for μ and σ, respectively. The priors used in this tutorial are pre-made in `Bilby`, and are in the `bilby.core.prior` file, along with other common choices of priors. The priors can also be defined in a similar way as the likelihood, but custom priors will be addressed later. With the problem defined by the full posterior (prior + likelihood), the sampler can be run with `bilby.run_sampler` and the results are plotted in a corner plot by `result.plot_corner()`.

   First, read into the documentation of `Bilby` and `dynesty` to understand what the arguments of `run_sampler` do. Run the sampler for multiple different configurations and examine the output plot. Some additional modifications you can explore is to draw samples from a non-gaussian distribution for the data (say student-t) and fit a gaussian to it, or modify the priors such that they are not uniform.

2. The second tutorial walks you through performing linear regression on a set of data with noise of unknown variance. In this tutorial, the model assumes the data is linear (with 2 parameters, the slope and the intercept), and that the noise is gaussian, but with unknown variance (with 1 additional parameter, on top of the signal model). The likelihood function in this tutorial uses the predefined Gaussian likelihood class already included in `Bilby`, so you should read into that class and understand the arguments to `bilby.core.likelihood.GaussianLikelihood`. The priors are again uniform for the 3 parameters. Again, run the tutorial and understand each line of it. Then, modify the parameters, rerun the analysis, and examine the input.

3. Create a custom prior class, and use it for sampling.

   - The base prior module is `bilby.prior.Prior`.
   - You need to override at least two methods of this class: `prob` and `rescale`. The method `prob` is the normalized density function, $p(x)$. Ensure that the functions are vectorized i.e., they work with both scalars and numpy arrays. The `rescale` method is inverse of the cumulative distribution function, $P(x)$, i.e., it accepts $y \in [0, 1]$ and returns x such that $y = P(x)$.
   - Try out with a prior peaked at zero with an exponential decay $\sim \exp(-|x|)$ up to $|x| \leq a$, then 0.
   - Check sanity of your function by calling the `sample` method.
   - Go over the source code of the base prior class[1] to understand what all methods have been inherited. Try calling each of them.

[1] https://git.ligo.org/lscsoft/bilby/-/blob/master/bilby/core/prior/base.py