Homework # 3

Learning Goal

The goal of this third exercise is to understand and implement three different generic inference algorithms: importance sampling (IS), Metropolis Hastings within Gibbs (MH Gibbs), and Hamiltonian Monte Carlo (HMC).

Task

Write IS, MH Gibbs, and HMC samplers that consume the output produced by the Daphne compiler and the evaluators you wrote in completing HW 2. These samplers should generate a sequence of (weighted, equally for MH Gibbs and HMC) samples from the posterior of the program-denoted model. Using this sequence of (weighted) samples you will be asked to compute posterior expectations and variances. It is strongly suggested that you separate concerns and code an evaluator that produces a stream of samples that is then subsequently consumed by separate plotting and expectation computing programs or workbooks.

It is up to you which evaluator you use when implementing these inference engines. We suggest that you use the evaluation based interpreter for IS and the graphical model-based interpreter for MH within Gibbs. Your HMC sampler can operate in either representation but is conceptually simpler to implement in the graphical model-based evaluator.

Note that the final program below is a “trick” question intended more to make you think about the problem it represents that to produce a good solution. Demonstrated creativity here will be appreciated and rewarded.

Additionally you may find it interesting to think about how to apply HMC in the setting of non-differentiable joint densities. There is a paper you might consider reading on the subject.

Rubric

Submit a `.pdf` file to gradescope.ca showing your answers for each section. Provide code snippets that document critical aspects of your implementation sufficient to allow us to quickly determine whether or not you individually completed the assignment.

You will be largely assessed based on the correctness of your implementation in terms of convergence of posterior expectations within wall-clock time constraints. Posterior expectations, variances or other quantities where indicated that are all within 5% of ground truth for a single program within 10 minutes of wall-clock runtime will be considered correct (please report runtimes for each sampler and each program). Each program is worth 12 points, 4 each for each sampler type. You must include sample trace plots and log joint density plots for every return variable in P1, P2, and P5 for HMC, and Gibbs sampling as a function of sampler iteration number. Additionally you must plot histograms of the posterior distributions obtained for each method for each program.

Program 1) (IS, MH Gibbs, and HMC)

```
(let [mu (sample (normal 1 (sqrt 5)))
           sigma (sqrt 2)
           lik (normal mu sigma)]
       (observe lik 8)
       (observe lik 9)
       mu)
```

Report the posterior mean and variance of `mu`.

Program 2) (IS, MH Gibbs, and HMC)

```
(defn observe-data [_ data slope bias]
  (let [xn (first data)
        yn (second data)
        zn (+ (* slope xn) bias)]
    (observe (normal zn 1.0) yn)
    (rest (rest data))))

(let [slope (sample (normal 0.0 10.0))
      bias  (sample (normal 0.0 10.0))
      data (vector 1.0 2.1 2.0 3.9 3.0 5.3
                   4.0 7.7 5.0 10.2 6.0 12.9)]
  (loop 6 data observe-data slope bias)
  (vector slope bias))
```

Report the posterior means and covariance matrix of `slope` and `bias`.

Program 3) (IS, MH Gibbs)

```
(let [data [1.1 2.1 2.0 1.9 0.0 -0.1 -0.05]
      likes (foreach 3 []
                     (let [mu (sample (normal 0.0 10.0))
                           sigma (sample (gamma 1.0 1.0))]
                       (normal mu sigma)))
      pi (sample (dirichlet [1.0 1.0 1.0]))
      z-prior (discrete pi)
      z (foreach 7 [y data]
          (let [z (sample z-prior)
                _ (observe (get likes z) y)]
            z))]
  (= (first z) (second z)))
```

Report the posterior probability that the first and second datapoint are in the same cluster, i.e. the posterior probability that z[1] == z[2].

Program 4) (IS, MH Gibbs)

```
(let [sprinkler true
      wet-grass true
      is-cloudy (sample (flip 0.5))

      is-raining (if (= is-cloudy true )
                    (sample (flip 0.8))
                    (sample (flip 0.2)))
      sprinkler-dist (if (= is-cloudy true)
                        (flip 0.1)
                        (flip 0.5))
      wet-grass-dist (if (and (= sprinkler true)
                               (= is-raining true))
                        (flip 0.99)
                        (if (and (= sprinkler false)
                                  (= is-raining false))
                          (flip 0.0)
                          (if (or (= sprinkler true)
                                   (= is-raining true))
                            (flip 0.9))))]
  (observe sprinkler-dist sprinkler)
  (observe wet-grass-dist wet-grass)
  is-raining)
```

Report the posterior probability that it is raining, i.e. of “is-raining.”

Program 5) (IS, MH Gibbs, HMC)

```
(let [x (sample (normal 0 10))
      y (sample (normal 0 10))]
  (observe (dirac (+ x y)) 7)
  [x y])
```

Report the marginal means and *variances* of the posterior of both `x` and `y`. Also, describe the strategy you used to solve this problem.