
Problem for Day 1. ***(optional)*** *Consider the statistical inverse problem*

$$Y = f(X) + E, \tag{1}$$

where $X \in \mathcal{N}(0, I_n), E \sim \mathcal{N}(0, \sigma^2 I_m)$ *where* I_n *and* I_m *are an* $n \times n$ *and* $m \times m$ *identity matrices,* $\sigma > 0$ *is the standard deviation of noise, and* $f : \mathbb{R}^n \to \mathbb{R}^m$ *is the forward operator. Furthermore, we can write the probability density function of* X *and* E *as*

$$\pi_X(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^n}} \exp\left(-\frac{1}{2}\|\mathbf{x}\|_2^2\right),$$
$$\pi_E(\mathbf{e}) = \frac{1}{\sqrt{(2\pi)^m \sigma^2}} \exp\left(-\frac{\|\mathbf{e}\|^2}{2\sigma^2}\right).$$

1. *Use Bayes' theorem to write the density function of the posterior distribution* $\pi_{X|Y=\mathbf{y}}(\mathbf{x})$ *up to a proportionality constant:*

   $$\pi_{X|Y=\mathbf{y}} \propto \cdots$$

   Assume that f *is smooth but not necessarily linear. Only express it in terms of* f, $\mathbf{x}$, $\mathbf{y}$, *and* σ.

2. *Show that the maximum a posteriori (MAP) estimate*

   $$\mathbf{x}_{MAP} := \arg\max_{\mathbf{x}} \ \pi_{X|Y=\mathbf{y}}(\mathbf{x})$$

   is equivalent to the solution of a Tikhonov regularized optimization problem:

   $$\mathbf{x}_{Tik} := \arg\min_{\mathbf{x}} \ \|f(\mathbf{x}) - \mathbf{y}\|_2^2 + \lambda\|\mathbf{x}\|^2,$$

   for some $\lambda > 0$. *Express* λ *in terms of* σ.

Problem for Day 2. *In this exercise we will sample from a standard normal distribution using the acceptance/rejection sampling method, based on the Cauchy proposal distribution. Consider the target density function (standard normal density function)*

$$f(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right),$$

and the proposal density function (Cauchy density function)

$$g(x) = \frac{1}{\pi(x^2+1)}.$$

open the code `exercise_4.py` *in day 2 folder and follow the following instructions.*

1. *write a Python functions* `f(x)` *and* `g(x)` *that computes the density functions of target Gaussian density f and proposal Cauchy density g.*

2. *set the step size $c = 2$ for acceptance/rejection algorithm and draw 2000 samples from the distribution of.*

3. *Plot the histogram of the samples and show that they approximate a standard-normal distribution.*

4. *Choose the step-size $c = \{1.1.52, 2\}$, and repeat the sampling. Compute the number of accepted samples. Which value is the best and why?*

Problem for Day 3. *Consider the statistical inverse problem*

$$Y = \mathcal{A}X + E, \tag{2}$$

where $X \in \mathbb{R}^2$ and $Y \in \mathbb{R}^3$ are real-valued multivariate random variables and $\mathcal{A} \in \mathbb{R}^{3\times 2}$ is a matrix given by

$$\mathcal{A} = \begin{pmatrix} 1 & -1 \\ 1 & -2 \\ 2 & 1 \end{pmatrix}.$$

Further more, suppose that the prior X and noise E follow the distributions $X \sim \mathcal{N}(0, I_2)$ and $E \sim \mathcal{N}(0, \sigma^2 I_3)$, where I_2 and I_3 are the 2 and 3 dimensional identity matrix, respectively.

1. *Use Bayes' theorem to write the density function of the posterior distribution $\pi_{X|Y=\mathbf{y}}(\mathbf{x})$ up to a proportionality constant:*

$$\pi_{X|Y=\mathbf{y}} \propto \cdots$$

   only express it in terms of $\mathcal{A}$, $\mathbf{x}$, $\mathbf{y}$, σ.

2. *open the code* `exercise.py` *in day 3 folder and follow the following instructions*

   (a) *write a function* `prior` *that computes the prior probability density of an input* `x`*.*

   (b) *define a Numpy matrix* `A` *that holds $\mathcal{A}$.*

   (c) *write a functions* `likelihood` *and* `posterior` *that computes the likelihood probability density and posterior probability density for an input* `x`*. Here use* `sigma` *and* `y_obs` *that is provided in the code.*

   (d) *Complete the code for the Metropolis-Hastings random walk algorithm based on transition strategy*

$$\mathbf{x}^\star = \mathcal{N}(\mathbf{x}, c^2 I_2),$$

   or equivalently

$$\mathbf{x}^\star = \mathbf{x} + c\mathbf{z}.$$

   Here, $\mathbf{z}$ is a sample from $\mathcal{N}(0, I_2)$ and c is the step-size in the random walk Metropolis-Hastings algorithm.

3. *Draw 50000 samples from the posterior distribution. Down-sample (skip every 10 samples) to create a near i.i.d. samples of the posterior. plot a 2D histogram of the posterior and mark the posterior mean on it.*

4. *Let the step size be $c \in \{0.001, 0.1, 10\}$. Plot the posterior 2D histogram for each step size and explain the differences. In your opinion, which value of c is better for this problem, and why?*

5. *Let noise variance be $\sigma^2 \in \{0.01, 0.1, 1\}$. Plot the posterior 2D histogram for each noise variance. What differences do you observe? Discuss uncertainty in the posterior mean estimation.*

Problem for Day 4. *In this problem, we consider the statistical inverse problem of 1D de-blurring, formulated as*

$$Y = \mathcal{A}X + E, \tag{3}$$

where X, Y and E are all random variables in $\mathbb{R}^n$, and $\mathcal{A} \in \mathbb{R}^{n\times n}$ is a blurring matrix. Complete the Python code `exercise_1.py` *in day 4 folder following these instructions.*

1. *the vector* `s` *contains a discretization of the interval $[0, 1]$ with step size $\Delta s = 1/100$. Use* `s` *to create a vector* `x` *that approximates the signal*

$$x(s) = \begin{cases} 1, & 0.2 \le s \le 3.5, \\ 2, & 0.5 \le s \le 0.7, \\ 0 & \text{otherwise.} \end{cases}$$

2. *Use the Python function* `A` *to add blurr to* `x` *and create blurred measurement* `y` *and add noise to create noisy measurement* `y_obs`*, using blurring standard deviation $\delta = 1$ (in the code this is defined by* `delta`*). Plot the original signal* `x` *and the blurred measurements* `y` *and noisy measurements* `y_obs`*.*

Now we use the Bayes' theorem to construct the posterior as

$$\pi_{X|Y=\mathbf{y}}(\mathbf{x}) \propto \pi_{Y|X}(\mathbf{y})\pi_X(\mathbf{x})$$

and investigate different choices for the prior $\pi_X(\mathbf{x})$.

Uncorrelated prior: *the first exercise is to choose the prior $\pi_X(\mathbf{x})$ to be*

$$X \sim \mathcal{N}(0, I_n),$$

where I_n is the n-dimensional identity matrix. Continue completing the code following these instructions.

3. *Write the density function of the posterior distribution up to the proportionality constant when $E \sim \mathcal{N}(0, \sigma^2 I_n)$, i.e.,*

$$\pi_{X|Y=\mathbf{y}}(\mathbf{x}) \propto \cdots$$

4. *Complete the code in* **`exercise_2.py`** *using the steps below in order to enable sampling from the posterior distribution of this inverse problem:*

   (a) *write a Python function* **`prior`** *that computes the log (un-normalized) prior density, i.e.,*

   $$\log \pi_X(\mathbf{x}) = -\frac{1}{2}\|x\|_2^2$$

   (b) *write a Python function* **`likelihood`** *that computes the log (un-normalized) likelihood density, i.e.,*

   $$\log \pi_{Y|X=\mathbf{x}}(\mathbf{y}) = -\frac{1}{2\sigma^2}\|\mathbf{y} - \mathcal{A}\mathbf{x}\|_2^2$$

   (c) *write a Python function* **`posterior`** *that computes the log (un-normalized) posterior deinsty.*

   (d) *using the random-walk Metropolis-Hasting algorithm, draw 10000 samples. Choose the step size c that gives you about 23% acceptance rate.*

5. *plot the posterior mean.*

6. *plot the point-wise variance as a measure of uncertainty, i.e., limit your samples to a point, e.g., $s = i\Delta s$ for a particular i, now your samples are real-valued samples. You can compute the variance of these samples. Then repeat for all points in* **`s`**.

7. *Where are the regions that has high(er) uncertainty.*

Increment (correlated) prior: *In this exercise we will set the prior on increments instead of pixel values. Let us first introduce the random variable for the increments:*

$$\mathbf{z}_i = \mathbf{x}_{i+1} - \mathbf{x}_i, \qquad i = 1, \ldots, n-1,$$

and let $\mathbf{z}$ *be the vector that contains* $\mathbf{z}_i$*. Define the Matrix* T *that transforms* $\mathbf{z}$ *into* $\mathbf{x}$*, i.e.,* $\mathbf{x} = T\mathbf{z}$

8. *Write how the matrix* T *look like.*

Now let Z *be the random variable associate with the vector* $\mathbf{z}$*, with the prior density function* $\pi_Z(\mathbf{z})$ *(the exact density will be discussed below).*

8. *Reformulate the statistical de-blurring inverse problem using the change of variable* $X = TZ$ *and* $\mathbf{x} = T\mathbf{z}$*. (you need to reformulate the prior to be* π_Z *and the likelihood to be* $\pi_{Y|Z}$*)*

Now we will investigate different types of priors for Z*. We will use Gaussian prior, Laplace prior and Cauchy priors. Laplace and Cauchy priors will promote sparsity in jumps, i.e., we, a priori, expect fewer jumps in the signal.*

8. *In the Python code* **`exercise_3.py`** *for day 4, write 3 functions that can compute the log density function for the prior, once for the Gaussian* $\mathcal{N}(0, \sigma^2_{prior} I_n)$, $\sigma_{prior} = 0.1$,

$$\pi_Z(\mathbf{z}) \propto \exp(-\frac{\|\mathbf{z}\|_2^2}{2\sigma^2_{prior}}),$$

once for the Laplace prior Laplace$(0, b)$*, zero mean and scaling* $b = 0.05$

$$\pi_Z(\mathbf{z}) \propto \exp(-\frac{\|\mathbf{z}\|_1}{b})$$

and Once for the Cauchy Cauchy$(0, \gamma)$*, zero mean and scaling* $\gamma = 0.005$.

$$\pi_Z(\mathbf{z}) \propto \prod_{i=1}^{n-1} \frac{1}{1 + (\frac{\mathbf{z}_i^2}{\gamma^2})}$$

Don't forget to take the log of the prior!!!

9. *Repeat the sampling as with the uncorrelated case. For each of the choices of prior make sure that you set the step size in the Metropolis-Hastings method such that you achieve acceptance rate of about 23%. Plot the posterior mean and point-wise variance. Explain the differences with respect to the uncorrelated prior. Which of these priors are the most suitable for the problem and why?*

Problem for Day 5. *In this exercise we will create a smooth prior and use it to solve the hydraulic conductivity problem.*

To create a smooth prior we use a covariance kernel $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ *and define a covariance matrix* $\mathcal{C}$*. We can the use this covariance matrix to define a Gaussian random variable.*

Use the code **`exercise_1.py`** *in day 5 folder and follow the instructions below*

1. *create a discretization of the interval* $[0, 1]$ *as a vector* $\mathbf{s}$ *and store it in a Python variable* `s`.

2. *create a python function* **`gaussian_cov_func`** *to evaluate the Gaussian covariance kernel:*

$$f(\mathbf{x}, \mathbf{x}') = \exp(-\frac{|\mathbf{x} - \mathbf{x}'|^2}{2\ell^2})$$

with $\ell = 0.05$ *the correlation length.*

3. *create a covariance matrix* $\mathcal{C}$ *with elements:*

$$[\mathcal{C}]_{ij} = f(\mathbf{s}_i, \mathbf{s}_j).$$

4. *use* **`np.random.multivariate_normal`** *to draw 5 samples from* $\mathcal{N}(0, \mathcal{C})$*. Plot the samples.*

5. *repeat the experiment for* $\ell = 0.2, 0.1$ *and* 0.05*. And plot the samples. How does* ℓ *effects the samples?*

6. *now repeat the experiment above with the exponential kernel:*

$$f(\mathbf{x}, \mathbf{x}') = \exp(-\frac{|\mathbf{x} - \mathbf{x}'|}{\ell})$$

How does the behavior of the samples change?

7. *Often in inverse problems, it is preferred to work with independent samples. However, components of $\mathcal{N}(0, \mathcal{C})$ are inherently correlated. We can perform a change of variable and create a set of independent variables using the change of variables*

$$X = \mathcal{C}^{1/2} Z \tag{4}$$

where X is the original random variable, $\mathcal{C}^{1/2}$ is the Cholesky factor of $\mathcal{C}$, and now, Z is a new variable with distribution $\mathcal{N}(0, I_n)$. For the Gaussian kernel above draw 5 samples by first drawing $Z \sim \mathcal{N}(0, I_n)$ and then multiplying them by the $\mathcal{C}^{1/2}$.

we will now use this prior to formulate the posterior distribution of the hydraulic imaging problem.

The code **`hydraulic.py`** *contains the numerical approximation of the forward operator. You can follow the code* **`exercise_2.py`** *to see how you can pass a conductivity (the unknown $\mathbf{x}$) to create 5 pressure profiles for 5 number of injections. The code will return a matrix where the i-th row corresponds to the discrete pressure profile for i-th injection/measurement.*

For the Hydraulic inverse problem we consider the statistical inverse problem:

$$Y_i = F_i(X) + E_i, \quad i = 1, \dots 5 \tag{5}$$

where F_i is the forward operator of the ith injection (in the code, the **`hydraulic.py`** *will perform all F_i simultanuously), And $E_i = \mathcal{N}(0, \sigma_i I_n)$, for $i = 1, \dots, 5$. Remember that the noise standard deviations σ_i are different for each measurement.*

8. *For a prior distribution $X \sim N(0, \mathcal{C})$ write the posterior distribution, up to constant of proportionality, i.e.,*

$$\pi_{X|Y_1=\mathbf{y}_1,\dots,Y_5=\mathbf{y}_5}(\mathbf{x}) \propto \cdots \tag{6}$$

Hint: Use the fact that the 5 measurements are independent, and therefore,

$$\pi_{Y_1,\dots,Y_5|X}(\mathbf{y}_1, \dots, \mathbf{y}_5) = \pi_{Y_1|X}(\mathbf{y}_1) \cdots \pi_{Y_5|X}(\mathbf{y}_5).$$

9. *Reformulate and rewrite the posterior using the change of variable (4), i.e.,*

$$\pi_{Z|Y_1=\mathbf{y}_1,\dots,Y_5=\mathbf{y}_5} \propto \cdots$$

Recall that in this case the prior distribution is simply

$$Z \sim \mathcal{N}(0, I_n)$$

10. *One measurement data is given to you in the file* **`obs.pickle`**. *The Python code* **`exercise_3.py`** *will read this file and create 2 variables* **`y_obs`** *and* **`sigma`** *which will hold the measurements $\mathbf{y}_1, \dots, \mathbf{y}_5$ and the noise standard deviations $\sigma_1 \dots, \sigma_5$.*

11. *complete the code in* `exercise_3.py` *to sample the posterior. Plot the posterior mean and point-wise posterior variance. Where in the solution do you suspect that there is a sudden increase in porosity?*

    These are steps to complete `exercise_3.py`

    (a) *create a discretization of the interval* $[0, 1]$ *in a vector* `s` *with* `N=100` *points.*

    (b) *initiate the hydraulic class with* `N` *with the command* `hydraulic = hydraulic_class(N)` .

    (c) *Create a covariance matrix of the prior using a Gaussian covariance kernel with length scale 0.1, i.e.,* `length_scale=0.1`.

    (d) *create the Cholesky factor of the covariance matrix.*

    For part (c) and (d) you can copy the code from the previous exercise.

    (e) *create a Python function that evaluates the log of the un-normalized prior density* $\pi_Z(\mathbf{z})$ *(standard normal distribution).*

    (f) *load the measurement file* `obs.pickle`.

    (g) *create a Python function that evaluates the log of the un-normalized likelihood density.*

    (h) *combine the two to create a Phython function that evaluates the log of posterior.*

    (i) *perform the random walk Metropolis-Hastings sampling method to draw sample 10000 samples from the posterior. The step size* $c = 0.003$ *is appropriate for this problem.*

    for (i) you can copy the code from the exercises in day 4.

    (j) *compute the mean and the point-wise variance.*

12. ***(optional)*** *what happens to uncertainty in estimation when pressure sensors are broken in half of the domain? You can simulate faulty pressure sensors by throwing away half of the output of* `hydraulic.forward`*, i.e., the forward operator becomes*

    ```
    hydraulic_broken = lambda x:  hydraulic.forwad(x)[:,N/2:]
    ```

    similarly you should discard half of the measurements in `y_obs`*, i.e.,*

    ```
    y_obs_broken = y_obs[:,N/2:]
    ```