DD2434 Machine Learning, Advanced Course
Assignment 1

Carl Henrik Ek (adapted by Pawel Herman)

Deadline 12:00 (noon) November 29th, 2016

You will present the assignment by a written report that you can mail to me at paherman@kth.se before the deadline. Please include "[mladv16]" in the subject line. From the report it should be clear what you have done and you need to support your claims with results. You are supposed to write down the answers to the specific questions detailed for each task. This report should clearly show how you have drawn the conclusions and come up with the derivations. Your assumptions, if any, should be stated clearly. For the practical part of the task you are not expected to include your code in the report but rather demonstrate and focus on the results of your experiments, i.e. show images and graphs together with your analysis.

Being able to communicate your results and conclusions is a key aspect of any scientific practitioner. It is up to you as a author to make sure that the report clearly shows what you have done. Based on this, and only this, we will decide if you pass the task. No detective work should be needed on our side. Therefore, neat and tidy reports please!

As far as the implementation is concerned, please use any programming/scripting language/environment that you feel comfortable with. I can just recommend Python with its suitable libraries and all the clues regarding the implementation are given here with Python in mind.

The grading of the assignments will be as follows,

- **E** Completed Task 2.1 and 2.2.
- **D** E + Completed Task 2.3.
- **C** D + Completed Task 2.4.
- **B** C + Completed Task 2.5. (in preparation)
- **A** B + Completed Task 2.6. (in preparation)

Tasks 2.5 and 2.6 will be published soon.

Abstract

In this assignment we will examine several different aspects of building models of data. In the first task we will look at a supervised scenario where we work with a model of a specific relationship between two different domains. This is a very common problem where we have observations in one domain, say an image of a face and then wish to infer the identity of the person. The second task we look at how we can perform unsupervised learning and learn a new representation of the data. This is related to finding hidden structures or patterns in the data which might contain important information. Finally we will end with a look at how we can approach model selection. This is very important as it gives us the tool to design different models and then choose the one that best represents our data. The important message that these exercises tries to convey is how we can integrate our beliefs with observations using a set of simple rules. The assignments are aimed at showing the key aspects of data modeling in a simple scenario such that our insights about the models are not "clouded" by the complexity data. It is left to you as a student to extend this knowledge to a realistic scenario with real data.

Remark Please bear in mind that the notation here is a bit different from what you are used to in the lectures. In short:

- vectors are marked as $\mathbf{x}$ instead of $\boldsymbol{x}$
- vector $\mathbf{y}$ corresponds in most places to $\boldsymbol{t}$ (we observe noisy $\boldsymbol{t}$ that describe unknown $\boldsymbol{y}=f(\boldsymbol{x})$)
- $\mathbf{X}$ and $\mathbf{Y}$ describe matrices of data $\mathcal{D}$ (vectors of inputs $\boldsymbol{x}_i$ and outputs $\boldsymbol{t}_i$, respectively, stack together)
- matrix $\mathbf{W}$ describes parameters (not $\boldsymbol{w}$) since we allow a general case of multidimensional outputs ($\mathbf{y}$ not y)

I The Prior $p(\mathbf{X})$, $p(\mathbf{W})$, $p(f)$

2.1 Theory

Regression is the task of estimating a continuous target variable $\mathbf{Y}$ from an observed variate $\mathbf{X}$. The target and the observed variates are related to each other through a mapping,

$$f : \mathbf{X} \rightarrow \mathbf{Y}, \tag{1}$$

where f indicates the mapping. Given input output pairs $\{\mathbf{x}_i, \mathbf{y}_i\}_1^N$ our task is to estimate the mapping f such that we can infer the associated $\mathbf{y}_i$ from previously unseen $\mathbf{x}_i$. In this task we will work with real vectorial data such that $\mathbf{x}_i \in \mathbf{X}$ where $\mathbf{x}_i \in \mathbb{R}^q$ and $\mathbf{y}_i \in \mathbf{Y}$ where $\mathbf{y}_i \in \mathbb{R}^D$. Being probabilistic means that we need to consider the uncertainty in both the observations as well as the relationship between the variates. Starting with the relationship between two *single* points $\mathbf{x}_i^j$ and $\mathbf{y}_i^j$ we can assume the following form of the likelihood,

$$p(\mathbf{y}_i^j | f, \mathbf{x}_i^j) \sim \mathcal{N}(f(\mathbf{x}_i), \sigma^2\mathbf{I}). \tag{2}$$

> **Question 1:** *Why Gaussian form of the likelihood is a sensible choice? What does it mean that we have chosen a spherical covariance matrix for the likelihood?*

Assuming that each output point is independent given the input and the mapping we can write the likelihood of the data as follows,

$$p(\mathbf{Y}|f, \mathbf{X}) = \prod_i^N p(\mathbf{y}_i | f, \mathbf{x}_i). \tag{3}$$

> **Question 2:** *If we do* **not** *assume that the data points are independent how would the likelihood look then? Remember that* $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N]$

The task of regression means that we wish to infer $\mathbf{y}_i$ from its corresponding variate $\mathbf{x}_i$. These two variates are related to each other by the mapping f so from a probabilistic view point we wish to find the mapping from the observed data. More specifically, taking uncertainty into account, what we wish to reach is the posterior distribution over the mapping given the observations,

$$p(f|\mathbf{X}, \mathbf{Y}). \tag{4}$$

2.1.1 Linear Regression

Please read carefully Chapter 3 in (Bishop 2006) before you start this part of the task. In order to proceed lets make an assumption about the mapping and model the relationship between the variates as a linear mapping. Further, let's make an assumption about the structure of the noise in the observations, assuming that they have been corrupted by additive Gaussian noise,

$$\mathbf{y}_i = \mathbf{W}\mathbf{x}_i + \boldsymbol{\epsilon}, \tag{5}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$. From this we can formulate the likelihood of the data,

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{W}) = \tag{6}$$

> **Question 3:** *What is the specific form of the likelihood above, complete the right-hand side of the expression.*

The inference task we are interested in here is to learn the mapping, i.e. to infer $\mathbf{W}$ from the data,

$$p(\mathbf{W}|\mathbf{X}, \mathbf{Y}) = \frac{1}{Z} p(\mathbf{Y}|\mathbf{X}, \mathbf{W}) p(\mathbf{W}). \tag{7}$$

In the above equation we can see that we need to formulate our belief of the model parameters $\mathbf{W}$ in a prior $p(\mathbf{W})$. We can make many different choices of priors, but a sensible choice would be to pick the conjugate prior i.e. a Gaussian prior over the parameters,

$$p(\mathbf{W}) = \mathcal{N}(\mathbf{W}_0, \tau^2\mathbf{I}). \tag{8}$$

> **Question 4:** *Explain the concept of conjugate distributions. Why is this a motivated choice?*

The prior distribution provides us with a tool that tells us how likely or "how far" a parameter is from our belief.

> **Question 5:** *The prior in Eq.8 is a spherical Gaussian. This means that the "preference" is encoded in terms of a* L_2 *distance in the space of the parameters. With this view, how would the preference change if the preference was rather encoded using a* L_1 *norm? Compare and discuss the different type of solutions these two priors would encode.*

The posterior is the object that integrates our prior beliefs with the data. In the next section we will see how this works in practice for the linear regression model derived above.

Question 6: *Derive the posterior over the parameters. Please, do these calculations by hand as it is very good practice. However, in order to pass the assignment you only need to outline the calculation and highlight the important steps.*

- *Why does it have the form that it does?*
- *What is the effect of the constant Z, are we interested in this?*

2.1.2 Non-parametric Regression

In the previous task we made the assumption that the relationship between the two variates was linear. This is quite a strong assumption that severely restricts the representative power of our model. The obvious way to proceed would be to add more parameters to f and use a higher-degree polynomial, but which one should we pick, degree 3 or 4, should we add a trigonometric function? These are tricky questions that require a lot of knowledge about the specific data that we are looking at. We want to stay general however, and the whole idea about Machine Learning is that we want the *data* to tell us all that without any need for us to specify it before we start building our model (beyond what prior can account for).

Let's take a step back and think how a Bayesian would think in this situation. The above argument just says that we have a large uncertainty in, not only in the the parameters of the mapping, but also in the actual *form* of the mapping. Bayesian reasoning allows us to deal with this, it is actually *exactly* this scenario that it was designed to deal with. We just need to formulate our uncertainty about the mapping in a prior over mappings and then use Bayes rule to reach the posterior. The problem is just that we need to somehow formulate a prior over a space of functions, which is quite a lot stranger mathematical object compared to the scalar valued parameters $\mathbf{W}$ in the previous task. Before proceeding with this task please read (Bishop 2006, p. 303-311).

We know from the lecture that Gaussian Processes ($\mathcal{GP}$s) can be used to represent prior distributions over the space of functions. Rather than specifying a specific parametric form of the function $\mathcal{GP}$s is a non-parametric model.

Question 7: *What is a non-parametric model and what is the difference between non-parametrics and parametrics? In specific discuss these two aspects of non-parametrics,*

- *Representability?*
- *Interpretability?*

We will now proceed to look at the regression problem where we replace the linear assumption in the mapping to use a non-parametric prior over the space of functions. Let's make the same assumption about the observations as we did in the linear case,

$$\mathbf{y}_i = f(\mathbf{x}_i) + \boldsymbol{\epsilon}. \tag{9}$$

This allows us to formulate the likelihood in the same manner as before. However, in the linear example we could easily formulate the relationship between $\mathbf{x}$, $\mathbf{y}$ and f as the latter had a simple parametric form. Now we cannot do this anymore. To proceed, lets define the output of the function f as its own random variable,

$$\mathbf{y}_i = f_i + \boldsymbol{\epsilon}, \tag{10}$$

where f_i is the *output* of the function at input location $\mathbf{x}_i$. The next step is to formulate the prior over the output of the function. This we can do using a $\mathcal{GP}$,

$$p(f|\mathbf{X}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, k(\mathbf{X}, \mathbf{X})) \tag{11}$$

where $k(\cdot,\cdot)$ is the covariance function and $\boldsymbol{\theta}$ is its parameters. We will refer to $\boldsymbol{\theta}$ as the *hyper-parameters* of the process. In this we have assumed that the data have been translated such as to have zero-mean such that we do not need to have a mean function in the prior.

> **Question 8:** *Explain what this prior does? Why is it a sensible choice? Use images to show your reasoning. Clue: use the marginal distribution to explain the prior*

Given this formulation we can now formulate the full model.

> **Question 9:** *Formulate the joint likelihood of the full model that you have defined above,*
>
> $$p(\mathbf{Y}, \mathbf{X}, f, \boldsymbol{\theta})$$
>
> *(Try to draw a very simple graphical model to clearly show the assumptions that you have made.)*

Unfortunately, we have added to our model a new variable that we are not really interested in. Specifically we have modeled the relationship between $\mathbf{Y}$ and f and also f and $\mathbf{X}$ but we really are interested in the relationship between $\mathbf{Y}$ and $\mathbf{X}$. The motivation behind this is that we now have the possibility to have uncertainty in each of these stages, in our beliefs of the functions, and in how we believe the output of the function have generated the observed data. But again, we are not interested in f and therefore the variable should be marginalised out. Performing the marginalisation implies calculating the following integral,

$$p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = \int p(\mathbf{Y}|f)p(f|\mathbf{X}, \boldsymbol{\theta})\mathrm{d}f. \tag{12}$$

> **Question 10:** *Explain the marginalisation in Eq.12,*
>
> - *Explain how this connects the prior and the data?*
> - *How does the uncertainty "filter" through this?*
> - *What does it imply that $\boldsymbol{\theta}$ is left on the left-hand side of the expression after marginalisation?*

2.2 Practical

Now we will implement the approach we studied in the previous part. Remember to save images and figures to support your claims in part 1 as this will make the presentation much easier to examine. There are a couple of packages in Python that are really useful:

```
import pylab as pb
import numpy as np
from math import pi
from scipy.spatial.distance import cdist

# To sample from a multivariate Gaussian
f = np.random.multivariate_normal(mu,K);
# To compute a distance matrix between two sets of vectors
D = cdist(x1,x2)
# To compute the exponetial of all elements in a matrix
E = np.exp(D)
```

2.2.1 Linear Regression

In this task we will implement the linear regression that we looked at in the previous task. We will examine both the prior and the posterior over the parameters **W** and evaluate the effect this will have on the model. To do so we will need to have some data to experiment with. What we want to show is that the methodology that we have learned is capable of recovering the true underlying mapping from the observed data. Therefore let's generate some data and then simply throw the generating parameters away.

$$y_i = w_0 x_i + w_1 + \epsilon \tag{13}$$

$$\mathbf{x} = [-1, -0.99, \ldots, 0.99, 1] \tag{14}$$

$$\epsilon \sim \mathcal{N}(0, 0.3) \tag{15}$$

$$\mathbf{W} = [-1.3, 0.5] \tag{16}$$

Question 11:

1. *Visualise the prior distribution over* **W**.
2. *Pick a single data-point from the data and visualise the posterior distribution over* **W**.
3. *Sample from the posterior and show a couple of functions.*
4. *Repeat* $2 - 3$ *by adding additional data points.*

Describe the plots and the behavior when adding more data? Is this a desirable behavior?

2.2.2 Non-parametric Regression

In this task we will implement and evaluate the effect of a $\mathcal{GP}$-prior. Specifically we will look at the squared exponential covariance function,

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 e^{-\frac{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)}{l^2}} \tag{17}$$

You will need to first formulate the prior distribution and then the posterior. How to do this can be found in (Bishop 2006, p. 306-308). First we will look at the prior.

> **Question 12:**
>
> 1. *Create a $\mathcal{GP}$-prior with a squared exponential co-variance function.*
> 2. *Sample from this prior and visualise the samples.*
> 3. *Show samples using different length-scale for the squared exponential.*
>
> *Explain the behavior of altering the length-scale of the covariance function.*

This has been said many times before but it is something that cannot be stressed enough, priors are very important as they allow us to formulate our uncertainty in our beliefs in a principled manner. However, more important is that we can combine our beliefs with observations, this is what facilitates learning. The object that contains this is the posterior distribution. We will now perform a simple experiment on the posterior. The posterior and the prior are the same object if we do not have any observed data.

> **Question 13:** *Explain the above statement, why is this?*

Lets generate some data that we know would not work particularly well using a linear model as in the previous task,

$$y_i = \sin(x_i) + \epsilon_i \tag{18}$$
$$\mathbf{x} = [-\pi, \ldots, \pi]^{\mathrm{T}} \tag{19}$$
$$\epsilon_i \sim \mathcal{N}(0, 0.5), \tag{20}$$

where the cardinality of $\mathbf{x}$ is 7, i.e. we have 7 data points. Now we have some observations of a noisy sinewave which we can use together with our prior to see the posterior distribution over the functions.

> **Question 14:**
>
> 1. *Compute the predictive posterior distribution of the model*
> 2. *Sample from this posterior with points both close to the data and far away from the observed data.*
> 3. *Plot the data, the predictive mean and the predictive variance of the posterior from the data*
>
> *Explain the behavior of the samples and compare the samples of the posterior with the ones from the prior. Is this behavior desirable? What would happen if you would add a diagonal covariance matrix to the squared exponential?*

II The Posterior $p(\mathbf{X}|\mathbf{Y})$

In the previous task we looked at learning a relationship between two variates $\mathbf{X}$ and $\mathbf{Y}$ such that we could infer one from the other. One way of thinking about this is that given the mapping f and the input $\mathbf{X}$ we specify the outputs $\mathbf{Y}$, you can think of $\mathbf{X}$ as a "representation" of $\mathbf{Y}$, i.e. that the former have generated the later. Actually this is exactly what we did in the linear regression task, we generated some data using a set of parameters $\mathbf{w}$ then we threw them away and later recovered them back, but we retained $\mathbf{x}$ in this process. In this task we will make things a bit more complicated by looking at representation learning. This means we will only observe the outputs $\mathbf{Y}$ and want to learn input $\mathbf{X}$ that can represent $\mathbf{Y}$. Why would we ever want to do this? Lets take the example of an image. Images are very high-dimensional objects, a typical HD image $\mathbf{y}_i$ concatenated into a vector will live in a space of $\mathbb{R}^{1920\times1080\times3}$. However, does the image actually have that many degrees-of-freedom? To simplify, given that you know the place the image was taken, the weather, the exact camera angle, the objects in the image wouldn't you be able to generate the pixel data? This is of course a massive simplification but ponder how many parameters you can come up with and this will be less than the number of pixels in the image. Let's call all these factors that we came up with and refer to them as *generating parameters* just as we said that $\mathbf{X}$ through f generated $\mathbf{Y}$ in Task I. Representation learning allows you to recover these generating parameters directly from the data. More specifically this relates to building a model of the data $\mathbf{Y}$ and then looking at the posterior distribution over the input to the model $\mathbf{X}$.

The other new thing that we will introduce in this task is *learning*. This means that we specify a model like in Task I and then *fit* this model to the data. It implies that the model has a set of parameters, which we now will infer from the data.

2.3 Theory

The focus here will be on the same linear models as in the first part (though non-parametric Gaussian Processes in Part I can also be handled in the similar framework). The main difference is that the input locations $\mathbf{X}$ are not known a priori but rather we want to infer them from data. We will refer to the input locations as the *latent representation* of the observed data $\mathbf{Y}$. Think about how this relates to the latent space models that you worked on in the first part of the course, where you used discrete latent *states* to represent continuous data. This is very much the same thing, except for that we want to find the latent space from data and that rather than being a discrete variable it is continuous.

Lets start with the linear model,

$$p(\mathbf{Y}, \mathbf{X}, \mathbf{W}) = p(\mathbf{Y}|\mathbf{X}, \mathbf{W})p(\mathbf{X})p(\mathbf{W}). \tag{21}$$

The next step will take a bit of thinking, being fully Bayesian we would like to invert the model above and look at the conditional distribution over the variables that we want to infer. Think about this, does it make sense? Actually, there is a simple relationship between $\mathbf{X}$ and $\mathbf{W}$: having one implies having the other one. As an example, if each $\mathbf{x}_i$ is multipled by a constant, it is the same as dividing the $\mathbf{W}$ by the same constant. To get away from this is we should only look at a single variable. But as our model contains both $\mathbf{X}$ and $\mathbf{W}$, how can we do this? In the Bayesian spirit, we can specify a prior over the variable that we are not interested in and marginalise it out from the model.

What does this actually mean, previously we have been using prior distributions as a mean of encoding our beliefs about a variable before seeing data. Another equally valid explanation is as encoding our *preference* of a variable.

Question 15: *Elaborate on this, why can one view a prior as encoding a preference?*

Let's specify the prior over the latent variables as a spherical gaussian,

$$p(\mathbf{X}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{22}$$

Question 16: *What type of "preference" does this prior encode?*

Now we can combine this prior with the likelihood, integrate out $\mathbf{X}$ and reach the marginal distribution,

$$p(\mathbf{Y}|\mathbf{W}) = \int p(\mathbf{Y}|\mathbf{X}, \mathbf{W})\mathrm{d}\mathbf{X}. \tag{23}$$

Question 17: *Perform the marginalisation in Eq. 23 and write down the expression. As previously, it is recommended that you do this by hand even though you only need to outline the calculations and show the approach that you would take to pass the assignment.*

2.3.1 Learning

So far we have only created models and looked at the posterior. Now we will take one step further and learn the parameters of the model. A good background to what we will go through here can be found in (Bishop 2006, p. 9,23,26,30,165, sec. 1.2.4-1.2.6). Let's start to do learning in a probabilistic model with the maximum-likelihood (ML) approach. So, we formulate the likelihood of the data and find the parameters that maximise it,

$$\hat{\mathbf{W}} = \mathrm{argmax}_{\mathbf{W}} p(\mathbf{Y}|\mathbf{X}, \mathbf{W}). \tag{24}$$

The next level is to perform maximum-a-posteriori (*MAP*) estimation. This means that we find the parameters that maximise the posterior distribution,

$$\hat{\mathbf{W}} = \mathrm{argmax}_{\mathbf{W}} \frac{p(\mathbf{Y}|\mathbf{X}, \mathbf{W})p(\mathbf{W})}{\int p(\mathbf{Y}|\mathbf{X}, \mathbf{W})p(\mathbf{W})\mathrm{d}\mathbf{W}} = \mathrm{argmax}_{\mathbf{W}} p(\mathbf{Y}|\mathbf{X}, \mathbf{W})p(\mathbf{W}). \tag{25}$$

There is also an in-between stage which is often referred to as Type-II Maximum-Likelihood which implies maximisation of the marginal likelihood where you integrate out one parameter and then maximise over another,

$$\hat{\mathbf{W}} = \mathrm{argmax}_{\mathbf{W}} \int p(\mathbf{Y}|\mathbf{X}, \mathbf{W})p(\mathbf{X})\mathrm{d}\mathbf{X}. \tag{26}$$

Question 18: *Compare these three estimation procedures above in log-space.*

- *How are they different?*
- *How are MAP and ML different when we observe more data?*
- *Why are the two last expressions of Eq. 25 equal?*

In the representation task we have a model with two variables $\mathbf{W}$ and $\mathbf{X}$ that interact. This means that a Type-II ML estimation is a sensible approach to learn the model.

Practical Optimisation In practice when performing optimisation on probabilistic models we often have to deal with exponentials. Exponentials are nice in many ways, they are for example infinitely differentiable, but they are a bit tricky to play with. Often we have the case that our parameters are actually in the exponents and then we can do a neat trick that makes life much easier. Rather than working directly on the exponent we perform all our learning in the log-space instead. The reason that we can do that is because $\log(\cdot)$ is a monotonic function and therefore it will not alter the location of the extremes of the function. Further, most optimisation packages are designed to minimise a function rather than maximising it. This means that in practice we often formulate our optimisation problem as a minimisation of the *negative log* of a probablitiy,

$$\hat{\theta} = \text{argmax}_{\theta} p(\mathbf{Y}|\theta) = \text{argmin}_{\theta} - \log\left(p(\mathbf{Y}|\theta)\right). \tag{27}$$

Now we will write down the objective function and its gradients which means we need to do some matrix algebra. There are plenty of literature resources helpful when working with matrices. The following two references, [URL] (Petersen and Pedersen 2006) and [URL] (Magnus and Neudecker 1988), are particularly worth recommending.

Question 19:

1. *Write down the objective function* $-log(p(\mathbf{Y}|\mathbf{W})) = \mathcal{L}(\mathbf{W})$.
2. *Write down the gradients of the objective with respect to the parameters* $\frac{\delta \mathcal{L}}{\delta \mathbf{W}}$

In the practical section of this task you will perform the optimisation above for a real dataset.

2.4 Practical

This practical part includes what is considered the bread and butter for a machine learning scientist, i.e. working with data. Let's generate some data so that we know what we are looking to recover,

$$\mathbf{Y} = f_{\text{lin}}(f_{\text{non-lin}}(\mathbf{x})) \tag{28}$$
$$\mathbf{x} = [0, \ldots, 4\pi] \tag{29}$$
$$|\mathbf{x}| = 100 \tag{30}$$
$$f_{\text{non-lin}}(x_i) = [x_i \sin(x_i), x_i \cos(x_i)] \tag{31}$$
$$f_{\text{lin}}(x') = \mathbf{A}^{\text{T}}\mathbf{x}' \tag{32}$$
$$\mathbf{A} = \mathbb{R}^{10\times 2} \tag{33}$$
$$\mathbf{A}_{ij} \sim \mathcal{N}(0, 1). \tag{34}$$

The values in the linear mapping are drawn from an independent Gaussian as we do not really care about the specific form of the mapping we only care about its `rank`.

Now we have generated a dataset $\mathbf{Y} \in \mathbb{R}^{100\times 10}$ which has been generated from a one-dimensional *generating parameter* $\mathbf{x} \in \mathbb{R}^{1\times N}$. The aim is now to recover $\mathbf{x}$, i.e a single line, given only $\mathbf{Y}$. This is a very general and incredibly important task in machine learning, i.e. how to discover the parameters that have generated some observations. It appears in many applications such as computer vision and computational biology just to name a few. It is important as many types of data are represented in high-dimensional spaces, which are very hard to interpret, providing the true generating parameters allows us to analyse the data and hopefully find the casual behavior in the data.

2.4.1 Linear Representation Learning

We have an objective function and we have the gradients with respect to the parameters that we want to learn. The actual optimisation can be done with the use of gradient descent. This is well implemented in `scipy.optimise`. Have a look at URL for the different methods that are available. Below is the simple structure that you need to implement in order to get the `fmin` function working,

```
import numpy as np
import scipy as sp
import scipy.optimize as opt

def f(x, *args):
    # return the value of the objective at x
    return val

def dfx(x,*args):
    # return the gradient of the objective at x
    return val

x_star = opt.fmin_cg(f,x0,fprime=dfx, args=args)
```

Question 20: *Plot the representation that you have learned. Explain why it looks the way it does. Was this the result that you expected? Hint: Plot* $\mathbf{X}$ *as a two-dimensional representation.*

Good Luck!

III References

[1] C.M. Bishop. *Pattern recognition and machine learning.* 2006

[2] K.P. Murphy. *Machine Learning: A Probabilistic Perspective* The MIT Press, 2012.

[3] K.B. Petersen and M.S. Pedersen. (2006) The matrix cookbook. Technical report. Technical University of Denmark. https://www.ics.uci.edu/~welling/teaching/KernelsICS273B/MatrixCookBook.pdf

[4] J.R. Magnus *et al. Matrix differential calculus with applications in statistics and econometrics.* Wiley, 1998.

[5] J. Hensman *et al.* (2014) GPy: A Gaussian process framework in Python. https://github.com/SheffieldML/GPy

[6] D.J.C. Mackay. (1991) Bayesian methods for adaptive models. PhD thesis. California Institute of Technology. http://thess.library.caltech.edu/25/

[7] I. Murray and Z. Ghahramani. (2005) A note on evidence and Bayesian Occam's razor. Technical report. http://homepages.inf.ed.ac.uk/imurray2/pub/05occam/occam.pdf