Math Foundations of ML, Fall 2022

Homework #4

Due Wednesday Octomber 19 at 5:00pm ET

As stated in the syllabus, unauthorized use of previous semester course materials is strictly prohibited in this course.

1. Recall the bump basis $\{\phi_n(t)\}_{n=1}^N$ from Homework 2, Problem 3 (Linear approximation with "bump" functions), and its span $\mathcal{T}_N$ equipped with the standard inner product. The dual basis $\{\tilde{\phi}_n(t)\}_{n=1}^N$ can be used to find the sampling functions (reproducing kernel) for $\mathcal{T}_N$, as

$$f(\tau) = \sum_{n=1}^N \langle \boldsymbol{f}, \tilde{\boldsymbol{\phi}}_n\rangle \phi_n(\tau) = \left\langle \boldsymbol{f}, \sum_{n=1}^N \phi_n(\tau)\tilde{\boldsymbol{\phi}}_n \right\rangle = \langle \boldsymbol{f}, \boldsymbol{k}_\tau\rangle, \quad \text{where } \boldsymbol{k}_\tau = \sum_{n=1}^N \phi_n(\tau)\tilde{\boldsymbol{\phi}}_n.$$

   (a) Fix $N = 10$ and compute the dual basis vectors of the bump basis from Homework 2, Problem 3. That is, find $\tilde{\boldsymbol{\phi}}_1, \ldots, \tilde{\boldsymbol{\phi}}_{10}$ so that if

$$f(t) = \sum_{n=1}^{10} \alpha_n \phi_n(t),$$

   we can compute the $\{\alpha_n\}_{n=1}^N$ using

$$\alpha_n = \int_0^1 f(t)\tilde{\phi}_n(t)\, dt.$$

   Turn in a plot of each of the ten $\tilde{\phi}_n(t)$.
   Solution.
   Please see script "p1a.py" for the code and Figure 1 for the plots.

   (b) Take $N = 10$ and plot $k_\tau(t)$ as a function of t for $\tau = .371238$. Create an $\boldsymbol{f} \in \mathcal{T}_N$ by drawing the expansion coefficients $\boldsymbol{\alpha}$ at random (`alpha = randn(N,1);` in MATLAB), and verify that $\langle \boldsymbol{f}, \boldsymbol{k}_\tau\rangle = f(\tau)$.
   Solution.
   Please see script "p1b.py" for the code and Figure 2 for the plots. I generated the expansion coefficients

$$\boldsymbol{\alpha} = [-2.319, 1.281, -1.191, -0.310, -0.989, 0.551, -0.960, 2.334, -0.176, -1.041],$$

   then I have $f(\tau) = -1.0709557$ and $\langle \boldsymbol{f}, \boldsymbol{k}_\tau\rangle = -1.0709578$. Therefore, we verified that $\langle \boldsymbol{f}, \boldsymbol{k}_\tau\rangle = f(\tau)$ for $\tau = .371238$.

   (c) Create an image of the kernel $k(s,t)$ for $(s,t) \in [0,1] \times [0,1]$ for the basis above — use at least a few hundred points for each of the arguments s and t. (In MATLAB you can display using `imagesc`.)
   Solution.
   $k(s,t) = \sum_{n=1}^N \phi_n(\tau)\tilde{\phi}_n(t)$ for $(s,t) \in [0,1] \times [0,1]$. Please see script "p1c.py" for the code and Figure 3 for the image of the kernel.

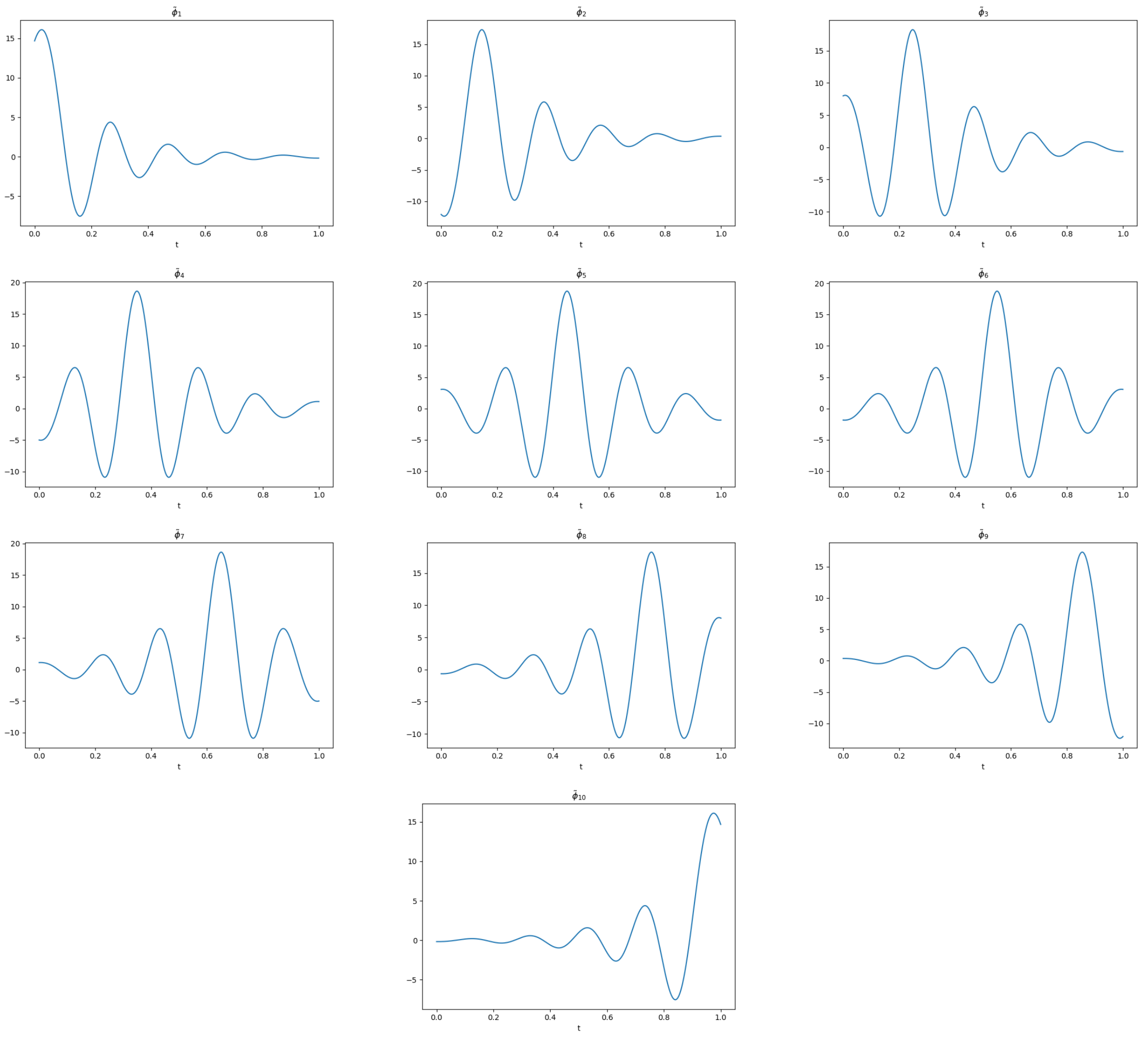

Figure 1: Each of the ten $\tilde{\boldsymbol{\phi}}_n$

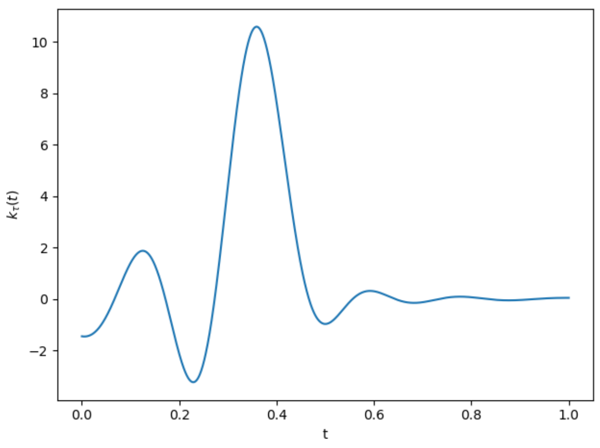

Figure 2: Plot of $k_\tau(t)$

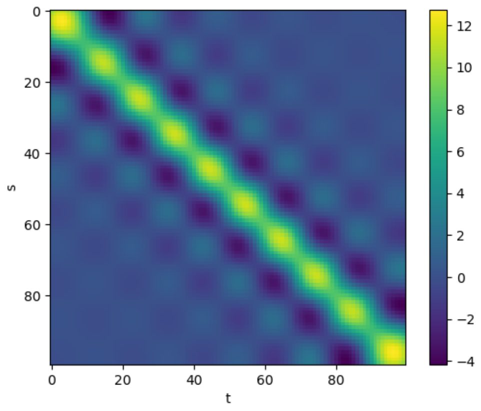

Figure 3: Image of the kernel $k(s,t)$.

2. In this problem, we will solve a stylized regression problem using the data set `hw04p2_data.mat`. This file contains (noisy) samples of a function $f(t)$ for $t \in [0,1]$. In fact, the data points were generated by sampling the function

$$f_{\text{true}}(t) = \frac{\sin(12(t+0.2))}{t+0.2}$$

at random locations then adding a random perturbation to the sample values. The sample locations are in the vector `T`, the sample values are in `y`. If you plot these, you will see that the samples are scattered more or less evenly across the interval. We are going to use kernel regression to form the estimate; in particular, we will use

$$k(s,t) = e^{-|t-s|^2/2\sigma^2}.$$

(a) Compute the kernel regression estimate with $\sigma = 1/10$ and $\delta = 0.004$. Plot your estimate $\hat{f}(t)$ overlaid on the data and $f_{\text{true}}(t)$. Compute the *sample error*[1]

$$\text{sample error} = \left(\sum_{m=1}^{M} |y_m - \hat{f}(t_m)|^2 \right)^{1/2},$$

and the *generalization error*

$$\text{generalization error} = \left(\int_0^1 |\hat{f}(t) - f_{\text{true}}(t)|^2 \right)^{1/2}$$

for your estimate. Comment on why this choice of σ was a good one.

Solution.

Please see script "p2a.py" for the code and Figure 4 for the plot. Sample error is 0.072407 and generalization error is 0.743171. The choice of σ yields a regression which closely matches the original curve with only the samples given (low generalization error), which is good.

[1]Also called the "training error".

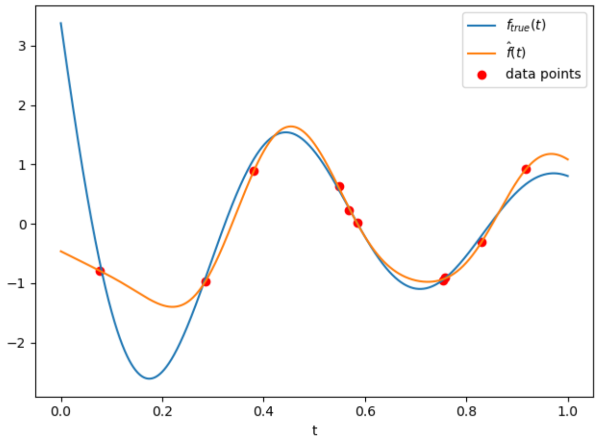

Figure 4: Plot of $\widehat{f}(t)$ overlaid on the data and $f_{\text{true}}(t)$.

σ	$1/2$	$1/5$	$1/20$	$1/50$	$1/100$	$1/200$
sample error	1.703	0.198	0.048	0.012	0.0084	0.0083
generalization error	1.200	0.553	0.984	1.179	1.229	1.255

Table 1: Sample and generalization errors for different σ's

(b) Repeat part (a) with $\sigma = 1/2, 1/5, 1/20, 1/50, 1/100, 1/200$, producing plots, sample errors, and generalization errors for your estimates for each σ. Comment on how the number of data points we see would affect the right choice of σ. *Solution.*
Please see script "p2b.py" for the code, Figure 5 for the plots and Table 1 for the sample and generalization erros. If σ is chosen too large, we see the model does not fit the samples well. If σ is too small, there is massive overfitting. As the number of samples go up, we would like to use smaller value of σ.

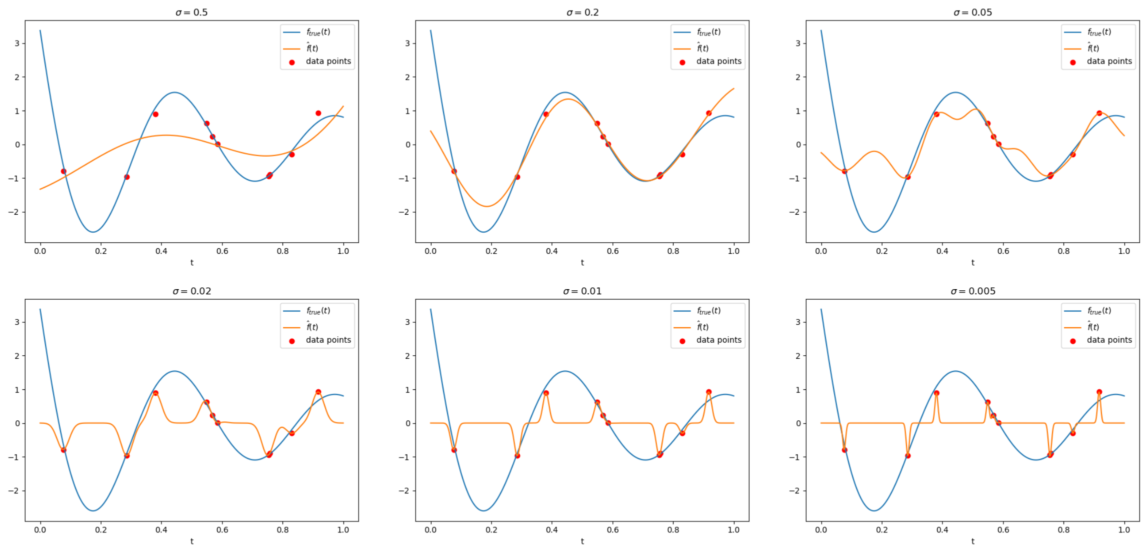

Figure 5: Plots for different σ.

3. Consider the set of bump basis vectors $\psi_1(t), \ldots, \psi_N(t)$, where

$$\psi_k(t) = g\left(t - k/N\right), \quad g(t) = e^{-100t^2} \tag{1}$$

Given a point t, define the nonlinear "feature map" as

$$\mathbf{\Psi}(t) = \begin{bmatrix} \psi_1(t) \\ \psi_2(t) \\ \vdots \\ \psi_N(t) \end{bmatrix}$$

Plot the feature map as a discrete set of coefficients[2] for $t = 1/3$ for $N = 10, 20, 50, 100, 200$. Compare to the radial basis kernel map

$$\mathbf{\Phi}_t(s) = k(s,t) = e^{-100|s-t|^2},$$

for $t = 1/3$ and $s \in [0,1]$. Discuss the relationship between kernel regression with a Gaussian radial basis function, and nonlinear regression using a basis of the form (1).

Solution.

Please see script "p3.py" for the code and Figure 6 for the plot. The feature map

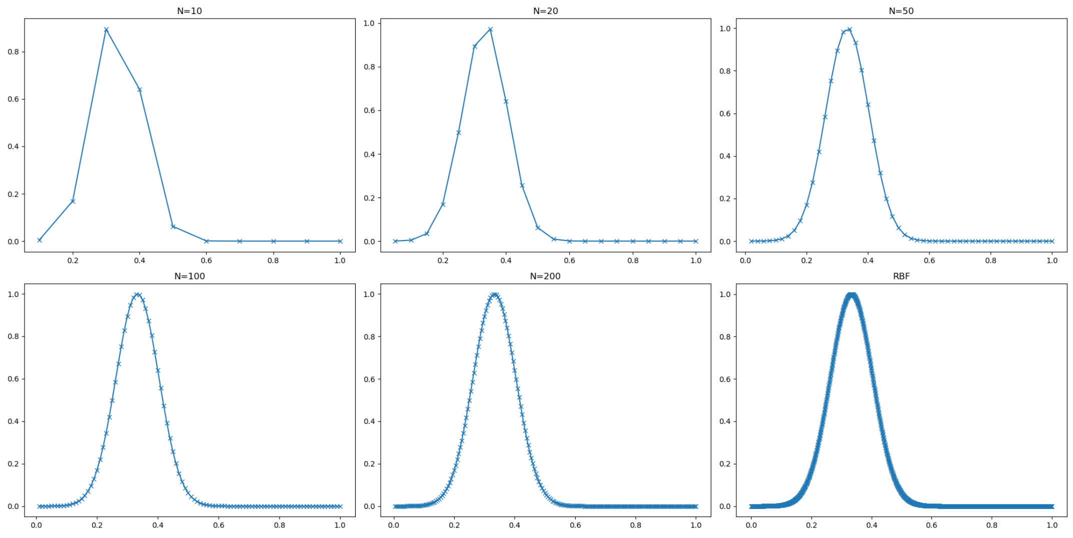

Figure 6: Basis functions for the various values of N and the radial basis kernel.

associated with the basis in (1) is finite dimensional whereas the feature map in kernel regression infinite dimensional. In fact, regression with the bases can be thought of as a finite dimensional approximation of the RBF kernel regression.

In kernel regression, the estimate at each new value of t is a linear combination of the kernel map sampled at the time instances which are present in the training data. In this case, if we had a set of training samples $\{t_m, y_m\}_{m=1}^M$, the estimate at $t_0 = 1/3$ would have been $y(t_0) = \sum_{m=1}^M \widehat{\alpha}\langle \phi(t_m), \phi(t)\rangle$. It can also be interpreted as a linear functional of the feature map itself: $y(t_0) = \langle \phi(t_0), \widehat{w}\rangle$.

[2]In MATLAB, use `plot(1:N,Psit(1:N),'o')`.

Similarly, for regression using the given basis, the estimate can be interpreted in two ways: first as a linear combination of the inner products of the feature map at t_0 and the feature maps of each of the training data point, second as a linear combination of the columns of the feature map matrix $\boldsymbol{A} = \begin{bmatrix} \boldsymbol{\Psi}(t_1)^T \\ \boldsymbol{\Psi}(t_2)^T \\ \vdots \\ \boldsymbol{\Psi}(t_M)^T \end{bmatrix}$.

For the two estimates to be equivalent as N increases, the two feature maps should be equivalent. For the present problem, $\Psi_k(t) = e^{-100(t-k/N)^2}$ would have resulted in such an equivalence.

4. Let

$$\boldsymbol{A} = \begin{bmatrix} 1.01 & 0.99 \\ 0.99 & 0.98 \end{bmatrix}$$

(a) Find the eigenvalue decomposition of $\boldsymbol{A}$ by hand. Recall that λ is an eigenvalue of $\boldsymbol{A}$ if for some $u[1], u[2]$ (entries of the corresponding eigenvector) we have

$$(1.01 - \lambda)u[1] + 0.99u[2] = 0$$
$$.99u[1] + (0.98 - \lambda)u[2] = 0.$$

Another way of saying this is that we want the values of λ such that $\boldsymbol{A} - \lambda\mathbf{I}$ (where $\mathbf{I}$ is the 2×2 identity matrix) has a non-trivial null space — there is a nonzero vector $\boldsymbol{u}$ such that $(\boldsymbol{A} - \lambda\mathbf{I})\boldsymbol{u} = 0$. Yet another way of saying this is that we want the values of λ such that $\det(\boldsymbol{A} - \lambda\mathbf{I}) = 0$. Once you have found the two eigenvalues, you can solve the 2×2 systems of equations $\boldsymbol{A}\boldsymbol{u}_1 = \lambda_1\boldsymbol{u}_1$ and $\boldsymbol{A}\boldsymbol{u}_2 = \lambda_2\boldsymbol{u}_2$ for $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$.

Show your work above, but feel free to check you answer using MATLAB/numpy.

Solution.

The characteristic polynomial of $\boldsymbol{A}$ is given by:

$$p_{\boldsymbol{A}}(\lambda) = \det(\lambda\mathbf{I} - \boldsymbol{A}) = \lambda^2 - 1.99\lambda + 0.0097.$$

The roots of $p_{\boldsymbol{A}}(\lambda)$ or the eigenvalues of $\boldsymbol{A}$ are $\lambda_1 = 1.98511$ and $\lambda_2 = 0.004886$. Solving the 2×2 systems of equations $\boldsymbol{A}\boldsymbol{u}_1 = \lambda_1\boldsymbol{u}_1$, we obtain the eigenvector:

$$\boldsymbol{u}_1 = \begin{bmatrix} 0.7124 \\ 0.7017 \end{bmatrix}.$$

Solving the 2×2 systems of equations $\boldsymbol{A}\boldsymbol{u}_2 = \lambda_2\boldsymbol{u}_2$, we obtain the eigenvector:

$$\boldsymbol{u}_2 = \begin{bmatrix} -0.7017 \\ 0.7124 \end{bmatrix}.$$

Therefore, the eigenvalue decomposition of $\boldsymbol{A}$ is:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top,$$

where

$$\boldsymbol{U} = \begin{bmatrix} \boldsymbol{u}_1 & \boldsymbol{u}_2 \end{bmatrix} = \begin{bmatrix} 0.7124 & -0.7017 \\ 0.7017 & 0.7124 \end{bmatrix},$$

and

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = \begin{bmatrix} 1.98511 & 0 \\ 0 & 0.004886 \end{bmatrix}.$$

(b) If $\boldsymbol{y} = \begin{bmatrix} 1 & 1 \end{bmatrix}^{\mathrm{T}}$, determine the solution to $\boldsymbol{Ax} = \boldsymbol{y}$.
Solution.

$$\begin{aligned} \boldsymbol{x} &= \boldsymbol{A}^{-1}\boldsymbol{y} = \boldsymbol{U}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}^{\top}\boldsymbol{y} \\ &= \begin{bmatrix} 0.7124 & -0.7017 \\ 0.7017 & 0.7124 \end{bmatrix} \begin{bmatrix} 1/1.98511 & 0 \\ 0 & 1/0.004886 \end{bmatrix} \begin{bmatrix} 0.7124 & 0.7017 \\ -0.7017 & 0.7124 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} -1.02919 \\ 2.05996 \end{bmatrix} \end{aligned}$$

(c) Now let $y = \begin{bmatrix} 1.1 & 1 \end{bmatrix}^{\mathrm{T}}$ and solve $\boldsymbol{Ax} = \boldsymbol{y}$. Comment on how the solution changed.
Solution.

$$\begin{aligned} \boldsymbol{x} &= \boldsymbol{A}^{-1}\boldsymbol{y} = \boldsymbol{U}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}^{\top}\boldsymbol{y} \\ &= \begin{bmatrix} 0.7124 & -0.7017 \\ 0.7017 & 0.7124 \end{bmatrix} \begin{bmatrix} 1/1.98511 & 0 \\ 0 & 1/0.004886 \end{bmatrix} \begin{bmatrix} 0.7124 & 0.7017 \\ -0.7017 & 0.7124 \end{bmatrix} \begin{bmatrix} 1.1 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} 9.07380 \\ -8.14594 \end{bmatrix} \end{aligned}$$

Since the difference between our two choices for $\boldsymbol{y}$ varied nontrivially in the direction of the eigenvector corresponding to the smallest eigenvalue of $\boldsymbol{A}$, the solution varied wildly.

(d) Suppose we observe

$$\boldsymbol{y} = \boldsymbol{Ax} + \boldsymbol{e}$$

with $\|\boldsymbol{e}\|_2 = 1$. We form an estimate $\tilde{\boldsymbol{x}} = \boldsymbol{A}^{-1}\boldsymbol{y}$. Which vector $\boldsymbol{e}$ (over all error vectors with $\|\boldsymbol{e}\|_2 = 1$) yields the maximum error $\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_2^2$?
Solution.
Since $\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_2^2 = \|\boldsymbol{A}^{-1}\boldsymbol{e}\|_2^2$, choosing $\boldsymbol{e}$ as the unit eigenvector corresponding to the greatest eigenvalue of $\boldsymbol{A}^{-1}$ (equivalently the smallest eigenvalue of $\boldsymbol{A}$) will maximize the error. Thus, $\boldsymbol{e} = \boldsymbol{u}_2 = \begin{bmatrix} -0.7017 \\ 0.7124 \end{bmatrix}$.

(e) Which (unit) vector $\boldsymbol{e}$ yields the minimum error?
Solution.
Choosing $\boldsymbol{e}$ as the unit eigenvector corresponding to the smallest eigenvalue of $\boldsymbol{A}^{-1}$ (equivalently the largest eigenvalue of $\boldsymbol{A}$) will minimize the error. Thus, $\boldsymbol{e} = \boldsymbol{u}_1 = \begin{bmatrix} 0.7124 \\ 0.7017 \end{bmatrix}$.

(f) Suppose the components of $\boldsymbol{e}$ are independent and identically distributed (i.i.d.) Gaussian random variables:

$$\boldsymbol{e} \sim \mathrm{Normal}(\mathbf{0}, \mathbf{I}).$$

What is the mean-square error $\mathbb{E}[\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_2^2]$?
Solution.

$$\begin{aligned}
\mathbb{E}\left[\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_2^2\right] &= \mathbb{E}\left[\|\boldsymbol{A}^{-1}\boldsymbol{e}\|_2^2\right] \\
&= \mathbb{E}\left[\operatorname{trace}\left(\boldsymbol{e}^\top \boldsymbol{A}^{-\top}\boldsymbol{A}^{-1}\boldsymbol{e}\right)\right] && (\boldsymbol{e}^\top \boldsymbol{A}^{-\top}\boldsymbol{A}^{-1}\boldsymbol{e} \text{ is a scalar}) \\
&= \mathbb{E}[\operatorname{trace}\left(\boldsymbol{A}^{-\top}\boldsymbol{A}^{-1}\boldsymbol{e}\boldsymbol{e}^\top\right)] && (\operatorname{trace}(\boldsymbol{AB}) = \operatorname{trace}(\boldsymbol{BA})) \\
&= \operatorname{trace}\left(\boldsymbol{A}^{-\top}\boldsymbol{A}^{-1}\mathbb{E}\left[\boldsymbol{e}\boldsymbol{e}^\top\right]\right) && (\mathbb{E}[\operatorname{trace}(\boldsymbol{A})] = \operatorname{trace}(\mathbb{E}[\boldsymbol{A}])) \\
&= \operatorname{trace}\left(\boldsymbol{A}^{-\top}\boldsymbol{A}^{-1}\mathbf{I}\right) && (\mathbb{E}\left[\boldsymbol{e}\boldsymbol{e}^\top\right] = \operatorname{Cov}(\boldsymbol{e}) = \mathbf{I}) \\
&= \operatorname{trace}\left(\boldsymbol{A}^{-\top}\boldsymbol{A}^{-1}\right) \\
&= \operatorname{trace}\left(\boldsymbol{U}\left(\boldsymbol{\Lambda}^{-1}\right)^2\boldsymbol{U}^\top\right) && (\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top) \\
&= \operatorname{trace}\left(\left(\boldsymbol{\Lambda}^{-1}\right)^2\boldsymbol{U}^\top\boldsymbol{U}\right) && (\text{again, } \operatorname{trace}(\boldsymbol{AB}) = \operatorname{trace}(\boldsymbol{BA})) \\
&= \operatorname{trace}\left(\left(\boldsymbol{\Lambda}^{-1}\right)^2\right) && (\boldsymbol{U} \text{ is an orthogonal matrix}) \\
&= \frac{1}{\lambda_1^2} + \frac{1}{\lambda_2^2} = 41888.5865.
\end{aligned}$$

(g) Verify your answer to part (f) in MATLAB/Python by taking $\boldsymbol{Ax} = \begin{bmatrix}1 & 1\end{bmatrix}^{\mathrm{T}}$, and then generating $10,000$ different realizations of $\boldsymbol{e}$ using the `randn` command, and then averaging the results. Turn in your code and the results of your computation.
Solution.
Please see script "p4g.py" for the code. The console output is: the mean-squared error = 41804.301073691866.

5. (a) Let $\boldsymbol{A}$ be a $N \times N$ symmetric matrix. Show that[3]

$$\operatorname{trace}(\boldsymbol{A}) = \sum_{n=1}^{N} \lambda_n,$$

where the $\{\lambda_n\}$ are the eigenvalues of $\boldsymbol{A}$.
Solution.
Observe the following property of the trace:

$$\operatorname{trace}(\boldsymbol{AB}) = \sum_{\ell}\sum_{k} A[\ell,k]B[k,\ell] = \sum_{k}\sum_{\ell} B[k,\ell]A[\ell,k] = \operatorname{trace}(\boldsymbol{BA})$$

Applying this property to the eigenvalue decomposition of $\boldsymbol{A}$ yields the desired answer:

$$\operatorname{trace}(\boldsymbol{A}) = \operatorname{trace}\left(\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathrm{T}}\right) = \operatorname{trace}\left(\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U}\boldsymbol{\Lambda}\right) = \operatorname{trace}(\boldsymbol{\Lambda}) = \sum_{n=1}^{N}\lambda_n.$$

[3]The trace of a (square) matrix is the sum of the elements on the diagonal: $\operatorname{trace}(\boldsymbol{A}) = \sum_{n=1}^{N} A[n,n]$.

(b) Now let $\boldsymbol{A}$ be an arbitrary $M \times N$ matrix. Recall the definition of the Frobenius norm:

$$\|\boldsymbol{A}\|_F = \left(\sum_{m=1}^{M} \sum_{n=1}^{N} |A[m,n]|^2 \right)^{1/2}.$$

Show that

$$\|\boldsymbol{A}\|_F^2 = \operatorname{trace}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}) = \sum_{r=1}^{R} \sigma_r^2,$$

where R is the rank of $\boldsymbol{A}$ and the $\{\sigma_r\}$ are the singular values of $\boldsymbol{A}$.

Solution.

Observe that the (i,i)-th element of $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ is

$$\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)[i,i] = \sum_{m=1}^{M} |A[m,i]|^2$$

Consequently,

$$\operatorname{trace}\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right) = \sum_{m=1}^{M} \sum_{n=1}^{N} |A[m,n]|^2 = \|\boldsymbol{A}\|_F^2.$$

Also, observe that applying the SVD of $\boldsymbol{A}$ yields the eigenvalue decomposition of $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A} = \boldsymbol{V\Sigma U}^{\mathrm{T}}\boldsymbol{U\Sigma V}^{\mathrm{T}} = \boldsymbol{V\Sigma}^2\boldsymbol{V}^{\mathrm{T}}$. Applying the result from part (a), we know that

$$\operatorname{trace}\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right) = \sum_{r=1}^{R} \sigma_r^2$$

(c) The *operator norm* (sometimes called the *spectral norm*) of an $M \times N$ matrix is

$$\|\boldsymbol{A}\| = \max_{\boldsymbol{x}\in\mathbb{R}^N,\ \|\boldsymbol{x}\|_2=1} \|\boldsymbol{Ax}\|_2.$$

(This matrix norm is so important, it doesn't even require a designation in its notation — if somebody says "matrix norm" and doesn't elaborate, this is what they mean.) Show that

$$\|\boldsymbol{A}\| = \sigma_1,$$

where σ_1 is the largest singular value of $\boldsymbol{A}$. For which $\boldsymbol{x}$ does

$$\|\boldsymbol{Ax}\|_2 = \|\boldsymbol{A}\| \cdot \|\boldsymbol{x}\|_2 \ \ ?$$

Solution.

$$\begin{aligned}
\|\boldsymbol{A}\| &= \max_{\|\boldsymbol{x}\|_2=1} \|\boldsymbol{Ax}\|_2 \\
&= \max_{\|\boldsymbol{x}\|_2=1} \sqrt{\boldsymbol{x}^\top\boldsymbol{A}^\top\boldsymbol{Ax}} \\
&= \sqrt{\max_{\|\boldsymbol{x}\|_2=1} \boldsymbol{x}^\top\boldsymbol{A}^\top\boldsymbol{Ax}} \\
&= \sqrt{\lambda_{\max}(\boldsymbol{A}^\top\boldsymbol{A})} \\
&= \sigma_1,
\end{aligned}$$

where $\lambda_{\max}(\boldsymbol{A}^\top \boldsymbol{A})$ is the largest eigenvalue of $\boldsymbol{A}^\top \boldsymbol{A}$. If $\boldsymbol{x}$ is the unit eigenvector associated with the largest eigenvalue $\sigma_1^2 = \lambda_{\max}(\boldsymbol{A}^\top \boldsymbol{A})$ of the matrix $\boldsymbol{A}^\top \boldsymbol{A}$, i.e., $\boldsymbol{A}^\top \boldsymbol{A}\boldsymbol{x} = \sigma_1^2 \boldsymbol{x}$, then we have

$$\|\boldsymbol{A}\boldsymbol{x}\|_2 = \sqrt{\boldsymbol{x}^\top \boldsymbol{A}^\top \boldsymbol{A}\boldsymbol{x}} = \sqrt{\sigma_1^2 \boldsymbol{x}^\top \boldsymbol{x}} = \sigma_1 \|\boldsymbol{x}\|_2 = \|\boldsymbol{A}\| \|\boldsymbol{x}\|_2.$$

Note that $\boldsymbol{x}$ is also the right singular vector of $\boldsymbol{A}$ corresponding to the largest singular value σ_1.

(d) Prove that $\|\boldsymbol{A}\| \leq \|\boldsymbol{A}\|_F$. Give an example of an $\boldsymbol{A}$ with $\|\boldsymbol{A}\| = \|\boldsymbol{A}\|_F$.

Solution.

Since each singular value $\sigma_r \geq 0$ and $\sigma_1^2 \leq \sum_{r=1}^{R} \sigma_r^2$, we have

$$\sigma_1 \leq \sqrt{\sum_{r=1}^{R} \sigma_r^2}$$

Combining results from part (b) and (c), we know that the equation above is equivalent to:

$$\|\boldsymbol{A}\| \leq \|\boldsymbol{A}\|_F$$

Since zero matrix or any rank-1 matrix $\boldsymbol{A}$ has $\|\boldsymbol{A}\| = \|\boldsymbol{A}\|_F$, one simple example is:

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}.$$