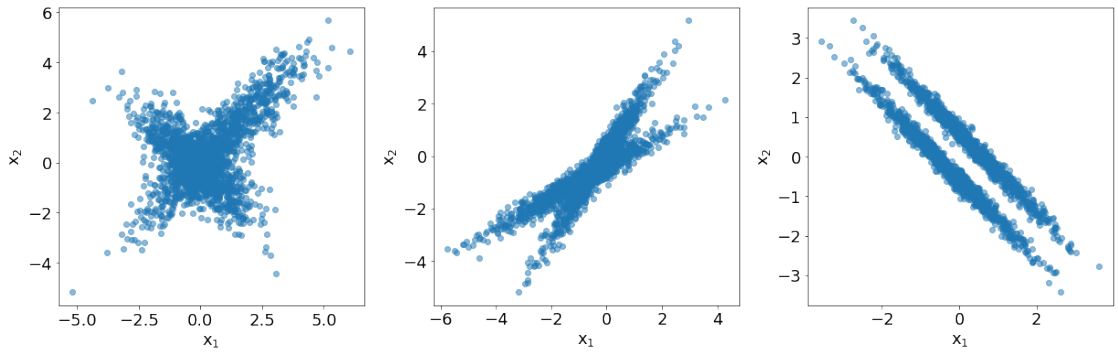
This homework is due on Thursday, November 19, 2020, at 10:59PM.

Self-grades are due on Thursday, November 26, 2020, at 10:59PM.

1 Implementation: SVD and PCA

In this problem we will implement Principal Component Analysis (PCA) using Singular Value Decomposition (SVD) in python.



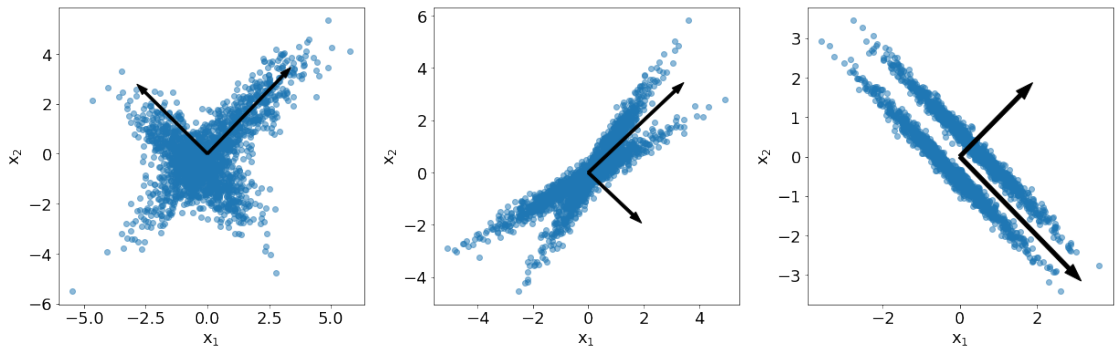
(a) Dataset 1 (b) Dataset 2 (c) Dataset 3

Figure 1: Example datasets that we will work with.

Figure 1 shows the three datasets that we will be working with. Each dataset is comprised of entries $\vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$. We will develop a procedure to calculate the principal components $\vec{v}_i$ and corresponding weights w_i for this dataset by calculating an SVD of the matrix A. Use the supplied iPython notebook to complete this problem.

a) Consider the following datasets. First, intuitively sketch the Principal Components (PCs) that best explain this dataset. You don't need to calculate the exact lengths of the principal components but their relative lengths should approximately be correct.

Solution



(a) Dataset 1 (b) Dataset 2 (c) Dataset 2

Figure 2: Approximate principal components for the two datasets.

b) We are given a set of data points $\vec{x} \in \mathbb{R}^2$. Does our data have 0 mean? If not, subtract the mean from this data and construct a new *de-meaned* matrix $\hat{A}$.

c) Explain the relationship between

i) Diagonalization of the sample covariance matrix $C = \frac{1}{N}\hat{A}^T\hat{A}$,

ii) Singular Value decomposition of $\hat{A}$, and

iii) Principal Components of $\hat{A}$.

Solution

Since the covariance matrix $\frac{1}{N}\hat{A}^T\hat{A}$ is a real, symmetric matrix, it is diagonalizable. Moreover, we can find an orthonormal set of basis vectors $\vec{v}_1, \vec{v}_2, \vec{v}_3, \ldots, \vec{v}_n$ of C. Diagonalizing the covariance matrix C with these eigenvectors, we can write

$$\frac{1}{N}\hat{A}^T\hat{A} = P\Lambda P^T, \tag{1}$$

where $P = \begin{bmatrix}\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_n\end{bmatrix}$

The SVD of $\hat{A}$ can be written as

$$\hat{A} = U\Sigma V^T \tag{2}$$

We can evaluate $\hat{A}^T\hat{A}$ using the SVD to obtain

$$\begin{aligned}\hat{A}^T\hat{A} &= \left(U\Sigma V^T\right)^T U\Sigma V^T \\ &= V\Sigma^T U^T U\Sigma V^T \\ &= V\Sigma^T\Sigma V^T\end{aligned}$$

$\Sigma^T\Sigma$ is a square, diagonal matrix and comparing this expression with Equation 1, we can say the $P = V$. Futhermore, $\Sigma^T\Sigma = N\Lambda$, or if λ_i is the i^{th} eigenvalue of C, $\sigma_i = \sqrt{N\lambda_i}$.

The principal components are given by the columns of V and the weights of the i^{th} principal component $\vec{v}_i$ is given by $\frac{\sigma_i}{\sqrt{N}}$, where σ_i is the i^{th} singular value of $\hat{A}$.

NOTE: Scaling the singular values by $\frac{1}{\sqrt{N}}$ is very important because without it, the weights increase as we increase the number of data points.

In class, we have learnt that we can calculate the SVD (and thereafter PCA) of the matrix A using the sample covariance matrix C. However, this approach is not used in practice for numerical reasons. Some of the advanced classes on numerical methods delve deeper into this topic. We will be using the inbuilt SVD-solver in python to get the SVD of our de-meaned matrix $\hat{A}$.

d) What are the dimensions of the matrix $\hat{A}$? If we were to compute the full-svd, what will be the dimensions of matrices U, Σ, and V if the SVD is written as

$$\hat{A} = U\Sigma V^T \tag{3}$$

Solution

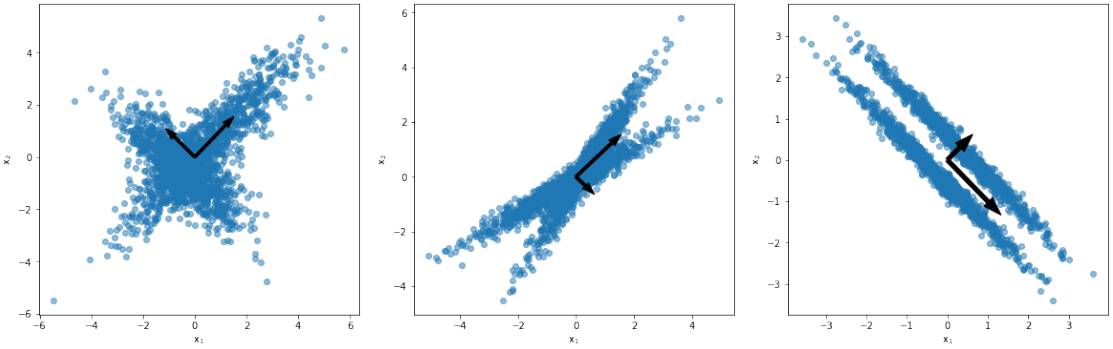
In the dataset, we have $m = 1000$ samples of data, each drawn from $n = 2$ dimensional space. This gives us a 1000×2 matrix A.

As a result of that, a full Singular Value decomposition will result in

- $U_{1000\times1000}$
- $\Sigma_{1000\times2}$
- $V_{2\times2}$

e) Compute the principal components (unit vectors) and corresponding weights for the datasets. Sketch them overlaid on the data (The weights have been scaled in the notebook to make visualization neater). Does this match what you had expected at the beginning of the problem? Comment on the differences in the principal components of the three datasets.

Solution



(a) Dataset 1 (b) Dataset 2 (c) Dataset 3

Figure 3: Weighted principal components for the datasets.

Dataset 01

$\vec{v}_1 = \begin{bmatrix} 0.70 \\ 0.72 \end{bmatrix}$, $w_1 = 1.72$.

$\vec{v}_2 = \begin{bmatrix} -0.72 \\ 0.70 \end{bmatrix}$, $w_2 = 1.12$.

In dataset, we have two blobs of data which are roughly similar in size and the direction of spread of these blobs is orthogonal. PCA captures this structure in the data very well, identifying the two main directions in which the data is spread out. Furthermore, because the blobs are roughly equal in size, we get principal component weights $w_1 = 1.72$ and $w_2 = 1.12$ which are similar.

Dataset 02

$\vec{v}_1 = \begin{bmatrix} 0.71 \\ 0.71 \end{bmatrix}$, $w_1 = 1.75$.

$\vec{v}_2 = \begin{bmatrix} -0.71 \\ 0.71 \end{bmatrix}$, $w_2 = 0.46$.

In this dataset, we again have two blobs but they are not orthogonal to each other. Here, the direction that maximally explains the variance in the data is not aligned with either of the blobs but the direction that is maximally aligned to both the blobs. The second principal component is orthogonal to the first, but since we have a much larger spread in the data along the first direction, we get a bigger difference in the principal component weights with $w_1 = 1.75$ and $w_2 = 0.46$.

Dataset 03

$\vec{v}_1 = \begin{bmatrix} 0.70 \\ -0.71 \end{bmatrix}$, $w_1 = 1.43$.

$\vec{v}_2 = \begin{bmatrix} 0.71 \\ 0.71 \end{bmatrix}$, $w_2 = 0.44$.

In the final dataset, we have 2 parallel blobs and similar to dataset 2, the direction that maximally explains the variance in the data is the common direction. Furthermore, the weight of the second principal component is driven by the distance between the two blobs.

2 The Moore-Penrose Pseudoinverse for "Fat" Matrices

Suppose that we have a set of linear equations described as $A\vec{x} = \vec{y}$. If A is invertible, we know that the solution is $\vec{x} = A^{-1}\vec{y}$. However, what if A is not a square matrix? In EE16A, you saw how this problem could be approached for tall matrices A where it really wasn't possible to find a solution that exactly matches all the measurements. The linear least-squares solution gives us a reasonable answer that asks for the "best" match in terms of reducing the norm of the error vector.

This problem deals with the other case — when the matrix A is short and fat. In this case, there are generally going to be lots of possible solutions — so which should we choose and why? We will walk you through the **Moore-Penrose pseudoinverse** that generalizes the idea of the matrix inverse and is derived from the singular value decomposition.

a) Suppose that you have the following matrix.

$$A = \begin{bmatrix} 1 & -1 & 1 \\ 1 & 1 & -1 \end{bmatrix}$$

Calculate the full SVD decomposition of A. That is to say, calculate U, Σ, V, such that

$$A = U\Sigma V^T, \text{ where } U \text{ and } V \text{ are unitary matrices.}$$

Leave all work in exact form, not decimal.

Note: Do NOT use a computer to calculate the SVD.

Solution

$$AA^\top = \begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix}$$

which has characteristic polynomial $\lambda^2 - 6\lambda + 8 = 0$, giving the eigenvalues 4 and 2. Solving $A\vec{v} = \lambda_i \vec{v}$ produces eigenvectors $\begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}^\top$ and $\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}^\top$ associated with eigenvalues 4 and 2 respectively.

The singular values are the square roots of the eigenvalues of $AA^\top$, so

$$\Sigma = \begin{bmatrix} 2 & 0 & 0 \\ 0 & \sqrt{2} & 0 \end{bmatrix}$$

and

$$U = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}.$$

We can then solve for the $\vec{v}$ vectors using $A^\top \vec{u}_i = \sigma_i \vec{v}_i$, giving $\vec{v}_1 = \begin{bmatrix} 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}^\top$ and $\vec{v}_2 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^\top$. The last $\vec{v}$ must be orthonormal to the other two, so we can pick $\begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}^\top$.

The SVD is:

$$A = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & \sqrt{2} & 0 \end{bmatrix} \begin{bmatrix} 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 1 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

b) Let us think about what the SVD does. Let us look at matrix A acting on some vector $\vec{x}$ to give the result $\vec{y}$. We have

$$A\vec{x} = U\Sigma V^T \vec{x} = \vec{y}.$$

Observe that U and V^T are unitary matrices, so they cannot change the norm of the input vector while Σ scales the input vector. We will try to "reverse" these operations one at a time and then put them together.

If U performs some transformation on the vector $\left(\Sigma V^T\right)\vec{x}$, what is the inverse of U that cancels its effect.

Solution

By orthonormality, we know that $U^T U = UU^T = I$. Therefore, U^T undoes the transformation.

c) Recall that Σ has the same dimensions as A (m by n with $m < n$). Now find some *diagonal* $\widetilde{\Sigma}$ that "inverts" Σ. That is $\widetilde{\Sigma}\Sigma = \tilde{I}_{n\times n}$ where $\tilde{I}_{n\times n}$ given below is a block diagonal identity matrix with the top left block being an m by m identity matrix. You may assume that A has rank m. By diagonal for some non square matrix, we mean that the top (for thin matrix) or left (for wide matrix) square submatrix to be diagonal and zeros for the remaining submatrix.

$$\tilde{I}_{n\times n} = \begin{bmatrix} I_{m\times m} & 0_{m\times(n-m)} \\ 0_{(n-m)\times m} & 0_{(n-m)\times(n-m)} \end{bmatrix}$$

Solution

If you observe the equation:

$$\Sigma\vec{x} = \vec{y}, \tag{4}$$

you can see that $\sigma_i x_i = y_i$ for $i = 0, ..., m-1$, which means that to obtain x_i from y_i, we need to multiply y_i by $\frac{1}{\sigma_i}$. For any $i > m-1$, the information in x_i is lost by multiplying with 0. Therefore, the reasonable guess for x_i is 0 in this case. That's why we padded 0s in the bottom of $\widetilde{\Sigma}$ given below:

$$\text{If } \Sigma = \begin{bmatrix} \sigma_0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_1 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \sigma_{m-1} & 0 & \cdots & 0 \end{bmatrix}, \text{ then } \widetilde{\Sigma} = \begin{bmatrix} \frac{1}{\sigma_0} & 0 & \cdots & 0 \\ 0 & \frac{1}{\sigma_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\sigma_{m-1}} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

In cases with

d) What is the inverse of V^T that cancels its effect.

Solution

By orthonormality, we know that $V^T V = VV^T = I$. Therefore, V undoes the transformation.

e) Try to use the previous parts to derive an "inverse" (which we will use $A^\dagger$ to denote). That is to say,

$$\vec{x} = A^\dagger \vec{y}.$$

The reason why the word inverse is in quotes (or why this is called a pseudo-inverse) is because we're ignoring the "divisions" by zero.

Solution

We can use the matrices we derived above to "undo" the effect of A and get the required solution. Of course, nothing can possibly be done for the information that was destroyed by the nullspace of A — there is no way to recover any component of the true $\vec{x}$ that was in the nullspace of A. However, we can get back everything else.

$$\begin{aligned} \vec{y} &= A\vec{x} = U\Sigma V^T\vec{x} \\ U^T\vec{y} &= \Sigma V^T\vec{x} && \text{Undoing the transformation by } U \\ \widetilde{\Sigma}U^T\vec{y} &= V^T\vec{x} && \text{Unscaling by } \widetilde{\Sigma} \\ V\widetilde{\Sigma}U^T\vec{y} &= \vec{x} && \text{Undoing the transformation by } V \end{aligned}$$

Therefore, we have $A^\dagger = V\widetilde{\Sigma}U^T$, where $\widetilde{\Sigma}$ is given in part (c).

f) Use $A^\dagger$ to solve for $\vec{x}$ in the following system of equations.

$$\begin{bmatrix} 1 & -1 & 1 \\ 1 & 1 & -1 \end{bmatrix} \vec{x} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$$

Solution

From the above, we have the solution given by:

$$\begin{aligned}\vec{x} &= A^{+}\vec{y} = V\widetilde{\Sigma}U^{T}\vec{y} \\ &= \begin{bmatrix} 0 & 1 & 0 \\ -\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{\sqrt{2}} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \\ &= \begin{bmatrix} 3 \\ \frac{1}{2} \\ -\frac{1}{2} \end{bmatrix}\end{aligned}$$

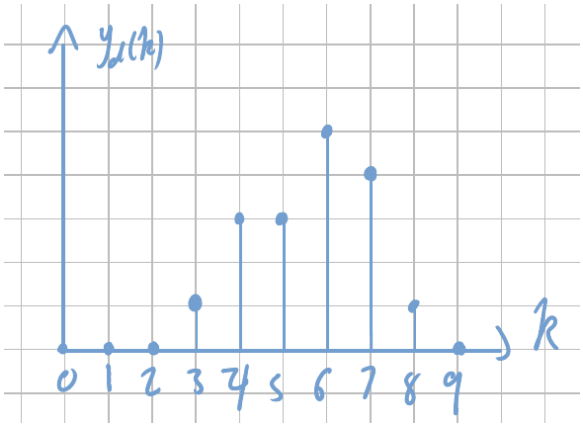
Therefore, the solution to the system of equations is:

$$\vec{x} = \begin{bmatrix} 3 \\ \frac{1}{2} \\ -\frac{1}{2} \end{bmatrix}$$

3 Piecewise Linear interpolation

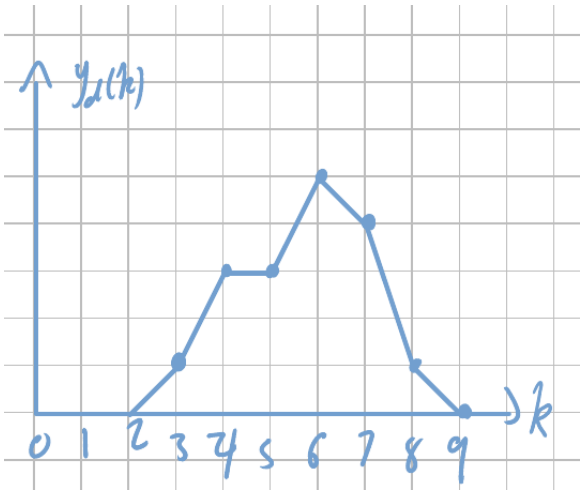
Suppose we have a discrete-time signal $y_d(k)$ that we would like to interpolate.

We will assume that the discrete-time signal is of *finite duration*– that is, the signal "begins" at some time k_1 and "ends" at some time k_2, and we can assume $y_d(k) = 0$ for $k < k_1$ and $k > k_2$. For example, if our discrete-time signal looked like this:



then we would have $k_1 = 3, k_2 = 8$.

One of the simplest ways to interpolate this signal would be to simply connect the points of the discrete-time signal with straight lines. If we were to interpolate the signal above this way, we would get this:



As you can see, the interpolated signal $y(t)$ is a straight line over intervals of the form $[k, k+1]$ for all integers k, although the entire function $y(t)$ is not itself a straight line. For this reason, we call this $y(t)$ the *piecewise linear* (PWL) interpolation of the discrete-time signal $y_d(k)$.

Although we've just described the PWL interpolation in an intuitive and somewhat *ad hoc* way, it turns out that the PWL interpolation can be expressed as a basis function interpolation. **in this problem, you will show how this is true**.

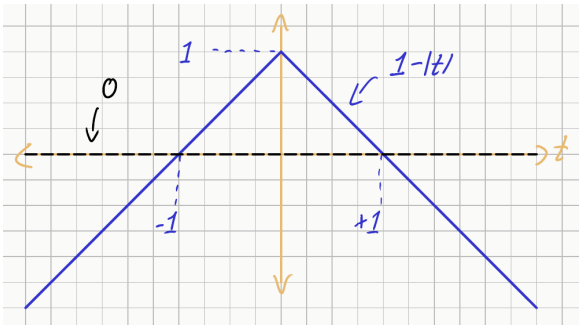
a) Consider the function $\phi(t)$ defined as,

$$\phi(t) = \begin{cases} 1 - |t|, & t \in [-1, 1] \\ 0, & \text{otherwise} \end{cases}. \tag{5}$$

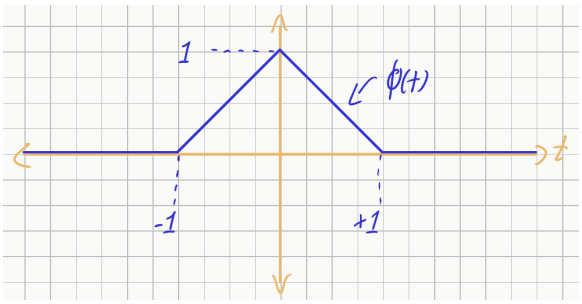
Sketch $\phi(t-k)$ for some arbitrary integer k. You may choose a specific integer for k in your sketch ($k = 3$ perhaps), or you may keep the sketch entirely in terms of k. The graph in the solution will be drawn in terms of k, so we encourage you to try it that way.

Solution

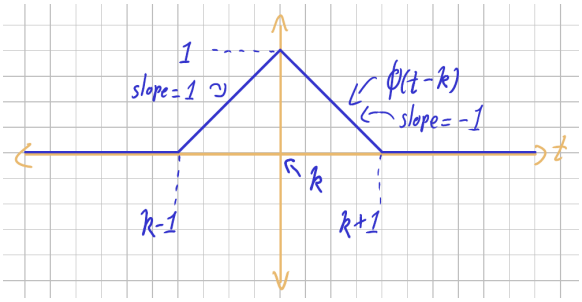
First, let's try sketching $\phi(t)$, just to ge our bearings straight. $\phi(t)$ is a piecewise combination of the following two functions:



Connecting these two functions over the domains defined in $\phi(t)$, we get the following sketch:



Now we can get the sketch that we want, namely $\phi(t-k)$, by substituting $t \mapsto t-k$:



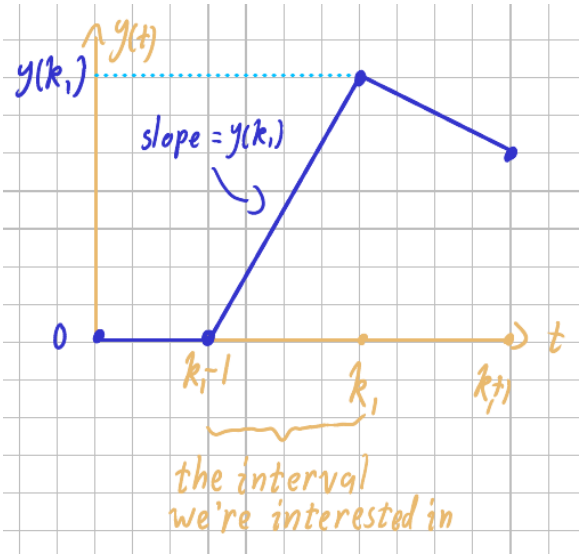
b) We'll be using the function $\phi(t)$ as the basis function for the PWL interpolation.

We will begin our analysis right at the beginning of the signal. Write the basis function and coefficient that captures the line of $y(t)$ from $t = k_1 - 1$ to $t = k_1$. That is to say, find real number α and integer p such that,

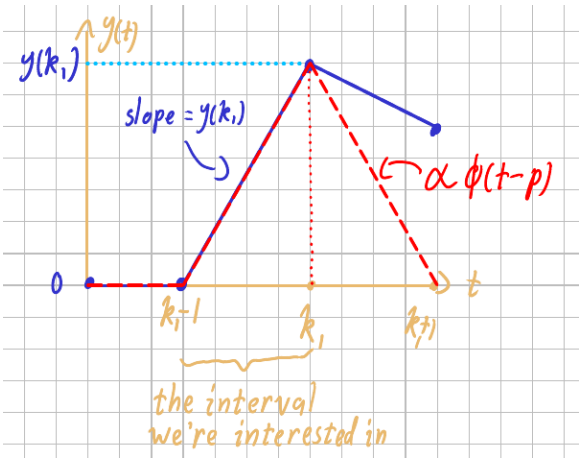
$$y(t) = \alpha\phi(t-p) \text{ for } t \in [k_1 - 1, k_1]$$

Solution

Recall that $y_d(k_1 - 1) = 0$ by assumption, so $y(k_1 - 1) = 0$, too. This means that the beginning of our interpolation will look like this:



Now, what we're looking for is a scaled, shifted version of the basis function $\phi(t)$ that matches up with $y(t)$ over the interval $[k_1 - 1, k_1]$. We can find it graphically, like so:



So now we can see what the scaled and shifted basis function looks like, but what about the actual α and p? Well, in order for the slopes to match up, we must take $\alpha = y_d(k_1)$. Then, based on where the "peak" of the basis function ended up, we must take $p = k_1$.

c) Now, consider any integer $k^\star$such that $k_1 < k^\star < k_2$. Over the interval $[k^\star, k^\star + 1]$, the interpolated signal $y(t)$ is a straight line. What is the equation of this line? In other words, **find real numbers m and b such that**

$$y(t) = mt + b, \text{ over the interval } [k^\star, k^\star + 1]. \tag{6}$$

Solution

Since $y(k^\star) = y_d(k^\star)$ and $y(k^\star + 1) = y_d(k^\star + 1)$, we can just find the slope as

$$\text{slope} = \frac{\text{rise}}{\text{run}} = \frac{y_d(k^\star + 1) - y_d(k^\star)}{1} = y_d(k^\star + 1) - y_d(k^\star) = m, \tag{7}$$

and we can find the intercept b by evaluating $mt + b$ at $t = k^\star$ and solving for b; that is,

$$y(k^\star) = y_d(k^\star) = (y_d(k^\star + 1) - y_d(k^\star))k^\star + b, \tag{8}$$

giving

$$b = y_d(k^\star) - k^\star(y_d(k^\star + 1) - y_d(k^\star)). \tag{9}$$

In summary, we have

$$m = y_d(k^\star + 1) - y_d(k^\star) \tag{10}$$
$$b = y_d(k^\star) - k^\star(y_d(k^\star + 1) - y_d(k^\star)). \tag{11}$$

d) Consider the function

$$g(t) = y_d(k^\star)\phi(t - k^\star) + y_d(k^\star + 1)\phi(t - (k^\star + 1))$$

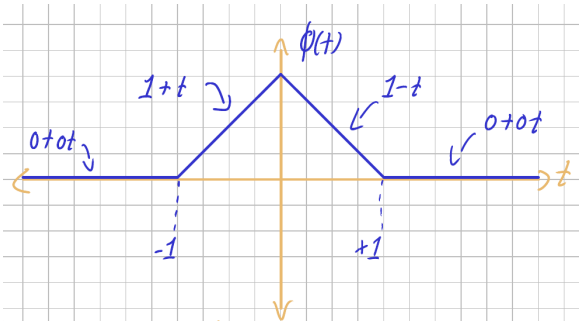
This function is also a straight line over the interval $[k^\star, k^\star + 1]$. What is the equation of the line over this region? Write it again in the form

$$y(t) = mt + b, \text{ over the interval } [k^\star, k^\star + 1]. \tag{12}$$

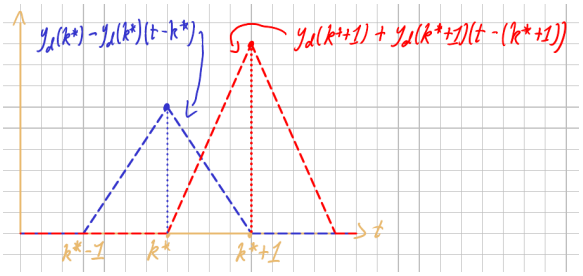
This should match your previous answer.

Solution

Consider how $\phi(t)$ itself is a piecewise linear function. In fact, it will be helpful to express it explicitly in this way:



With this in mind, let's sketch the two terms of $g(t)$ separetely, on the same axis:



Based on this sketch, we can write the part of $g(t)$ in the interval $[k^\star, k^\star + 1]$ as

$$g(t) = y_d(k^\star) - y_d(k^\star)(t - k^\star) + y_d(k^\star + 1) + y_d(k^\star + 1)(t - (k^\star + 1)) \tag{13}$$
$$= (y_d(k^\star + 1) - y_d(k^\star))t + y_d(k^\star) - k^\star(y_d(k^\star + 1) - y_d(k^\star)). \tag{14}$$

This gives

$$m = y_d(k^\star + 1) - y_d(k^\star) \tag{15}$$
$$b = y_d(k^\star) - k^\star(y_d(k^\star + 1) - y_d(k^\star)). \tag{16}$$

which is indeed the same as the previous part.

e) Given what you've shown in the previous parts, we can now express the PWL interpolation of $y_d(k)$ as a sum of shifted ϕ functions. Find the coefficients α_k such that

$$y(t) = \sum_{k=-\infty}^{\infty} \alpha_k \phi(t - k). \tag{17}$$

hint: remember, our goal is to show that the PWL interpolation is a basis function interpolation *with basis function ϕ. In that sense, the α_k have already been chosen (consult your notes and the relavent discussion handout), and you need only verify that they are correct.*

Solution

First, let's take the hint. We want to show that the PWL interpolation is the basis function interpolation of $y_d(k)$ with the basis function ϕ. Recall that, for any basis function ψ, the basis function interpolation is

$$y(t) = \sum_{k=-\infty}^{\infty} y_d(k)\psi(t - k), \tag{18}$$

which suggests that we take $\alpha_k = y_d(k)$. To show that this is a good choice for α_k, we must show that

$$y(t) = \sum_{k=-\infty}^{\infty} y_d(k)\phi(t - k), \tag{19}$$

by showing that the equality holds over each interval of the form $[k, k + 1]$ with k an integer. Think about it this way: the union of all such intervals is the entire real line, so if the equality holds for each of those intervals, then it holds over the entire real line too, and that's what we want to show.

With that said, let's consider a specific interval, say $[k^\star, k^\star+1]$. Over this interval, the summation simplifies quite a bit: for all $k < k^\star$ and $k > k^\star + 1$ we have $\phi(t) = 0$ for $t \in [k^\star, k^\star + 1]$, so

$$y(t) = \sum_{k=-\infty}^{\infty} y_d(k)\phi(t - k) = y_d(k^\star)\phi(t - k^\star) + y_d(k^\star + 1)\phi(t - (k^\star + 1)). \tag{20}$$

over the interval $[k^\star, k^\star + 1]$. From parts (c) and (d) we know that

$$y(t) = y_d(k^\star)\phi(t - k^\star) + y_d(k^\star + 1)\phi(t - (k^\star + 1)) \tag{21}$$

over the interval $[k^\star, k^\star + 1]$. Therefore, it's also true that

$$y(t) = \sum_{k=-\infty}^{\infty} y_d(k)\phi(t - k), \tag{22}$$

over the interval $[k^\star, k^\star + 1]$. This holds for any choice of $k^\star$, so what we've really shown is that this equation holds for *all* intervals of the form $[k, k + 1]$. By our reasoning from before, this means that

$$y(t) = \sum_{k=-\infty}^{\infty} y_d(k)\phi(t - k), \tag{23}$$

for all t, which is what we wanted to show.

4 Lagrange Interpolation by Polynomials

Given n distinct points and the corresponding sampling of a function $f(x)$, $(x_i, f(x_i))$ for $0 \leq i \leq n-1$, the Lagrange polynomial interpolation is the polynomial of the least degree that passes through all of the given points.

Given n distinct points and the corresponding evaluations, $(x_i, f(x_i))$ for $0 \leq i \leq n-1$, the Lagrange polynomial interpolation is the $n-1^{th}$ degree polynomial

$$P(x) = \sum_{i=0}^{i=n-1} f(x_i)L_i(x),$$

where

$$L_i(x) = \prod_{j=0, j\neq i}^{j=n-1} \frac{(x - x_j)}{(x_i - x_j)} = \frac{(x - x_0)}{(x_i - x_0)} \cdots \frac{(x - x_{i-1})}{(x_i - x_{i-1})} \frac{(x - x_{i+1})}{(x_i - x_{i+1})} \cdots \frac{(x - x_{n-1})}{(x_i - x_{n-1})}. \tag{24}$$

Here is an example: for two data points, $(x_0, f(x_0)) = (0, 4)$, $(x_1, f(x_1)) = (-1, -3)$, we have

$$L_0(x) = \frac{x - x_1}{x_0 - x_1} = \frac{x - (-1)}{0 - (-1)} = x + 1$$

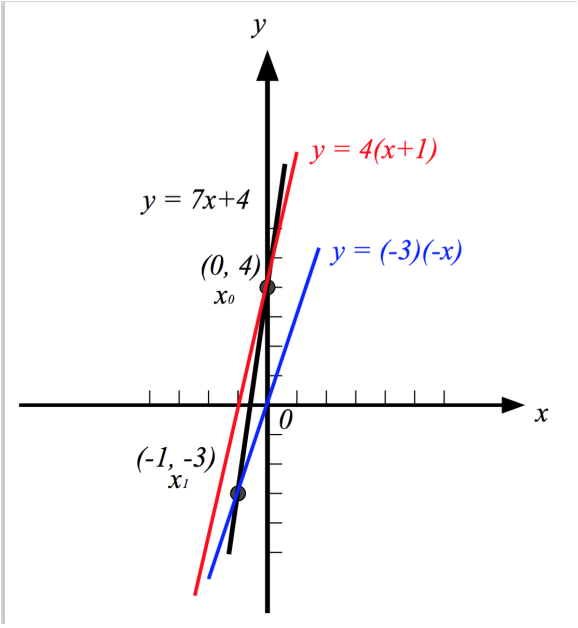
and

$$L_1(x) = \frac{x - x_0}{x_1 - x_0} = \frac{x - (0)}{(-1) - (0)} = -x.$$

Then

$$P(x) = f(x_0)L_0(x) + f(x_1)L_1(x) = 4(x + 1) + (-3)(-x) = 7x + 4.$$

We can sketch those equations on the 2D plane as follows:



In the figure above, the red line is the 0^{th} interpolating polynomial L_0 weighted by the 0^{th} function values $f(x_0)$, $y = f(x_0)L_0 = 4(x+1)$. The blue line is the 1^{st} interpolating polynomial L_1 weighted by the 1^{st} function values $f(x_1)$, $y = f(x_1)L_1 = (-3)(-x) = 3x$. The black line is the interpolated signal, $P(x) = 7x + 4$.

a) Before we find the Lagrange interpolation, let us first use interpolation by global polynomials so we can verify our Lagrange interpolation results. Using the polynomial function basis $\{1, x, x^2, \cdots x^{n-1}\}$, the interpolation problem can be cast into finding the coefficients $a_0, a_1, a_2, \cdots, a_{n-1}$ of the function

$$g(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1}x^{n-1}$$

such that $g(x_i) = f(x_i)$ for n samples of a function $(x_i, f(x_i))$ with $i \in \{0, 1, 2, ..., n-1\}$.

Given three data points, $(2, 3)$, $(0, -1)$, **and** $(-1, -6)$, **find a polynomial** $g(x) = a_2x^2 + a_1x + a_0$ **fitting the three points using global polynomial interpolation. Is this polynomial unique?** That is, is it the only second degree polynomial that fits this data?

It is computationally expensive to do this process for large numbers of points, which is why we use the Lagrange interpolation method.

Solution

Plugging in the three data points into $f(x) = a_2x^2 + a_1x + a_0$, we get:

$$\begin{bmatrix} 1 & 2 & 4 \\ 1 & 0 & 0 \\ 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 3 \\ -1 \\ -6 \end{bmatrix}$$

Solving these equations for the coefficients a_2, a_1 and a_0, we get $g(x) = -x^2 + 4x - 1$.

This polynomial is the unique degree 2 polynomial defined by these three distinct points.

b) The set of Lagrange polynomials $\{L_i(x)\},\ i \in \{0, 1, 2, ..., n-1\}$ is a new function basis for the subspace of degree $n-1$ or lower polynomials. **Find the $L_i(x)$ given by Eq. 24 corresponding to the three sample points in (a).** Show your work.

Solution

Assume $x_0 = 2$, $x_1 = 0$ and $x_2 = -1$, each corresponding $L_i(x)$ is:

$$L_0(x) = \frac{(x-x_1)}{(x_0-x_1)}\frac{(x-x_2)}{(x_0-x_2)} = \frac{(x-0)}{(2-0)}\frac{(x-(-1))}{(2-(-1))} = \frac{x^2+x}{6}$$

$$L_1(x) = \frac{(x-x_0)}{(x_1-x_0)}\frac{(x-x_2)}{(x_1-x_2)} = \frac{(x-2)}{(0-2)}\frac{(x-(-1))}{(0-(-1))} = \frac{x^2-x-2}{-2}$$

$$L_2(x) = \frac{(x-x_0)}{(x_2-x_0)}\frac{(x-x_1)}{(x_2-x_1)} = \frac{(x-2)}{(-1-(2))}\frac{(x-0)}{(-1-0)} = \frac{x^2-2x}{3}$$

c) $P(x)$ is the sum of the Lagrange polynomials weighted by the function value at the corresponding points, giving the Lagrange interpolation of the given points. **Find the Lagrange polynomial interpolation $P(x)$ that goes through the three points in (a). Compare the result to the global polynomial interpolation of the same points, which you calculated in (a). Are they different from each other? Why or why not? Reason using the degree of the polynomials.**

Solution

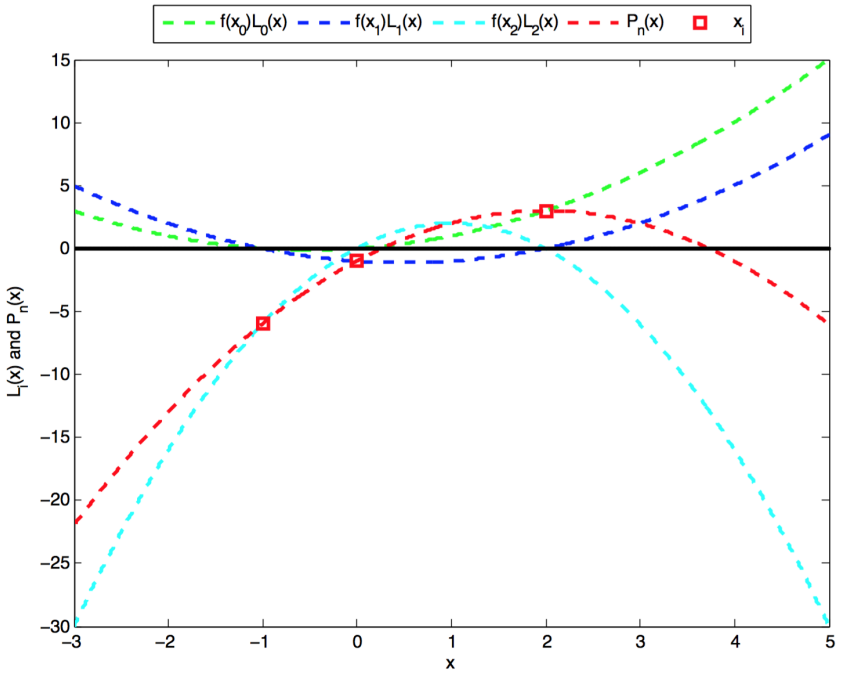
$$P(x) = f(x_0)L_0(x) + f(x_1)L_1(x) + f(x_2)L_2(x) = 3 \times \frac{x^2+x}{6} + (-1) \times \frac{x^2-x-2}{-2} + (-6) \times \frac{x^2-2x}{3}$$

Therefore, $P(x) = -x^2 + 4x - 1$.

The answer is the same as the one we computed in (a). For these three points, we cannot have a polynomial of less than 2 degree fitting all of them. Lagrange interpolation must construct the least degree ($n = 2$) polynomial passing the three points, which must be unique.

d) **Plot $P(x)$ and each $f(x_i)L_i(x)$.** You can use a plotting utility (e.g. matplotlib) and or plot by hand.

Solution



e) **Show that $P(x_i) = f(x_i)$ for all x_i.** That is, show that the Lagrange interpolation passes through all given data points. Show this symbolically in the general case, not just for the example above.

Solution

For each x_i, the corresponding $L_i(x)$ is $L_i(x) = \prod_{j=0, j\neq i}^{j=n-1} \frac{(x - x_j)}{(x_i - x_j)}$. Plugging in x_i, we get

$$L_i(x_i) = \prod_{j=0; j\neq i}^{j=n-1} \frac{(x_i - x_j)}{(x_i - x_j)} = 1.$$

For any other $L_m(x)$, where $m \neq i$, plug in x_i, $L_m(x_i) = \prod_{j=0, j\neq m}^{j=n-1} \frac{(x_i - x_j)}{(x_m - x_j)}$. Because $m \neq i$, $L_m(x_i)$ must have this term $\frac{(x_i - x_i)}{(x_m - x_i)} = 0$. Hence all $L_m(x_i)$ must be zero.

Therefore, $P(x_i) = \sum_{k=0}^{k=n-1} f(x_k)L_k(x_i) = f(x_i) \cdot 1 = f(x_i)$.

5 Homework Process and Study Group

Citing sources and collaborators are an important part of life, including being a student!
We also want to understand what resources you find helpful and how much time homework is taking, so we can change things in the future if possible.

a) **What sources (if any) did you use as you worked through the homework?**

b) **If you worked with someone on this homework, who did you work with?**
List names and student ID's. (In case of homework party, you can also just describe the group.)

c) **Roughly how many total hours did you work on this homework?**

d) **Do you have any feedback on this homework assignment?**