I. Vector Spaces and Linear Representations

A first look at linear representations

In the first section of this course, we will develop a mathematical framework for **representing** and **approximating** functions. Just so we are all clear on the terminology, a *function* is a mapping from $\mathbb{R}^d$ to $\mathbb{R}$, which we write $\boldsymbol{f} : \mathbb{R}^d \to \mathbb{R}$.

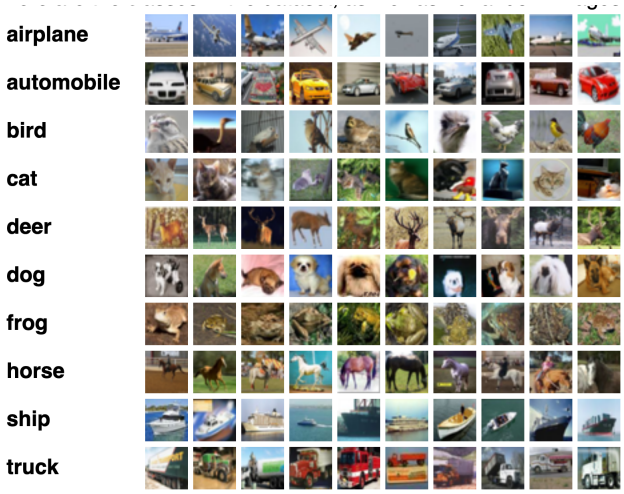
Many problems in machine learning (and statistics, and signal processing, and imaging, ...) can be abstracted as fitting a function to a series of data points. That is, we are given pairs of points $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y \in \mathbb{R}$, and want to find a function $f : \mathbb{R}^d \to \mathbb{R}$ such that

$$f(\boldsymbol{x}_i) \approx y_i, \quad \text{for all} \quad i = 1, \ldots, n.$$

The idea is that once we find such an $\boldsymbol{f}$, we can evaluate it at any $\boldsymbol{x}$ we want.

Examples:

1. Suppose that the $\boldsymbol{x}_i$ are images with d pixels, and we take $y_i \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ to mean that the image contains a { airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck }, respectively.



A famous data for this scenario (with $d = 1024$) is 'CIFAR10'. Fitting such an $\boldsymbol{f}$ corresponds to training a classifier for this data set.

2. With

$$\boldsymbol{x}_i = \begin{bmatrix} \text{\# home team record} \\ \text{home team point differential} \\ \text{home team strengh of schedule} \\ \text{\# visit team record} \\ \text{visit team point differential} \\ \text{visit team strength of schedue} \end{bmatrix}, \quad y_i = \text{probability home team wins},$$

fitting an $\boldsymbol{f}$ to the data successfully would have obvious implications for how you might wager on future games.

3. Suppose that

$$\boldsymbol{x}_i = \begin{bmatrix} \text{current location} \\ \text{destination location} \\ \text{time of day} \end{bmatrix}, \quad y_i = \text{travel time to destination}$$

Then evaluating $\boldsymbol{f}$ gives us a way to predict how long it will take to get where we are trying to go.

Fitting a function to data is a fundamental problem that we will return to repeatedly throughout this course. But first we have an even more fundamental question to consider first. Presumably, we will be feeding these data points into some kind of computer which will then calculate a mapping $\boldsymbol{f}$ that fits the points. This begs the question:

How does a computer represent a function?

This "representation" should allow the computer to manipulate and store the function. So then let's start with another qualitative question:

What types of things can computers manipulate and store?

I would say that the answer to this second question is "long lists of numbers". So now we arrive at

How can we represent a functions, mappings from $\mathbb{R}^d$ to $\mathbb{R}$, as lists of numbers?

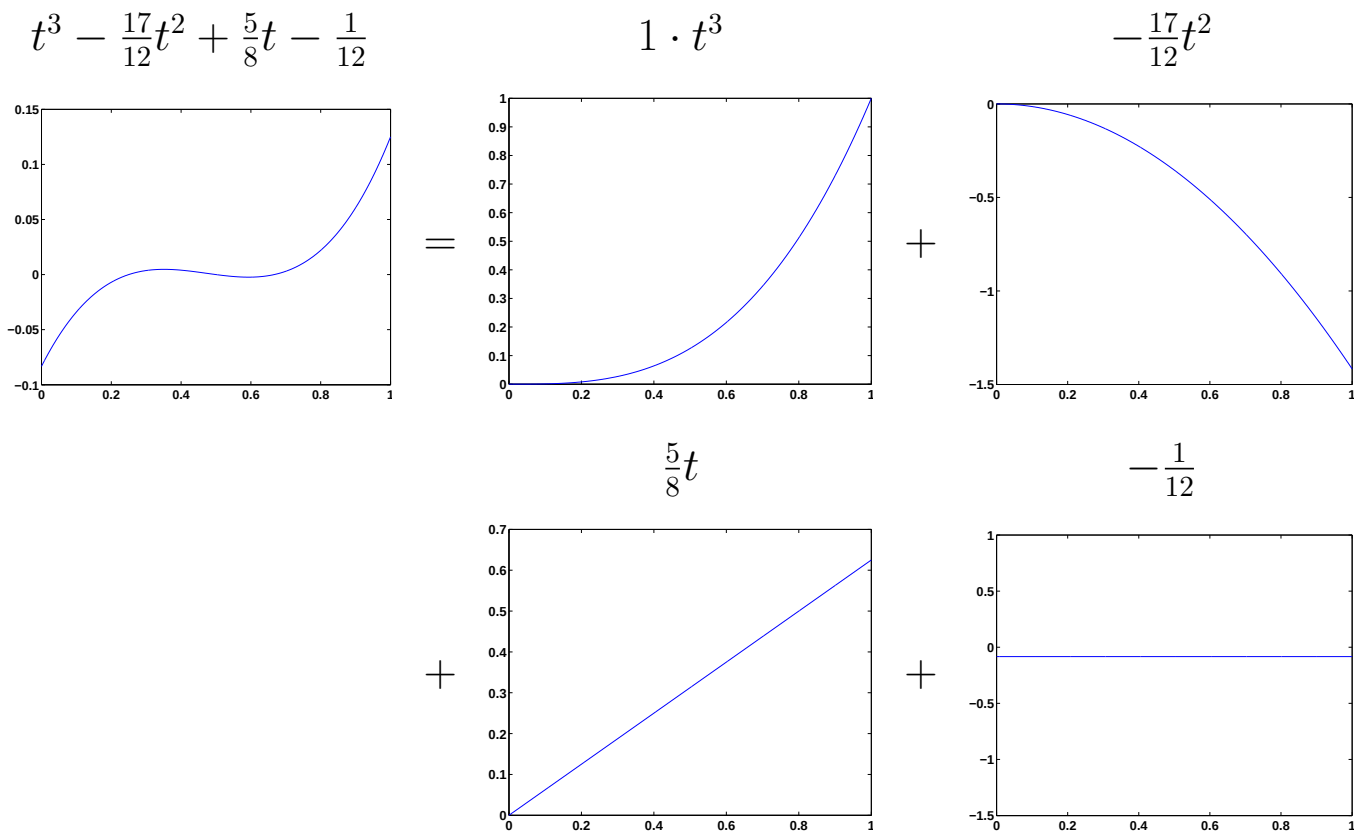
There are many ways, but let's start with a few that are very familiar.

For the rest of this note, we will work in one dimension. We will denote our covariate by t instead of x (following the convention in signal processing).

Example: Polynomials

Any m-th order polynomial can be written as a super position of the $m+1$ functions $1, t, t^2, \ldots, t^m$. (Indeed, this is pretty much the definition of polynomial.)

For example:



In this particular example, we can write

$$f(t) = \sum_{k=1}^{4} \alpha_k \, t^{k-1}, \quad \text{where} \quad \boldsymbol{\alpha} = \begin{bmatrix} -1/12 \\ 5/8 \\ -17/12 \\ 1 \end{bmatrix}.$$

More generally, we can write any mth order polynomial with a $\boldsymbol{\alpha} \in \mathbb{R}^{m+1}$ as $f(t) = \sum_{k=1}^{m} \alpha_k \, t^{k-1}$. We see that as m gets larger, this class gets richer and richer.

As you might remember from Calculus, "nice" functions can be re-written as "infinite degree" polynomials using a Taylor series. For

instance, on the interval $[-1/2, 1/2]$, we can write

$$e^t = \sum_{k=0}^{\infty} \alpha_k \cdot t^k, \quad \text{where } \alpha_k = \frac{1}{k!},$$

$$\log(1+t) = \sum_{k=0}^{\infty} \alpha_k \cdot t^k \quad \text{where } \alpha_k = \frac{(-1)^{k+1}}{k},$$

$$\sin(2\pi t) = \sum_{k=0}^{\infty} \alpha_k \cdot t^k \quad \text{where } \alpha_k = \begin{cases} \frac{(-1)^{(k+3)/2}(2\pi)^k}{k!} & k \text{ odd} \\ 0 & k \text{ even} \end{cases}$$

Well-defined "infinite degree" polynomials are called **analytic functions**[1]. For these functions, there is a systematic way of computing the expansion coefficients to represent $\boldsymbol{f}$ on some interval (say $[-1/2, 1/2]$ again),

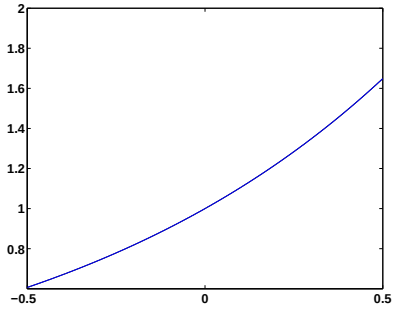
$$f(t) = \sum_{k=0}^{\infty} \alpha_k \cdot t^k, \quad \text{where} \quad \alpha_k = \frac{f^{(k)}(0)}{k!}, \tag{1}$$

where $\boldsymbol{f}^{(k)}$ is the k-th derivative of $\boldsymbol{f}$.

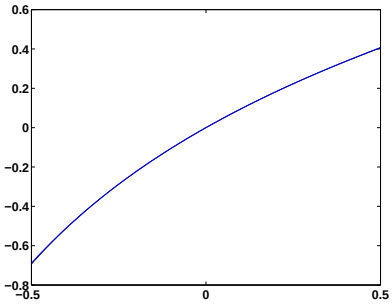
Taylor series also gives us a way to **approximate** general functions by (finite degree) polynomials: we simply truncate the sum above. Here are the three examples above with the series truncated to the first six terms:

[1] More technically, a function $f(t)$ is called analytic on an interval $[a, b]$ if for any $t_0 \in [a, b]$, the infinite sum $\sum_{k \geq 0} a_k (t - t_0)^k$ converges to $f(t)$ for some choice of $\{a_k\}$.

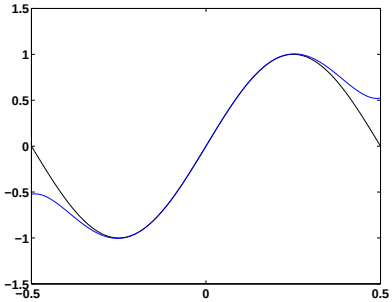
$$e^t \approx \sum_{k=1}^{6} \frac{1}{k!} t^k$$



$$\log(1+t) \approx \sum_{k=1}^{6} \frac{(-1)^{k+1}}{k} t^k$$



$$\sin(2\pi t) \approx 2\pi t - \frac{(2\pi)^3}{6} t^3 + \frac{(2\pi)^5}{120} t^5$$



The exp and log examples are pretty much spot-on with only six terms, while the sin example is still suffering a little on the edges.

It is important to realize that Taylor series is not the only way to build up a function as a sum of polynomials, and despite its convenience, it has a few unsatisfying properties. First, there are infinitely differentiable (but not analytic) functions whose Taylor series converges, but does not equal the original function anywhere. Second, we know from the Weierstrass Approximation theorem[2] that *any* continuous function can be approximated arbitrarily well on an interval by a polynomial of sufficient degree, and since very broad classes of functions can be approximated by continuous functions[3], this means

[2] https://en.wikipedia.org/wiki/Stone-Weierstrass_theorem.

[3] In particular, if $\int_a^b |f(t)|^p dt < \infty$, then for every $\epsilon > 0$ there is a continuous

that pretty much any function you can think of, continuous or not, can be approximated arbitrarily well by a polynomial. As the expansion in (1) only applies to functions with an infinite number of derivatives, Taylor series gives us no guidance on how to approximate general functions. Finally, we can see in the examples above that it is often the case that the Taylor approximation is very accurate around $t = 0$, but much less accurate as t moves away from the origin. In fact, the standard motivation for Taylor series is to get a very accurate approximation at one point. But if the goal is to approximate the function throughout the interval, then truncating the Taylor series does not provide the best approximating polynomial.

Example: Lagrange polynomials

There is also a classical method for fitting a polynomial to a set of data points. Suppose, as in the discussion above, I am given a set of $M+1$ samples[4] $y_0, \ldots, y_M$ at locations $t_0, \ldots, t_M$ and I want to find a polynomial that passes through these points.

It is a fact that there is a unique polynomial of order M that passes through $M+1$ distinct points (2 points define a line, 3 point define a parabola, etc). In fact, there is actually fairly straightforward expression for this polynomial; we take

$$f(t) = \sum_{m=0}^{M} \alpha_m \, p_m(t)$$

function $c(t)$ such that $\int_a^b |f(t) - c(t)|dt < \epsilon$.

[4] We are starting the index for the data at 0 instead of 1 in this example, simply because it makes the notation easier below.

where $\alpha_m = y_m$, and

$$p_m(t) = \prod_{\substack{0 \le k \le M \\ k \ne m}} \frac{t - t_k}{t_m - t_k}.$$

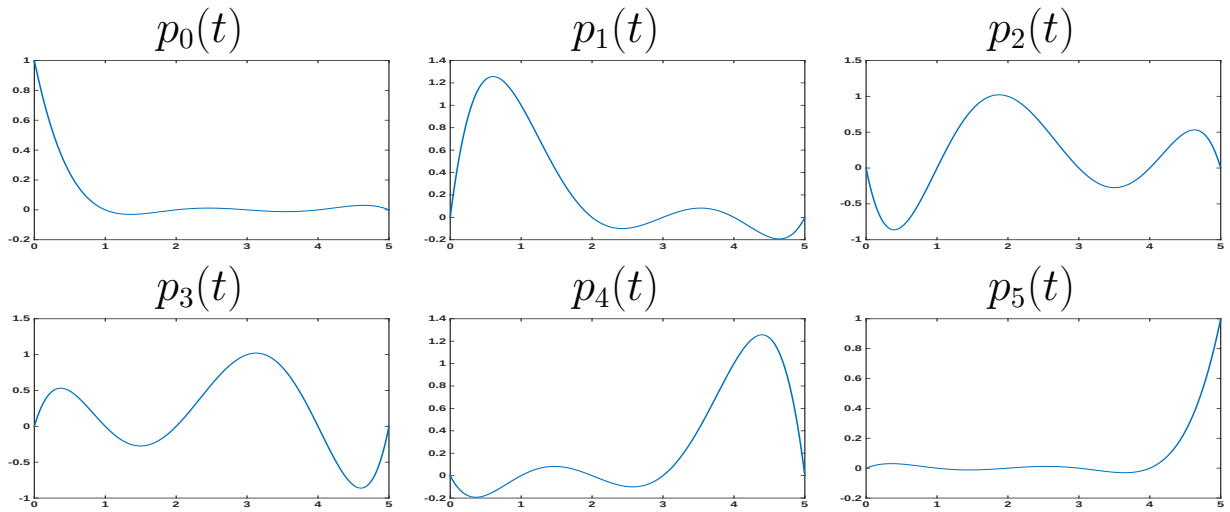
It should be clear from the expression above that each $p_m(t)$ is a different M-th order polynomial, and

$$p_m(t_k) = \begin{cases} 1, & m = k \\ 0, & m \ne k. \end{cases}$$

One consequence of these expressions is that we can reproduce an M-th order polynomial from $M+1$ samples — if $f(t)$ is a polynomial of order M and $t_0, \ldots, t_M$ obey $t_i \ne t_j$ for $i \ne j$, then

$$f(t) = \sum_{m=0}^{M} f(t_m)\, p_m(t) \quad \text{for all} \quad t \in \mathbb{R}.$$

Here are the 6 basis functions $p_k(t)$ for $M = 5$ and t_m on the integers ($t_m = m,\ m = 0, \ldots, M$):



One problem with using polynomials to interpolate (and to even represent functions in general) is that they can be unstable away from the region in which they are fit. No matter what polynomial $f(t)$ is, it goes to either $+\infty$ or $-\infty$ as $t \to \infty$ (and same for $t \to -\infty$.

Polynomial interpolation can also be unstable in the sample region itself. On the homework, we will actually explore a famous demonstration of this due to Runge[5].

Exercises. Below, take $t_0 = 0$, $t_1 = 1$, $t_2 = 2$.

1. For this set of data points, sketch $p_0(t), p_1(t), p_2(t)$ below.

2. Take $y_0 = 1$,$y_1 = 2$, $t_2 = -1$. Find quadratic polynomial $f(t)$ such that $f(t_m) = y_m$ for $m = 0, 1, 2$. Sketch the answer below along with the data points $\{(t_m, y_m)\}$.

[5] https://en.wikipedia.org/wiki/Runge's_phenomenon.

Example: Polynomial splines

A more stable way to interpolate between a sequence of discrete points is by using a **polynomial spline**. Given a sequence of locations $t_1, t_2, \ldots, t_K$ and function values at those locations $y_1, y_2, \ldots, y_K$, the ℓth order polynomial spline is the function $f(t)$ which obeys:

$$f(t_k) = y_k, \text{ for } k = 1, \ldots, K,$$

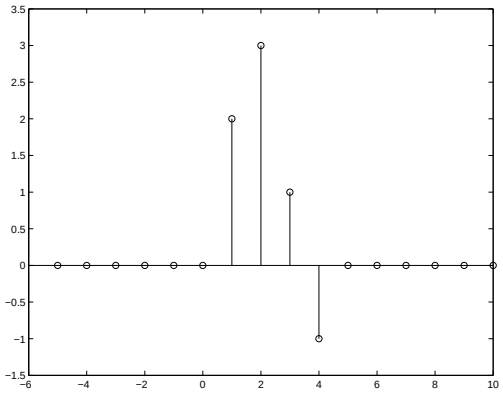
and

$f(t)$ is an ℓth order polynomial between the t_k,

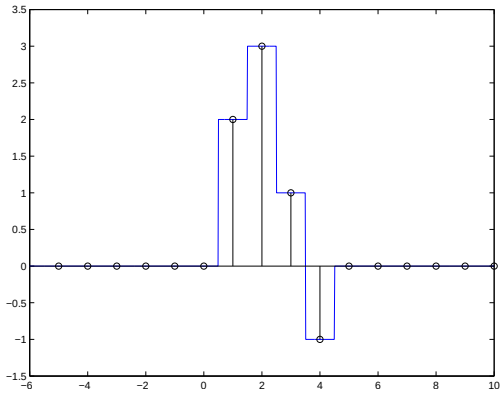
and

$f(t)$ has $\ell - 1$ continuous derivatives at the t_k.

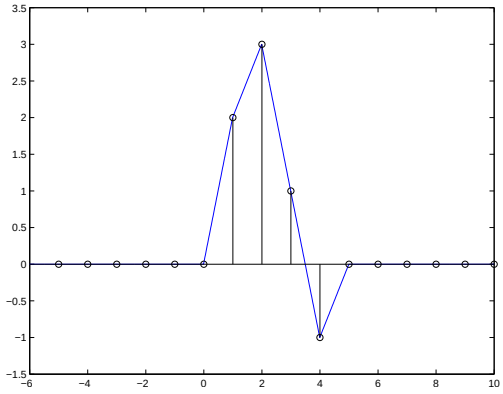
For example, if we have data points at the integers
$t_1 = 1,\ t_2 = 2,\ t_3 = 3,\ t_4 = 4$
with values
$y_1 = 2,\ y_2 = 3,\ y_3 = 1,\ y_4 = -1$
and values of zeros at the other integers,



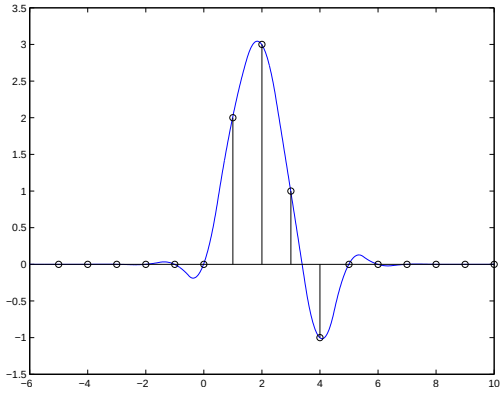
then here is the zero-th order interpolation,



the linear interpolation,



and the quadratic interpolation,



Exercises:

1. Suppose that we want to interpolate $t_1 = 1, t_2 = 2, t_3 = 3, t_4 = 4$ with values $y_1 = 2, y_2 = 3, y_3 = 1, y_4 = -1$ with a second-order spline ($\ell = 2$). How many unknowns are there? How many linear equations restrict these unknowns? What are some possible additional constraints we could add to make the solution unique?

2. Suppose that we have M points $(t_1, y_2), \ldots, (t_M, y_M)$ that we want to interpolate with a cubic polynomial spline ($\ell = 3$). How many unknowns are there? How many linear equations restrict these unknowns? What are some possible additional constraints we could add to make the solution unique?

3. Suppose that we have M points $(t_1, y_2), \ldots, (t_M, y_M)$ that we want to interpolate with a polynomial spline of order ℓ. How many unknowns are there? How many linear equations restrict these unknowns? What are some possible additional constraints we could add to make the solution unique?

Example: B-splines

Along with giving us a technique for interpolating data points, we can think of polynomial splines as a set of functions that can be generated by taking linear combinations of basis functions — these basis functions are called B-splines. In what follows below, we flesh this idea out slowly.

Piecewise constant. Let's start with the simplest case, the 0th order interpolation above. Suppose we are given samples on the integers, $\{(t_k, y_k)\}_{k\in\mathbb{Z}}$ where $t_k = k$ (almost everything below can be generalized to different sampling patterns). In this case, we create interpolate the samples to form $f(t)$ for simply by finding the "nearest neighbor" sample to t:

$$f(t) = y_{\text{round}(t)} = \begin{cases} \vdots & \\ y_{-1} & -1.5 \le t < -0.5 \\ y_0 & -0.5 \le t < 0.5 \\ y_1 & 0.5 \le t < 1.5 \\ \vdots & \end{cases}.$$

It is easy to see that this is a function that is piecewise constant between the half-integers that can be written as

$$f(t) = \sum_{k=-\infty}^{\infty} \alpha_k \, b_0(t-k), \quad b_0(t) = \begin{cases} 1 & -1/2 \le t < 1/2 \\ 0 & \text{else} \end{cases} \tag{2}$$

where we can simply take $\alpha_k = y_k$. For the example in the previous section, we have for non-zero values of y_k, and so we get a function composed of four "blocks".

Three things should be clear that this point:

1. A function $f(t)$ generated using (2) will be piecewise constant between the half-integers for any choice of $\{\alpha_k\}$.
2. Every function that is piecewise constant between the half-integers can be written in the form (2) for some $\{\alpha_k\}$, and this representation is unique (different $\{\alpha_k\}$ generate different functions).
3. Every function that is piecewise constant between the half-integers is completely characterized by its samples on the integers. Just take $\alpha_k = f(k)$ in (2) above.

We can reword the first two points using the language of linear algebra. Let $\mathcal{S}_0$ be the set of piecewise constant functions,

$$\mathcal{S}_0 = \text{functions } \boldsymbol{f} \text{ that are piecewise constant between the half-integers.}$$

Then $\{b_0(t-k)\}_{k\in\mathbb{Z}}$ form a **basis** for $\mathcal{S}_0$ — every member of $\mathcal{S}_0$ is in the linear span of the $\{b_0(t-k)\}_{k\in\mathbb{Z}}$, and there is a unique mapping from $f \in \mathcal{S}_0$ to coefficient sequences $\{\alpha_k\}_{k\in\mathbb{Z}}$.

Piecewise linear. We can follow a similar development for piecewise linear functions. If we are given samples y_k on the integers ($t_k = k$ as above), then we can linearly interpolate to $f(t)$ as

$$f(t) = \lambda y_{\lceil t \rceil} + (1-\lambda) y_{\lfloor t \rfloor}, \quad \text{where} \quad \lambda = t - \lfloor t \rfloor,$$

where $\lfloor t \rfloor$ is the largest integer smaller than t (round down) and $\lceil t \rceil$ is the smallest integer greater than t (round up).

Take

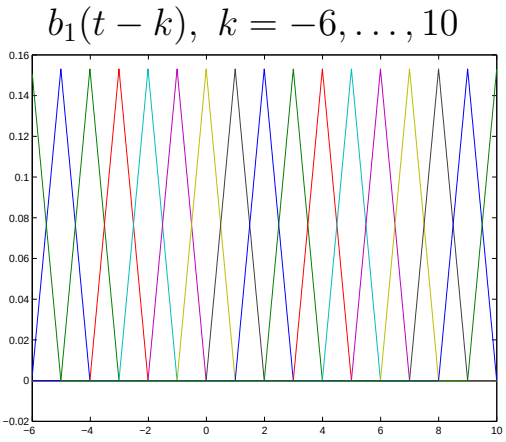
$$\mathcal{S}_1 = \text{functions that are piecewise linear between the integers.}$$

It is a fact that any function in $\mathcal{S}_1$ can be written as a superposition

of 'hat' functions:

$$f(t) = \sum_{k=-\infty}^{\infty} \alpha_k \, b_1(t-k), \quad b_1(t) = \begin{cases} t+1 & -1 \le t \le 0 \\ 1-t & 0 \le t \le 1 \\ 0 & \text{else} \end{cases} . \qquad (3)$$

Here is a sketch of a bunch of these functions for a range of k:



Working a quick example should convince you that every function of the form (3) is piecewise linear between the integers.

Exercise. Sketch $f(t) = 3b_1(t+1) + 5b_1(t) + 2b_1(t-1) - b_1(t-2)$.

It should also now be clear that any function in $\mathcal{S}_1$ can be generated by adjusting the α_k. And since for any integer ℓ, all of the $b_1(t-k)$ are zero at $t = \ell$ except $b_1(t-\ell)$, we will always have $\alpha_k = f(k)$. So an analogous set of three properties for $\mathcal{S}_1$ that we also had for $\mathcal{S}_0$:

1. A function generated using (3) will be in $\mathcal{S}_1$ for any $\{\alpha_k\}$.
2. Every function in $\mathcal{S}_1$ can be written in the form (3) for some $\{\alpha_k\}$.
3. Every function in $\mathcal{S}_1$ is completely characterized by its samples on the integers; we can take $\alpha_k = f(k)$ in (3).

So again, the $\{b_1(t-k)\}_{k\in\mathbb{Z}}$ are a basis for the space $\mathcal{S}_1$.

As a final point of interest, note that we can generate b_1 by convolving[6] b_0 with itself:

$$b_1(t) = (b_0 * b_0)(t) = \int_0^1 b_0(s)b_0(t-s)ds.$$

Piecewise quadratic. The story gets more complicated for splines of order ≥ 2, but the fundamentals lessons stay the same. First of all, we have seen in the previous section that given a set of data points, the quadratic interpolation is not unique, though it can be made unique by imposing additional constraints on the end points. We will address interpolation below, but first let's look at the B-spline space $\mathcal{S}_2$. Define

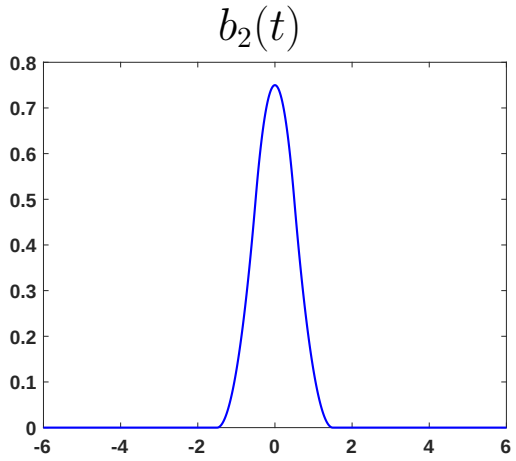
$$b_2(t) = \begin{cases} (t+3/2)^2/2 & -3/2 \leq t \leq -1/2 \\ -t^2 + 3/4 & -1/2 \leq t \leq 1/2 \\ (t-3/2)^2/2 & 1/2 \leq t \leq 3/2 \\ 0 & |t| \geq 3/2 \end{cases}.$$

Note that the B-spline function $b_2(t)$ can be generated as:

$$b_2(t) = (b_1 * b_0)(t) = (b_0 * b_0 * b_0)(t).$$

[6]Recall the general definition of convolution of two functions whose domain is the real-line: $(g*h)(t) = \int_{-\infty}^{\infty} g(s)h(t-s)ds$. The limits of the integral are determined by the support of g.

A picture of this function is shown below.



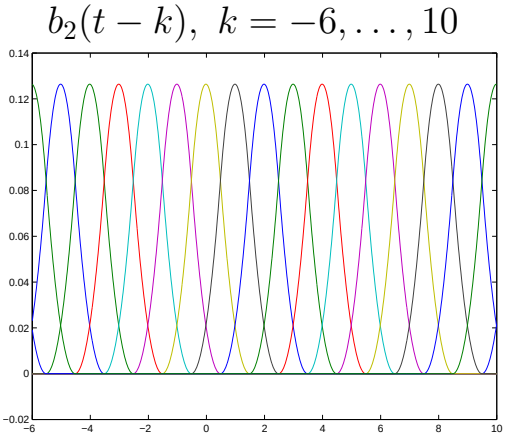
As we can readily see, it is quadratic between the half integers, and is continuous. You can also check that it is differentiable at the half integers, $t \in \{-3/2, -1/2, 1/2, 3/2\}$ are the relevant points, but not twice differentiable at these points. Let's now define

$$\mathcal{S}_2 = \left\{ \boldsymbol{f} \ : \ f(t) = \sum_{k=-\infty}^{\infty} \alpha_k b_2(t-k), \text{ for some } \{\alpha_k\} \right\}. \quad (4)$$

The question is whether or not $\mathcal{S}_2$ is the set of all quadratic splines, functions that are 2nd order polynomials between the half-integers, and are continuous and differentiable at the half-integers. As $b_2(t)$ has these properties, any linear combination of its integer shifts will as well. so It is easy to see that any function in $\mathcal{S}_2$ is a quadratic spline.

So now the question is: is every quadratic spline in $\mathcal{S}_2$? The answer is "yes", but showing this is non-trivial and takes more effort than we are willing to put in at this point. But it a fact that the $\{b_2(t-k)\}_{k\in\mathbb{Z}}$ is a basis for $\mathcal{S}_2$.

Unlike $\mathcal{S}_0$ and $\mathcal{S}_1$, there is not a simple relationship between the expansion coefficients $\{\alpha_k\}$ and the samples $f(k)$ at the integers. The main reason to see this is that the shifts of $b_2(t)$ overlap, as shown here:



For any integer k, we will have

$$\begin{aligned} f(k) &= \alpha_{k-1} b_2(1) + \alpha_k b_2(0) + \alpha_{k+1} b_2(-1) \\ &= \frac{\alpha_{k-1}}{8} + \frac{3\alpha_k}{4} + \frac{\alpha_{k+1}}{8}, \end{aligned} \tag{5}$$

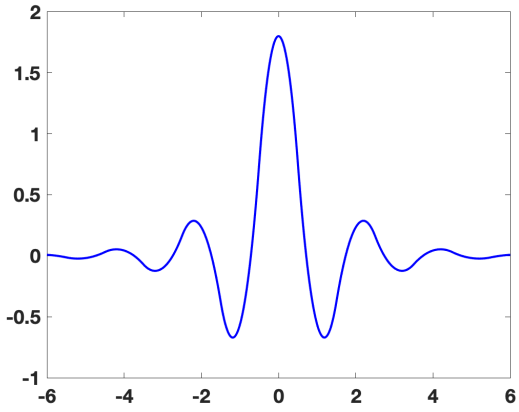
which in general will not be equal to α_k. But there is a systematic way to compute the α_k given an $\boldsymbol{f} \in \mathcal{S}_2$; we can take

Given an $\boldsymbol{f} \in \mathcal{S}_2$, there is also a concrete way to compute the expansion coefficients. There exists a complementary function $\tilde{b}_\ell(t)$, called the (second order) **dual** B-spline such that

$$\alpha_k = \int_{-\infty}^{\infty} f(t) \tilde{b}_\ell(t-k)\, dt.$$

There is no closed form for this function, but it can readily be computed. A plot is below.

$\tilde{b}_2(t)$



This function $\tilde{b}_2(t)$ is itself a quadratic spline, though unlike $b_2(t)$ it technically has infinite extent.

Is a spline in $\mathcal{B}_2$ completely characterized by its samples at the integers? It turns out, the answer to this is also "yes". We know that $\boldsymbol{f} \in \mathcal{B}_2$ is uniquely characterized by its expansion coefficients $\{\alpha_k\}$, so this is a question about whether we can recover $\{\alpha_k\}_{k\in\mathbb{Z}}$ from the samples $\{f(k)\}_{k\in\mathbb{Z}}$. From (5), we know how to map the coefficient sequence $\{\alpha_k\}$ into the sample sequence $\{f(k)\}$, so now the question becomes whether we can invert this mapping. This is a set of linear equations, but the problem is that there is an infinity of them, one for each $k \in \mathbb{Z}$. Nevertheless, this infinite system of equations is indeed invertible; however, even if there are only a finite number of samples that are non-zero, there will in general be an infinite number of α_k that are non-zero. If you don't believe that, try to solve (5) for the

simple example in the previous section, with

$$f(k) = y_k = \begin{cases} 2, & k = 1, \\ 3, & k = 2, \\ 1, & k = 3 \\ -1, & k = 4, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

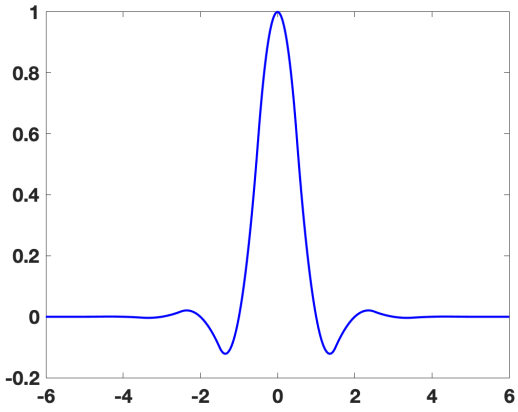
You will quickly see that an infinite number of the α_k have to be non-zero for this to work out.

There is also a generating equation similar to (4) to compute $f(t)$ directly from its samples. This done with the (second order) **cardinal spline** $\hat{b}_2(t)$; from the samples $\{f(k)\}_{k\in\mathbb{Z}}$ we can interpolate to the entire domain using

$$f(t) = \sum_{k=-\infty}^{\infty} f(k)\, \hat{b}_2(t-k).$$

Explaining how this function is derived is more than we want to do at this point, and as with the dual there is no closed form expression for it. But it can readily be computed, and is plotted below.

$\hat{b}_2(t)$



We also have $\hat{b}_2(t) \in \mathcal{S}_2$, and similar to the dual function it will have infinite extent (this is intimately related to the fact that the solution to the inverse problem put forth in (6) has infinite extent as well).

L-th order B-splines. The story for quadratic splines can be extended to L-th order splines. Let $b_L(t)$ be defined as

$$b_L(t) = b_{L-1}(t) * b_0(t) = \underbrace{b_0(t) * \cdots * b_0(t)}_{L \text{ times}}.$$

Using induction, we can see that $b_L(t)$ is an Lth order polynomial between the integers if L is odd, or between the half-integers if L is even. We can also show that $b_L(t)$ is continuous with $L-1$ continuous derivative at the integers / half-integers. Moreover, the space

$$\mathcal{S}_L = \left\{ \boldsymbol{f} \ : \ f(t) = \sum_{k=-\infty}^{\infty} \alpha_k b_L(t-k), \text{ for some } \{\alpha_k\} \right\}.$$

contains only and all L-th order splines — the $\{b_L(t-k\}_{k\in\mathbb{Z}}$ is a basis for $\mathcal{S}_L$. It is also possible to generate an $\boldsymbol{f} \in \mathcal{S}_L$ from its samples on the integers $\{f(k)\}_{k\in\mathbb{Z}}$, although this inverse problem becomes poorly conditioned as L gets larger.

Obviously, as L gets larger, $\mathcal{S}_L$ gets richer and richer — for any function, there is an L large enough so that it can be approximated with arbitrary accuracy with an element in $\mathcal{S}_L$. But there is another way we make the space richer and richer — we can shrink the grid on which the knots lie. For example, we could define

$$\mathcal{S}_0^j = \left\{ \boldsymbol{f} \ : \ f(t) = \sum_{k=-\infty}^{\infty} \alpha_k b_0(2^j t - k) \right\},$$

which is the space of functions that are piecewise constant on intervals of length 2^{-j}. As j gets larger, this representation gets richer and richer, as the intervals get smaller and smaller. Similarly, we could define

$$\mathcal{S}_1^j = \left\{ \boldsymbol{f} \ : \ f(t) = \sum_{k=-\infty}^{\infty} \alpha_k b_1(2^j t - k) \right\},$$

which will be the space of piecewise linear functions on intervals of length 2^{-j}. This space again gets richer and richer as j increases.

Exercises

1. Consider the set of functions that can be written as

$$f(t) = \alpha_0 b_2(t) + \alpha_1 b_2(t-1).$$

Suppose you know that $f(0) = 1$ and $f(1) = -1$. What must α_0 and α_1 be?

2. Consider the set of functions that can be written as

$$f(t) = \alpha_0 b_0(t) + \alpha_1 b_0(t-1) + \alpha_2 b_0(t-2) + \alpha_3 b_0(t-3).$$

How to I find the $\alpha_0, \alpha_1, \alpha_2, \alpha_3$ such that $f(t)$ is as close[7] to $\cos(t)$ as possible? This problem is a little open ended ... you might formulate the search for the $\boldsymbol{\alpha}_k$ as an optimization problem.

[7] Of course, the answer will depend on how you define "close"...

Example: Trigonometric polynomials (Fourier Series)

A (real-valued) **trigonometric polynomial** of degree N on the interval $[0, 1]$ is any function that can be written as

$$f(t) = a_0 + \sum_{k=1}^{N} a_k \cos(2\pi k t) + \sum_{k=1}^{N} b_k \sin(2\pi k t). \tag{7}$$

Note that it takes $2N + 1$ coefficients to specify a trignometric polynomial of degree N.

Re-writing the expression above makes it a little more clear where the term "trigonometric polynomial" comes from. From the Euler identity, we know that

$$e^{j2\pi kt} = \cos(2\pi k t) + j \sin(2\pi k t),$$

where $j = \sqrt{-1}$ is the imaginary unit[8]. Then we can re-write (7) as

$$f(t) = \text{Re}\left\{ \sum_{k=0}^{N} \alpha_k \, e^{j2\pi kt} \right\},$$

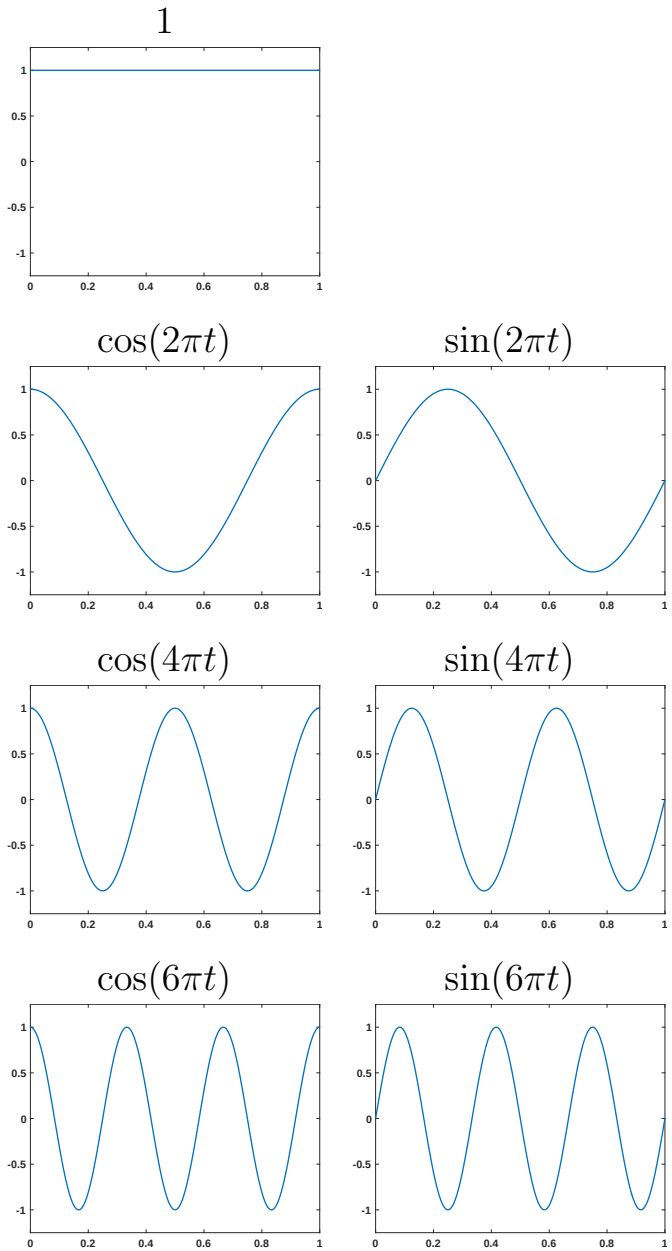
where $\alpha_0 = a_0$, $\alpha_k = a_k - j b_k$ for $k \geq 1$ are now complex expansion coefficients (we have replaced $2N + 1$ real coefficients with one real coefficient and N complex coefficients). Since

$$e^{j2\pi kt} = \left(e^{j2\pi t}\right)^k,$$

what is inside the brackets looks like a polynomial, but with t^k replaced by $(e^{j2\pi t})^k$, i.e. instead of t raised to different powers, we have a trigonometric function of t raised to different powers.

[8] Electrical engineers are the only people in the world that use j for $\sqrt{-1}$ instead of i. But that is my background, so that is the notation we are going with.

While similar in form to polynomials, the basis functions you are using to represent the function are qualitatively different. Instead of monotonic t^k growing at different rates, we are using harmonic sinusoids oscillating at different frequencies:



Just as with polynomials, almost any function can be written as an "infinite degree" trigonometric polynomial. Under only the mildest conditions (e.g. some power of f is integrable), we can write

$$f(t) = a_0 + \sum_{k=1}^{\infty} a_k \cos(2\pi k t) + \sum_{k=1}^{\infty} b_k \sin(2\pi k t), \tag{8}$$

for some $\{a_k, b_k\}$. Moreover, there is an explicit formula to compute these expansion coefficients:

$$a_0 = \int_0^1 f(t)dt, \;\; a_k = 2\int_0^1 f(t)\cos(2\pi k t)\,dt, \;\; b_k = 2\int_0^1 f(t)\sin(2\pi k t)\,dt.$$

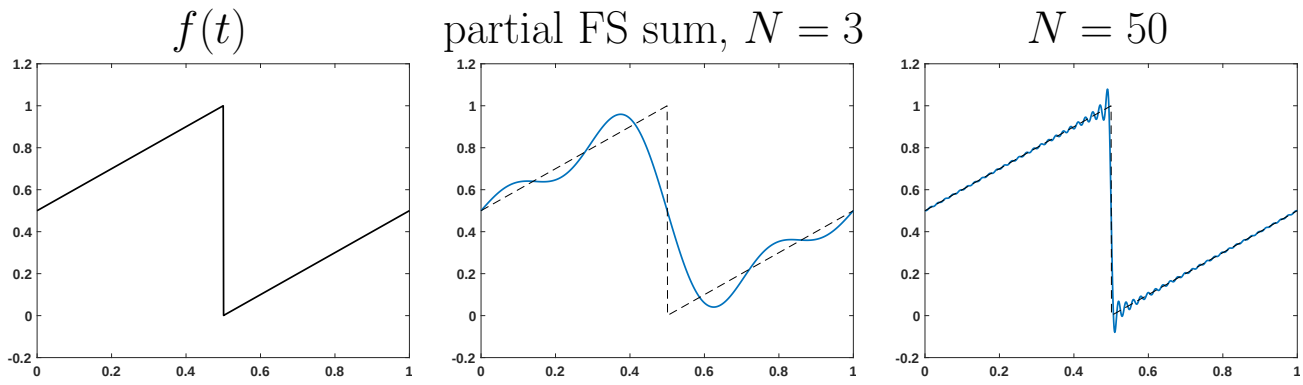
The representation (8) is called the **Fourier series**, and it plays a massive role in our understanding and modeling of the physical world. Our understanding of wave phenomena, communications signals, or differential equations would not be the same without being able to break down functions as sums of sinusoids. Fourier himself[9] discovered it in the 1820s while studying heat diffusion. It was a conceptual as well as analytical breakthough, as nobody had ever thought to try to approximate arbitrary (possible discontinuous) functions using template functions that were perfectly smooth (infinitely differentiable).

Of course, we can approximate a general function by a trigonometric polynomial simply by truncating the expression in (8). Here is the classic example of doing this for a discontinuous function; we take $f(t)$ to be the "sawtooth" function on $[0, 1]$:

$$f(t) = \begin{cases} t + 1/2, & 0 \le t < 1/2, \\ t - 1/2, & 1/2 \le t \le 1 \end{cases}.$$

[9] https://en.wikipedia.org/wiki/Joseph_Fourier.

Here is a picture of $f(t)$, along with the Fourier series truncated to 7 ($N = 3$) and 101 ($N = 50$) terms:



Note that the Fourier approximation struggles more near the discontinuity than it does where the function is smooth. This is typical behavior when approximating a discontinuous function with perfectly smooth (infinitely differentiable) functions. Nevertheless, we do pretty well for a modest number of coefficients in the approximation.

Bases and discretization

We have now seen several example of how to represent and/or approximate functions using linear combinations of fixed functions (monomials, splines, sinusoids). In doing this, we translate the function, a mapping of a continuous variable, into a discrete list of numbers.

These numbers represent weights used to build up the function out of a set of pre-determined building blocks ("basis functions"). This framework gives us a systematic way to manipulate functions of a continuous variable by operating on discrete vectors. This allows us to unleash the power of **linear algebra**.

It often times also gives us a straightforward way to simplify or compress functions. As you can see from the sawtooth Fourier series example, although it technically takes an infinity of sinusoids to build up the signals, we can get away with 50 if we are willing to suffer some loss. We will see some more examples of this later in this section.

In this set of notes, we have just gotten our first taste of basis expansions. What we will do next is develop a systematic method for taking a function and breaking it down into a superposition of basis functions. We will also discuss how to optimally approximate a function using a fixed number of basis functions — this simple idea has an incredible number of applications.

To do these things correctly, we need to first build up some mathematical machinery so we can avoid talking in hazy terms. We start in the next section with precise (but abstract) definitions of **linear vector space**, **norm**, and **inner product**.

Good sources for the material in Section I and II of these notes include:

- G. Strang, "Linear Algebra and its Applications"
- N. Young, "An Introduction to Hilbert Space"
- Naylor and Sell, "Linear Operator Theory in Engineering and Science"