EECS 16B Designing Information Devices and Systems II
Spring 2019 Discussion Worksheet

Discussion 12A

Interpolation

During the control theory module of the course, we encountered the notion of *sampling*; that is, taking a continuous time signal $y(t)$ and converting into a discrete-time signal $y_d(k)$ by taking *samples* at a given time step Δ. Expressed in symbols, we would say $y_d(k) = y(\Delta k)$. The notion of doing the opposite (that is, converting a discrete-time signal $y_d(k)$ into a continuous-time signal $y(t)$ in such a way that $y(k\Delta) = y_d(k)$) is called *interpolation*.

Interpolation is an important – and indeed necessary – part of any discrete-time system that interacts with the real world, since the real world is always in continuous-time. In other words, interpolation allows for an *interface* between discrete information and the continuous world. In its role as an interface, we have actually already met interpolation, as we used it implicitly during the control theory module. Recall that for a sampled discrete-time system, applying the discrete-time control $u(k) = y$ actually means that we apply the constant continuous-time input $u(t) = y$ over the interval $k\Delta \leq t \leq (k+1)\Delta$ to the continuous-time system. This means that $u(t)$ is an *interpolation* of $u(k)$ – specifically a *zero-order hold* interpolation, as you'll see below.

Another important use of interpolation is in *reconstruction* of a continuous-time signal that was converted to a discrete-time system previously. Consider a continuous-time audio signal, for example. In order to be stored in digital memory, or to be transmitted digitally, this signal must be sampled into discrete-time. However, the human ear hears in continuous-time. So when the time comes to listen to the audio signal, the signal must be interpolated *back* into continuous-time. In other words, the original signal must be reconstructed. An interesting and practical question is whether or not we can ever reconstruct the signal *perfectly* from the samples; we haven't shown you the tools needed to answer that question yet, but you'll want to keep it in mind.

The essential strategy for interpolating a signal is to express the interpolation as a weighted sum of some chosen continuous-time functions $\phi_i(t)$; that is, to write

$$y(t) = \sum_i \alpha_i \phi_i(t),$$

where the α_i weights on functions $\phi_i(t)$ are determined by the specific discrete-time samples $y_d(k)$ in question. From this general starting point, two specific methods emerge: the method of *interpolation by basis functions*, and the method of *interpolation by global polynomials*.

Interpolation by Basis Functions

In this method, we construct the interpolation rather directly, as

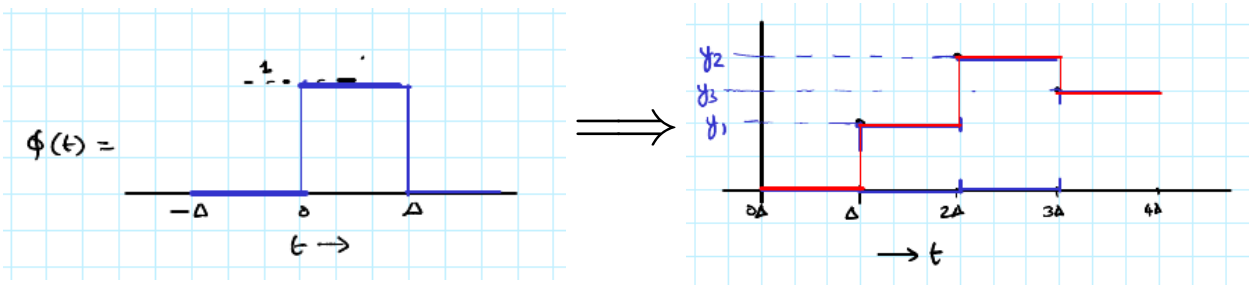
$$y(t) = \sum_{k=0}^{N-1} y_d(k)\phi(t - k\Delta),$$

where $\phi(t)$ is a *basis function* [1] of our choosing. The basis function ϕ may be any function we like that has the following two properties:

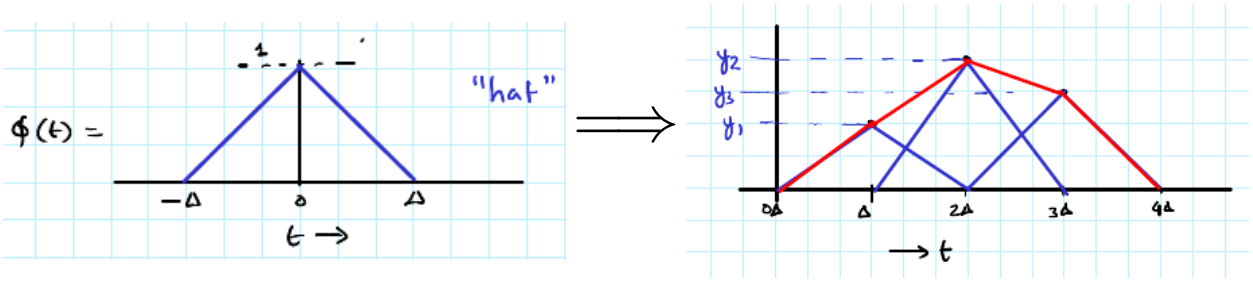
- $\phi(0) = 1$;
- $\phi(k\Delta) = 0$ for all $k \neq 0$.

That is, the function is 1 when $k = 0$ and 0 at all other k multiples of Δ. These two properties ensure that the $y(t)$ we constructed in the summation above satisfies $y(k\delta) = y_d(k)$.

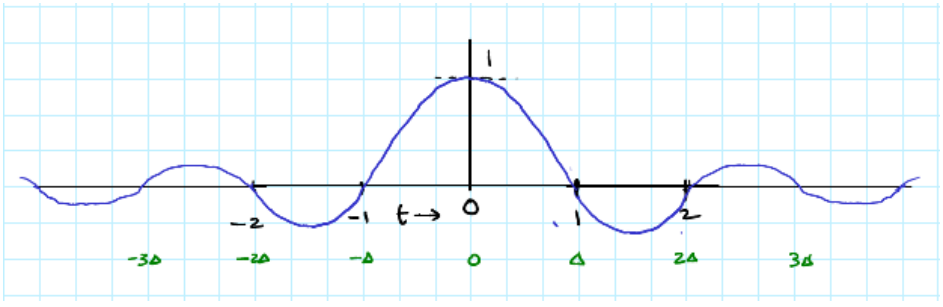
This method gives us lots of interpolation schemes. For every new basis function we come up with, we get a new interpolation scheme. Several common interpolation schemes can be written in terms of basis functions. For example, zero-order hold (ZOH) interpolation can be expressed using a box-shaped basis function in this way:



Similarly, piecewise linear (PWL) interpolation can be expressed using a triangle-shaped basis function:



Another useful basis function is the *sinc* function $\text{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$, where we take $\phi(t) = \text{sinc}(t/\Delta)$. The sinc function is differentiable and *band-limited*, the importance of which will become clear in the DFT module.



[1] Based on the similarity of the name, you probably suspect that basis functions are related to the basis of a vector space in some way. This is actually correct, in a way, but an exploration of this relationship is sadly out of scope.

Interpolation by Global Polynomials

Another method of interpolation is to find a polynomial that passes through the discrete-time data. In other words, we can construct $y(t)$ as

$$y(t) = \sum_{i=0}^{N-1} a_i t^i,$$

where the a_i are chosen in such a way that $y(k\Delta) = y_d(k)$. That is, we are summing polynomials t^i weighted by a_i to reconstruct the continuous-time signal. Notice that, since t^i do not satisfy the basis function properties, this method is distinct from the basis function method.

How can we choose a_i so that $y(t)$ is a valid interpolation? Since we require that $y(k\Delta) = y_d(k)$ for $0 \leq k \leq N-1$, it turns out that we have a system of linear equations for a_i, namely

$$\begin{aligned}
y_d(0) &= a_0 \\
y_d(1) &= a_0 + a_1\Delta + a_2\Delta^2 + \ldots + a_{N-1}\Delta^{N-1} \\
y_d(2) &= a_0 + a_1(2\Delta) + a_2(2\Delta)^2 + \ldots + a_{N-1}(k\Delta)^{N-1} \\
&\vdots \\
y_d(N-1) &= a_0 + a_1(N-1)\Delta + a_2((N-1)\Delta)^2 + \ldots + a_{N-1}((N-1)\Delta)^{N-1}.
\end{aligned}$$

Since we have N equations and N unknowns, we can try to solve for the a_i using these equations. We write the above system of equations in matrix form as

$$\begin{bmatrix} y_d(0) \\ y_d(1) \\ y_d(2) \\ \vdots \\ y_d(N-1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ 1 & \Delta & (\Delta)^2 & \ldots & \Delta^{N-1} \\ 1 & 2\Delta & (2\Delta)^2 & \ldots & (2\Delta)^{N-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & (N-1)\Delta & ((N-1)\Delta)^2 & \ldots & ((N-1)\Delta)^{N-1} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{N-1} \end{bmatrix},$$

which we will write as $\vec{y} = V\vec{a}$ for brevity. Now we can solve for $\vec{a}$ to get the interpolating polynomial, as long as the matrix V is invertible.

It turns out that the determinant of this matrix is

$$\det(V) = \prod_{0 \leq i \leq j \leq N-1} (j\Delta - i\Delta) = (\Delta)(2\Delta)\ldots((N-1)\Delta) = (N-1)!\Delta \neq 0$$

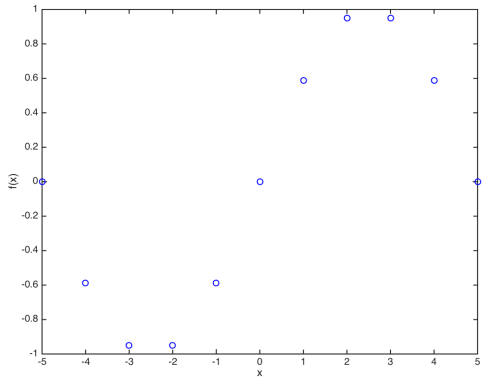
due to the *Vandermonde structure* of the matrix. Since V always has a nonzero determinant, we know that $\vec{y} = V\vec{a}$ always has a unique solution that we can find. Interestingly, it also means that the polynomial interpolation of a discrete-time signal is unique.

Questions

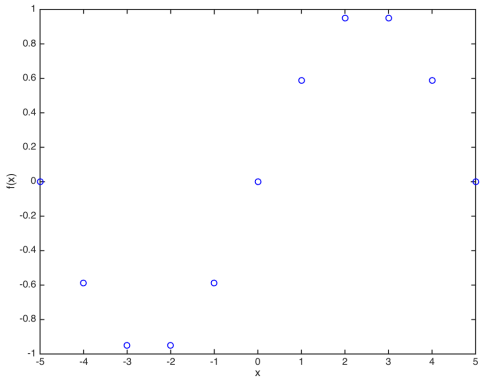
1. Warm-up: Interpolation

Samples from the sinusoid $f(x) = \sin(0.2\pi x)$ are shown below. Draw the results of interpolation using each of the following three methods:

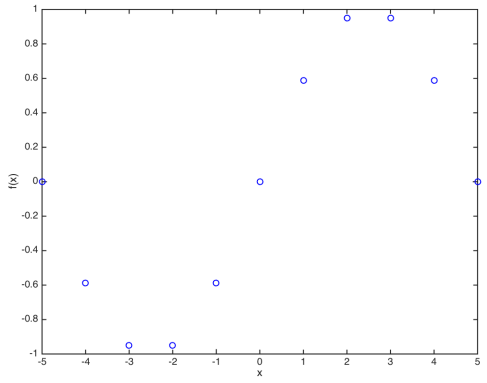
(a) Zero order hold interpolation



(b) Linear interpolation



(c) Sinc interpolation



2. How *not* to choose Δ

Consider the function $f(t) = \sin(0.2\pi t)$.

(a) At what period Δ should we sample so that sinc interpolation recovers a function that is identically zero?

(b) At what period Δ should we sample so that sinc interpolation recovers the function $\hat{f}(x) = -\sin\left(\frac{1}{15}\pi x\right)$?

3. Lagrange interpolation and polynomial basis

In practice, to approximate some unknown or complex function $f(x)$, we take n evaluations/samples of the function, denoted by $\{(x_i, y_i \triangleq f(x_i));\ 0 \leq i \leq n-1\}$. With the Occam's razor principle in mind, we try to fit a polynomial function of least degree (which is $n-1$) that passes through all the given points.

(a) Using the polynomial basis $\{1, x, x^2, \cdots, x^{n-1}\}$, the fitting problem can be cast into finding the coefficients $a_0, a_1, \cdots, a_{n-1}$ of the function

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

such that $g(x_i) = y_i$, $\forall i = 0, 1, \cdots n-1$. Find out the set of equations that need to be satisfied, and write them in a matrix form $A\vec{a} = \vec{y}$, with $\vec{a} = [a_0, a_1, \cdots, a_{n-1}]^T$ and $\vec{y} = [y_0, y_1, \cdots, y_{n-1}]^T$

(b) Now we observe that in order to find those coefficients, we need to calculate $\vec{a} = A^{-1}\vec{y}$. The matrix inversion is computationally expensive and numerically inaccurate when n is large. The idea of Lagrange interpolation is to use a different set of basis $\{L_0(x), L_1(x), \cdots, L_{n-1}(x)\}$, which has the property that

$$L_i(x_j) = \begin{cases} 1 & \text{if } j = i \\ 0 & \text{if } j \neq i \end{cases}$$

With that the fitting problem becomes finding the coefficients $b_0, b_1, \cdots, b_{n-1}$ of the function

$$h(x) = b_0 L_0(x) + b_1 L_1(x) + b_2 L_2(x) + \cdots + b_{n-1} L_{n-1}(x)$$

such that $h(x_i) = y_i$, $\forall i = 0, 1, \cdots n-1$. Again, find out the set of equations that need to be satisfied, and write them in a matrix form. What do you observe?

(c) Show that if we define

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

then the property required in part (b) is satisfied. What is the intuition behind this construction?

(d) Based on the previous two parts, write down the explicit form of $h(x)$ with the samples $\{(x_i, y_i);\ 0 \leq i \leq n-1\}$. The resulting formula is the so called Lagrange polynomial which passes through the n sampled points.

(e) Find the Lagrange polynomial given evaluated samples $f(-1) = 3, f(0) = -4, f(1) = 5, f(2) = -6$.

4. (Practice) Zero-Order Hold

Consider the following voltage samples:

Time (t)	Voltage (V)
0	0
0.25	0.48
0.5	0.84
0.75	1
1	0.9
1.25	0.6
1.5	0.14
1.75	-0.35
2	-0.75
2.25	-0.97
2.5	-0.95
2.75	-0.7

(a) Draw the zero-order hold interpolation of the samples

(b) What type of function is this?

Contributors:

- Alex Devonport.
- John Maidens.
- Yuxun Zhou.