正弦插值sinc-interpolation

# Matlab code for sinc interpolation

## sinc\_interp

**%Matlab code for sinc interpolation**

**function y = sinc\_interp(x,u)**

**% 原序列x**

**m = 0:length(x)-1;**

**U = length(u);**

**y = zeros(1,U);**

**for i=1:U**

**y(i) = sum(x.\*sinc(m- u(i)));%当m与u(i)相等时，sinc(0)=1**

%sinc函数在整数(除0以外)处,值都是0;

%由于m都是整数，所以当u(i)为整数时，sinc(m-u(i))=[00...00 100..00](选中1存在的位置)

%当u(i)不是整数的时候，此时m-u(i)就是一个序列，sinc(m-u(i))结果存在一个特征，靠近需要插入的位置

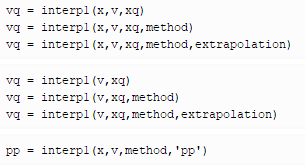
%结果很大，约0.97之上，而其他点都很小。这样保证得出的新点满足插值的条件。

**end**

**end**

## interp1函数

1-D data interpolation (table lookup)：一维数据插值。



**vq = interp1(x,v,xq)** returns interpolated values of a 1-D function at specific query points using **linear interpolation**. Vector x contains the sample points, and v contains the corresponding values, v(x). Vector xq contains the coordinates of the query points.(默认是线性插值)

If you have multiple sets of data that are sampled at the same point coordinates, then you can pass v as an array. Each column of array v contains a different set of 1-D sample values.

example

**vq = interp1(x,v,xq,method)** specifies a string for choosing an alternative interpolation method: **'nearest', 'next', 'previous', 'linear','spline','pchip', or 'cubic**'. The default method is 'linear'. 默认是线性插值

example

**vq = interp1(x,v,xq,method,extrapolation)** specifies a strategy for evaluating points that lie outside the domain of x. Set extrapolation to the string, 'extrap', when you want to use the method algorithm for extrapolation. Alternatively, you can specify a scalar value, in which case, interp1 returns that value for all points outside the domain of x.

example

**vq = interp1(v,xq)** returns interpolated values and assumes a default set of sample point coordinates. The default points are the sequence of numbers from 1 to n, where n depends on the shape of v:

When v is a vector**, the default points are 1:length(v).**

When v is an array, the default points are 1:size(v,1).

Use this syntax when you are not concerned about the absolute distances between points.

**vq = interp1(v,xq,method)** specifies any of the alternative interpolation methods and uses the default sample points.

**vq = interp1(v,xq,method,extrapolation)** specifies an extrapolation strategy and uses the default sample points.

**pp = interp1(x,v,method,'pp')** returns the piece-wise polynomial form of v(x) using the method algorithm.

### 示例：Interpolation of Coarsely Sampled **Sine Function**

Define the sample points, x, and corresponding sample values, v.

**x = 0:pi/4:2\*pi;**

**v = sin(x);**

Define the query points to be a finer sampling over the range of x.

**xq = 0:pi/16:2\*pi;**

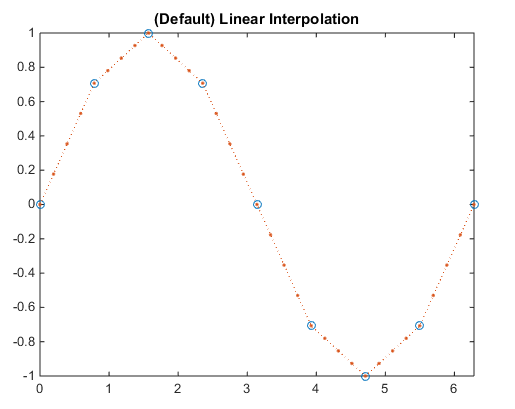
Interpolate the function at the query points and plot the result.

**figure,vq1 = interp1(x,v,xq);**

**plot(x,v,'o',xq,vq1,':.');**

**xlim([0 2\*pi]);**

**title('(Default) Linear Interpolation');**



**Now evaluate v at the same points using the 'spline' method.**

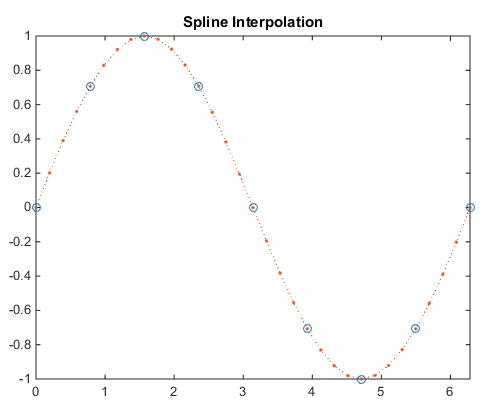
**figure**

**vq2 = interp1(x,v,xq,'spline');**

**plot(x,v,'o',xq,vq2,':.');**

**xlim([0 2\*pi]);**

**title('Spline Interpolation');**



**更多示例，参考MATLAB的doc interp1。**

# 示例Example 1

%The function is the sequence **x[n] = n a^n u[n].**

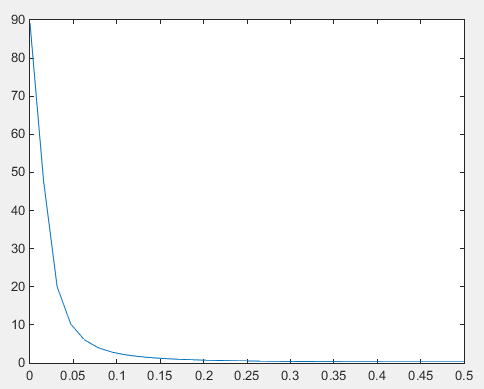
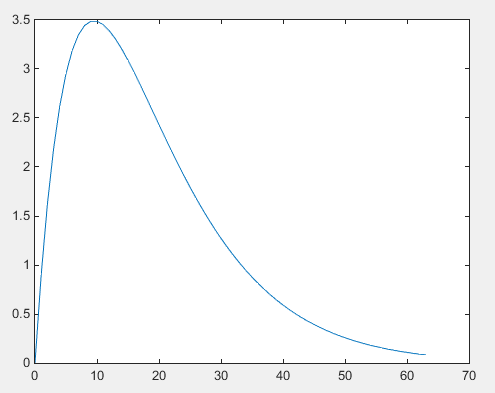
a = 0.9;

N = 64;

n = 0:N-1;

x = n.\*a.^n;

figure(1),plot(n,x)



**Magnitude of Fourier transform**

The function is essentially band-limited

y = fft(x);

k= 0:N/2;

plot(k/N,abs(y(1:N/2+1)));

**Interpolations**

**s = linspace(0,63,512);**

**x2 = sinc\_interp(x,s);**

**plot(s(1:256),x2(1:256));**

**hold**

**xi = interp1(n,x,s);**

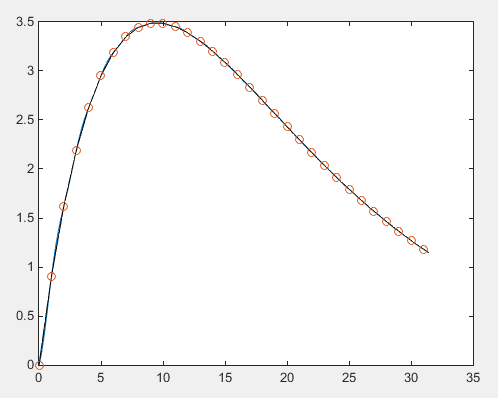
**plot(s(1:256),xi(1:256),'k');**

**plot(n(1:N/2),x(1:N/2),'o');**

**hold off**

**The first half of the data is shown.**

The black curve is the Matlab interp method. **The blue curve is sinc interpolation**. The circles are sample values.

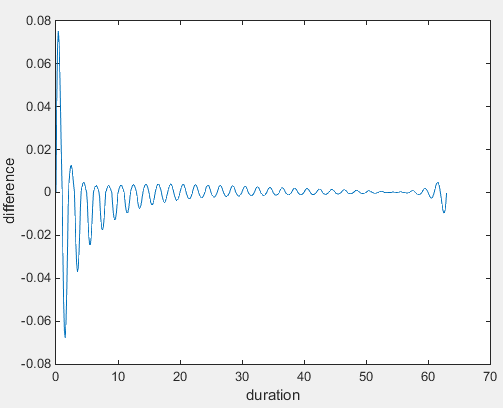


**Interpolation difference**

plot(s,xi-x2);

ylabel('difference');

xlabel('duration');



The following shows the difference between the two interpolation methods.

# 示例Example2

The function is the sequence x[n] = a^n u[n].

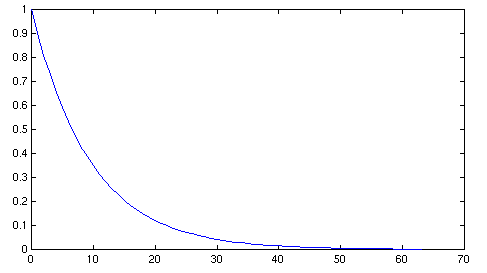
a = 0.9;

N = 64;

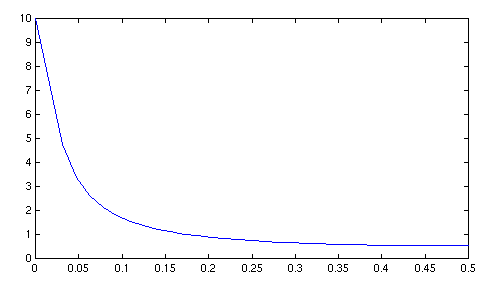
n = 0:N-1;

x = a.^n;

Sampled values



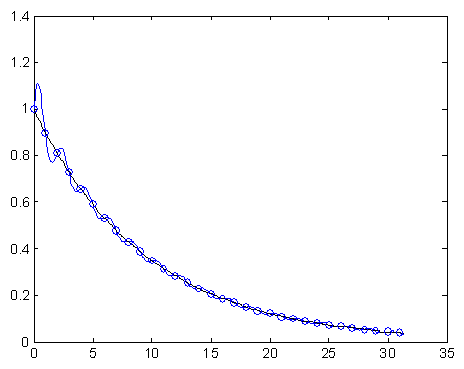
**Magnitude of Fourier transform**



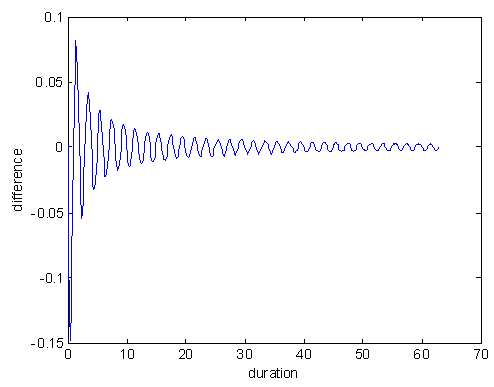
Not exactly band-limited

Interpolations

The black curve is the Matlab interp method. The blue curve is sinc interpolation. The circles are sample values.



**Interpolation difference**



# Conclusion

**sinc interpolation is seldom optimal. It almost always exhibits ringing, sometimes severely.**