Slide
078/096

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Interpolation

https://docs.scipy.org/doc/scipy/reference/tutorial/interpolate.html

https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.interp1d.html

scipy.interpolate.interp1d

```
def __init__(self, x, y, kind='linear', axis=-1,
             copy=True, bounds_error=None, fill_value=np.nan,
             assume_sorted=False):
```

class scipy.interpolate.interp1d(*x, y, kind='linear', axis=-1, copy=True, bounds_error=None, fill_value=nan, assume_sorted=False*) [source]

Interpolate a 1-D function.

How to return a function?

031

x and *y* are arrays of values used to approximate some function f: `y = f(x)`. This class returns a function whose call method uses interpolation to find the value of new points.

Note that calling interp1d with NaNs present in input values results in undefined behaviour.

kind : *str or int, optional*

Specifies the kind of interpolation as a string ('linear', 'nearest', 'zero', 'slinear', 'quadratic', 'cubic', 'previous', 'next', where 'zero', 'slinear', 'quadratic' and 'cubic' refer to a spline interpolation of zeroth, first, second or third order; 'previous' and 'next' simply return the previous or next value of the point) or as an integer specifying the order of the spline interpolator to use. Default is 'linear'.

Slide
079/096

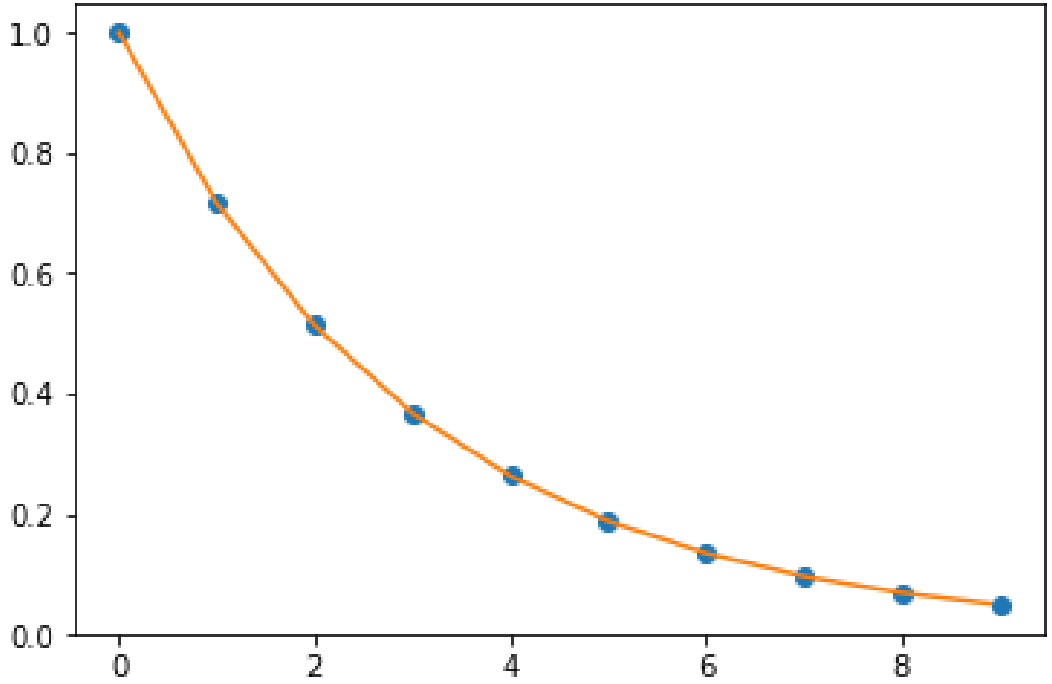
```python
import numpy as np
import matplotlib.pyplot as plt
from scipy import interpolate
x = np.arange(0, 10)
y = np.exp(-x/3.0)
f = interpolate.interp1d(x, y)
```

031

f is an instance.

```python
xnew = np.arange(0, 9, 0.1)
ynew = f(xnew)   # use interpolation function returned by `interp1d`
plt.plot(x, y, 'o', xnew, ynew, '-')
plt.show()
```

f is a callable instance.



How to return a piecewise- defined function?

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Slide
080/096

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

```
f.__dict__
```

```
{'_call': <function scipy.interpolate.interpolate.interp1d._call_linear>,
 '_extrapolate': False,
 '_fill_value_above': array([nan]),
 '_fill_value_below': array([nan]),
 '_fill_value_orig': array(nan),
 '_kind': 'linear',
 '_y': array([[1.        ],
        [0.71653131],
        [0.51341712],
        [0.36787944],
        [0.26359714],
        [0.1888756 ],
        [0.13533528],
        [0.09697197],
        [0.06948345],
        [0.04978707]]),
 'axis': 0,
 'bounds_error': True,
 'copy': True,
 'x': array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 'y': array([1.        , 0.71653131, 0.51341712, 0.36787944, 0.26359714,
        0.1888756 , 0.13533528, 0.09697197, 0.06948345, 0.04978707])}
```

f is a callable object, with instance attributes x and y.

x: sorted

Slide
081/096

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

class myinterp1d

For each item in xnew, denoted as x0, compute y0

1. find interval i: x[i-1]<x0<=x[i]
2. m=(y[i]-y[i-1])/(x[i]-x[i-1])
3. y0=y[i]+m*(x0-x[i-1])

```
import numpy as np
class myinterp1d(object):
    def __init__(self, x, y):
        self.x=x
        self.y=y

    def __call__(self, xnew):
        ynew=[]
        for x0 in xnew:
            if x0<=self.x[0]:
                ynew.append(self.y[0])
            elif x0>=self.x[-1]:
                ynew.append(self.y[-1])
            else:
                hi=next(filter(lambda x: x0<x[1], enumerate(self.x)))[0]
                lo=hi-1
                m=(self.y[hi]-self.y[lo])/(self.x[hi]-self.x[lo])
                ynew.append(self.y[lo]+m*(x0-self.x[lo]))
        ynew=np.array(ynew)
        return ynew
x = np.arange(0, 10)
y = np.exp(-x/3.0)
f = myinterp1d(x, y)
xnew = np.arange(0, 9, 0.1)
ynew = f(xnew)   # use interpolation function returned by `interp1d`
plt.plot(x, y, 'o', xnew, ynew, '-')
plt.show()
```

032

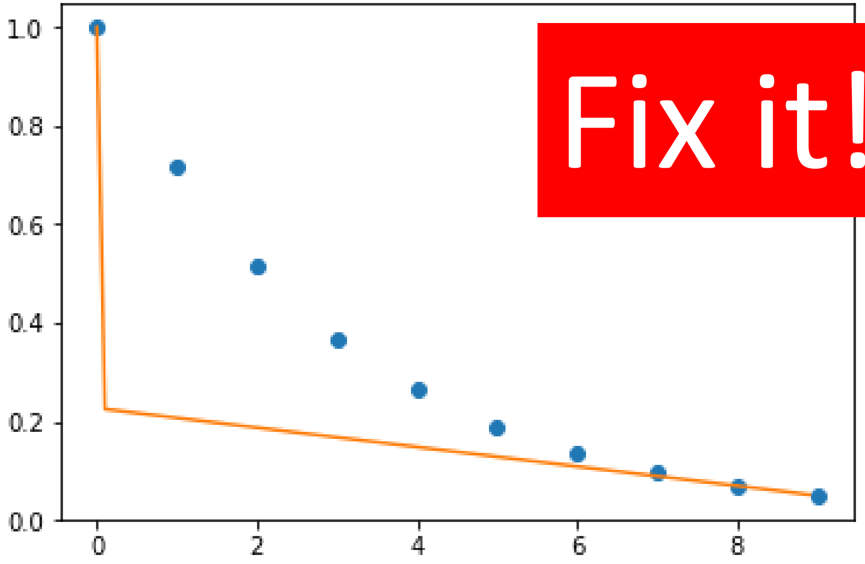
Slide
082/096

How to return a list of functions as piecewise-defined function ???

```python
import numpy as np
import matplotlib.pyplot as plt
class myinterp1d(object):
    def __init__(self, x, y):
        self.x=x
        self.y=y
        self.nInt=len(x)-1  #number of intervals
        self.f=self._linear()

    def _linear(self):
        f=[]
        for i in range(self.nInt):
            m=(y[i+1]-y[i])/(x[i+1]-x[i])
            b=y[i]-m*x[i]
            f.append(lambda x: m*x+b)
        return f

    def _linear_interp(self, x0):
        if x0<=self.x[0]:
            return y[0]
        elif x0>=self.x[-1]:
            return y[-1]
        else:
            i=next(filter(lambda s: x0<s[1], enumerate(self.x)))[0]
            return self.f[i-1](x0)

    def __call__(self, xnew):
        return np.array([self._linear_interp(x0) for x0 in xnew])

x = np.arange(0, 10)
y = np.exp(-x/3.0)
f = myinterp1d(x, y)
xnew = np.arange(0, 9, 0.1)
ynew = f(xnew)   # use interpolation function returned by `interp1d`
plt.plot(x, y, 'o', xnew, ynew, '-')
plt.show()
```

```python
def _linear(self):
    f=[]
    for i in range(self.nInt):
        m=(y[i+1]-y[i])/(x[i+1]-x[i])
        b=y[i]-m*x[i]
Hint: ⇨   f.append(lambda x: m*x+b)
    return f
```

033



Fix it!

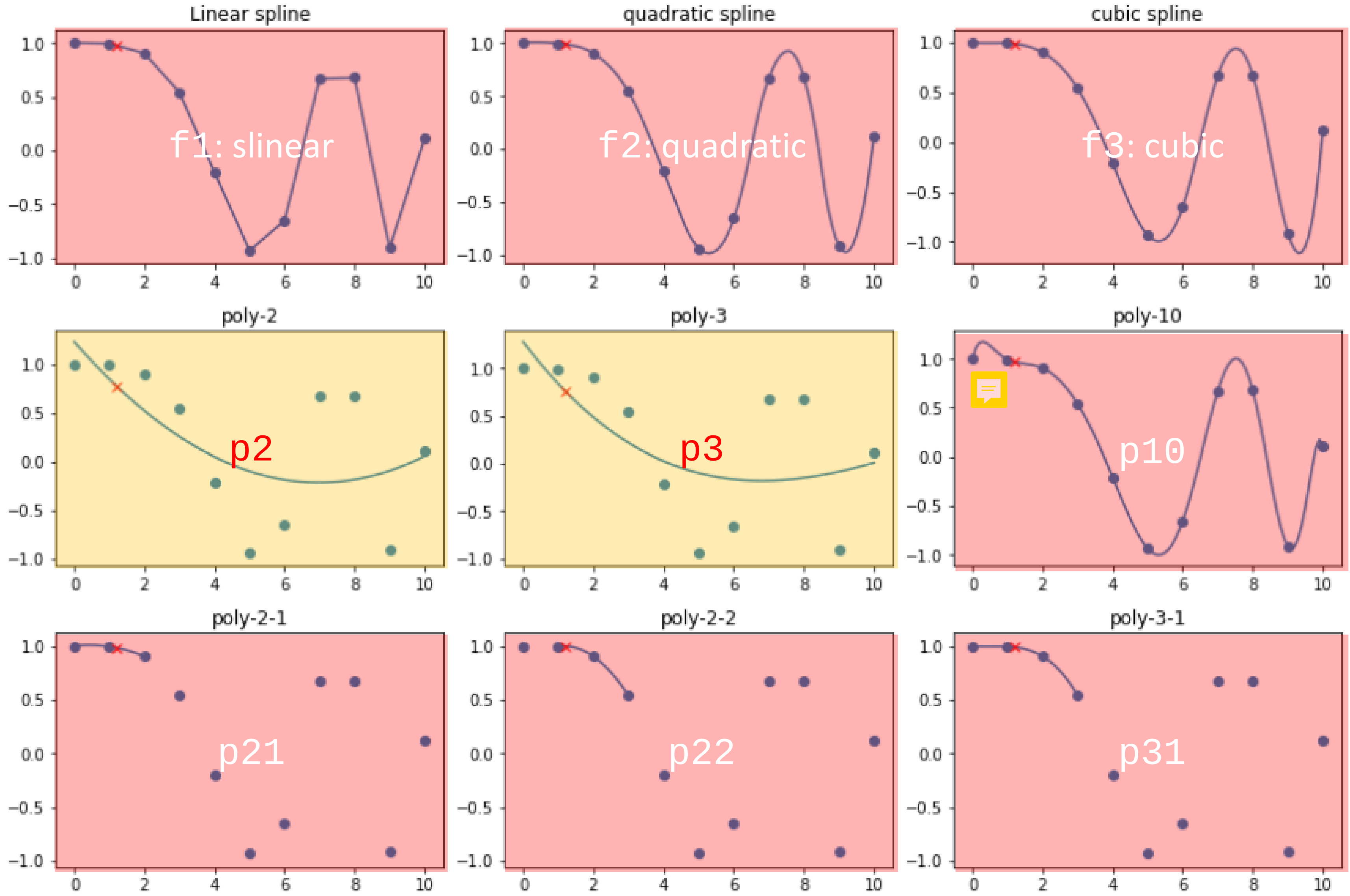
SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

Slide
083/096

interpolation vs curve fitting



QF666
Programming and
Computational
Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide 084/096

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

https://docs.scipy.org/doc/numpy/reference/generated/numpy.polyfit.html

https://docs.scipy.org/doc/numpy/reference/generated/numpy.polyval.html

numpy.polyfit

numpy.polyfit(*x, y, deg, rcond=None, full=False, w=None, cov=False*) [source]

Least squares polynomial fit.

Fit a polynomial `p(x) = p[0] * x**deg + ... + p[deg]` of degree *deg* to points *(x, y)*. Returns a vector of coefficients *p* that minimises the squared error.

034

x, y, deg ⇨ p

p, x ⇨ y

This x is not that x.

numpy.polyval

numpy.polyval(*p, x*) [source]

Evaluate a polynomial at specific values.

If *p* is of length N, this function returns the value:

```
p[0]*x**(N-1) + p[1]*x**(N-2) + ... + p[N-2]*x + p[N-1]
```

If *x* is a sequence, then *p(x)* is returned for each element of *x*. If *x* is another polynomial then the composite polynomial *p(x(t))* is returned.

Slide 085/096

```python
from scipy.interpolate import interp1d
#Linear interpolation by scipy.interpolate.interp1d
f1=interp1d(x,y,kind='slinear')

#Quadratic spline interpolation by scipy.interpolate.interp1d
f2=interp1d(x,y,kind='quadratic')

#Cubic spline interpolation by scipy.interpolate.interp1d
f3=interp1d(x,y,kind='cubic')

#Curve fitting (quadratic polynomial) by numpy.polyfit
p2=np.polyfit(x,y,2)

#Curve fitting (cubic polynomial) by numpy.polyfit
p3=np.polyfit(x,y,3)

#Curve fitting (polynomial of degree 10) by numpy.polyfit
p10=np.polyfit(x,y,10)

#Local quadratic interpolation
p21=np.polyfit(x[:3], y[:3],2)
p22=np.polyfit(x[1:4], y[1:4],2)

#Local cubic interpolation
p31=np.polyfit(x[:4], y[:4],3)
```

- ✓ interp1d
- ✓ polyfit

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

```python
#Interpolation approximation of fucntion value at x0
x0=1.2
print('f1(x0):', f1(x0))
print('f2(x0):', f2(x0))
print('f3(x0):', f3(x0))
print('p2(x0):', np.polyval(p2,x0))
print('p3(x0):', np.polyval(p3,x0))
print('p10(x0):', np.polyval(p10,x0))
print('p21(x0):', np.polyval(p21,x0))
print('p22(x0):', np.polyval(p22,x0))
print('p31(x0):', np.polyval(p31,x0))
```

```
f1(x0): 0.6759084722655498
f2(x0): 0.6698977406045836
f3(x0): 0.670255452402847
p2(x0): 0.6983127386423347
p3(x0): 0.6786229551598073
p10(x0): 0.6703199161128766
p21(x0): 0.6694801124347486
p22(x0): 0.671302351171146
p31(x0): 0.6702090079293077
```

✓ polyval

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

```
import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"]=[12,8]

fig, ax = plt.subplots(3, 3)

xp=np.linspace(0,10,101)
yp=f1(xp)
ax[0,0].scatter(x, y)
ax[0,0].plot(xp, yp)
ax[0,0].plot(x0, f1(x0), 'rx')
ax[0,0].set_title('Linear spline')

xp=np.linspace(0,10,101)
yp=f2(xp)
ax[0,1].scatter(x, y)
ax[0,1].plot(xp, yp)
ax[0,1].plot(x0, f2(x0), 'rx')
ax[0,1].set_title('quadratic spline')

xp=np.linspace(0,10,101)
yp=f3(xp)
ax[0,2].scatter(x, y)
ax[0,2].plot(xp, yp)
ax[0,2].plot(x0, f3(x0), 'rx')
ax[0,2].set_title('cubic spline')

xp=np.linspace(0,10,101)
yp=np.polyval(p2,xp)
ax[1,0].scatter(x, y)
ax[1,0].plot(xp, yp)
ax[1,0].plot(x0, np.polyval(p2,x0), 'rx')
ax[1,0].set_title('poly-2')

xp=np.linspace(0,10,101)
yp=np.polyval(p3,xp)
ax[1,1].scatter(x, y)
ax[1,1].plot(xp, yp)
ax[1,1].plot(x0, np.polyval(p3,x0), 'rx')
ax[1,1].set_title('poly-3')
```

```
xp=np.linspace(0,10,101)
yp=np.polyval(p10,xp)
ax[1,2].scatter(x, y)
ax[1,2].plot(xp, yp)
ax[1,2].plot(x0, np.polyval(p10,x0), 'rx')
ax[1,2].set_title('poly-10')

xp=np.linspace(0,2,21)
yp=np.polyval(p21,xp)
ax[2,0].scatter(x, y)
ax[2,0].plot(xp, yp)
ax[2,0].plot(x0, np.polyval(p21,x0), 'rx')
ax[2,0].set_title('poly-2-1')

xp=np.linspace(1,3,21)
yp=np.polyval(p22,xp)
ax[2,1].scatter(x, y)
ax[2,1].plot(xp, yp)
ax[2,1].plot(x0, np.polyval(p22,x0), 'rx')
ax[2,1].set_title('poly-2-2')

xp=np.linspace(0,3,31)
yp=np.polyval(p31,xp)
ax[2,2].scatter(x, y)
ax[2,2].plot(xp, yp)
ax[2,2].plot(x0, np.polyval(p31,x0), 'rx')
ax[2,2].set_title('poly-3-1')

#plt.tight_layout(pad=0.4, w_pad=0.5, h_pad=1.0)
plt.tight_layout()
plt.show()
```

```python
import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"]=[12,8]

xp=np.linspace(0,10,101)
xp21=np.linspace(0,2,21)
xp22=np.linspace(1,3,21)
xp31=np.linspace(0,3,31)
xps=[[xp, xp, xp],
    [xp, xp, xp],
    [xp21, xp22, xp31]]

yps=[[f1(xp), f2(xp), f3(xp)],
    [np.polyval(p2,xp), np.polyval(p3,xp), np.polyval(p10,xp)],
    [np.polyval(p21,xp21), np.polyval(p22,xp22), np.polyval(p31,xp31)]]

yp0=[[f1(x0), f2(x0), f3(x0)],
     [np.polyval(p2,x0), np.polyval(p3,x0), np.polyval(p10,x0)],
     [np.polyval(p21,x0), np.polyval(p22,x0), np.polyval(p31,x0)]]

titles=[['Linear spline', 'quadratic spline', 'cubic spline'],
        ['poly-2', 'poly-3', 'poly-10'],
        ['poly-2-1', 'poly-2-2', 'poly-3-1']]

fig, ax = plt.subplots(3, 3)

for i in range(3):
    for j in range(3):
        ax[i,j].scatter(x, y)
        ax[i,j].plot(xps[i][j], yps[i][j])
        ax[i,j].plot(x0, yp0[i][j], 'rx')
        ax[i,j].set_title(titles[i][j])

#plt.tight_layout(pad=0.4, w_pad=0.5, h_pad=1.0)
plt.tight_layout()
plt.show()
```

Use loop

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

Interpolation (1)

https://www.mathworks.com/help/matlab/interpolation.html

035

`interp1`	1-D data interpolation (table lookup)
`interp2`	Interpolation for 2-D gridded data in meshgrid format
`interp3`	Interpolation for 3-D gridded data in meshgrid format
`interpn`	Interpolation for 1-D, 2-D, 3-D, and N-D gridded data in ndgrid format
`griddedInterpolant`	Gridded data interpolation
`pchip`	Piecewise Cubic Hermite Interpolating Polynomial (PCHIP)
`spline`	Cubic spline data interpolation
`ppval`	Evaluate piecewise polynomial
`mkpp`	Make piecewise polynomial
`unmkpp`	Extract piecewise polynomial details
`padecoef`	Padé approximation of time delays
`interpft`	1-D interpolation (FFT method)

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

SMU Singapore Management University

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

interp1

1-D data interpolation (table lookup)

Syntax

```
vq = interp1(x,v,xq)
vq = interp1(x,v,xq,method)
vq = interp1(x,v,xq,method,extrapolation)
```

```
vq = interp1(v,xq)
vq = interp1(v,xq,method)
vq = interp1(v,xq,method,extrapolation)
```

```
pp = interp1(x,v,method,'pp')
```

Cubic spline **end conditions**:

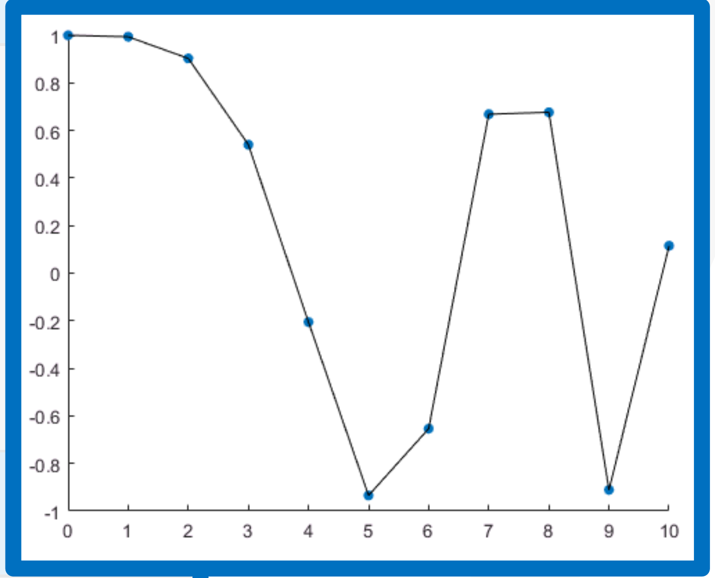
- ✓ Not-a-knot
- Clamped
- Natural
- Periodic

035

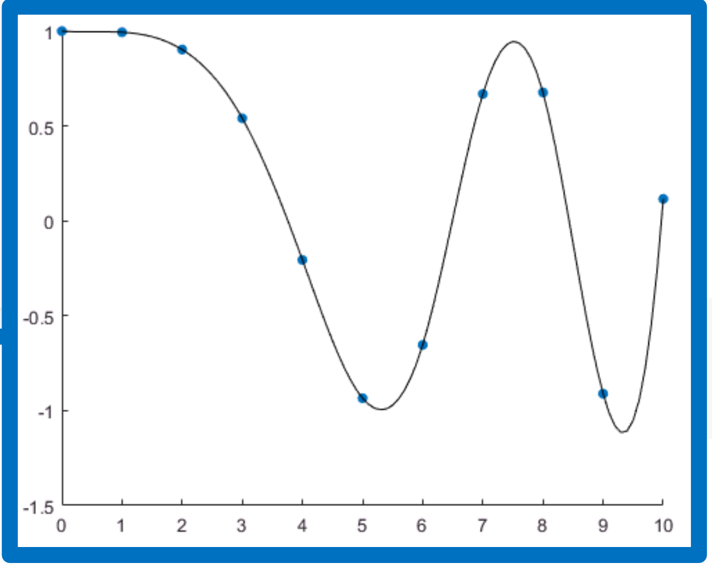
Slide
091/096

Interpolation

```
% (Default) linear interpolation
figure();
scatter(x,y,30,'filled');
hold on;
xq=linspace(0,10,101);
yq=interp1(x,y,xq);
plot(xq,yq,'k-')
```




```
% 'spline': cubic spline with not-a-knot end condition
% x0, x1, ..., xN: third derivative is continuous at x1 and xN-1
figure();
scatter(x,y,30,'filled');
hold on;
xq=linspace(0,10,101);
yq=interp1(x,y,xq,'spline');
plot(xq,yq,'k-')
```



035

SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

Polynomials

https://www.mathworks.com/help/matlab/polynomials.html

polyfit and polyval

polyfit

Polynomial curve fitting

Syntax

```
p = polyfit(x,y,n)
[p,S] = polyfit(x,y,n)
[p,S,mu] = polyfit(x,y,n)
```

Description

`p = polyfit(x,y,n)` returns the coefficients for a polynomial `p(x)` of degree `n` that is a best fit (in a least-squares sense) for the data in `y`. The coefficients in `p` are in descending powers, and the length of `p` is `n+1`

$$p(x) = p_1x^n + p_2x^{n-1} + ... + p_nx + p_{n+1}.$$

polyval

Polynomial evaluation

Syntax

```
y = polyval(p,x)
[y,delta] = polyval(p,x,S)
y = polyval(p,x,[],mu)
[y,delta] = polyval(p,x,S,mu)
```

Description

`y = polyval(p,x)` evaluates the polynomial `p` at each point in `x`. The argument `p` is a vector of length `n+1` whose elements are the coefficients (in descending powers) of an nth-degree polynomial:

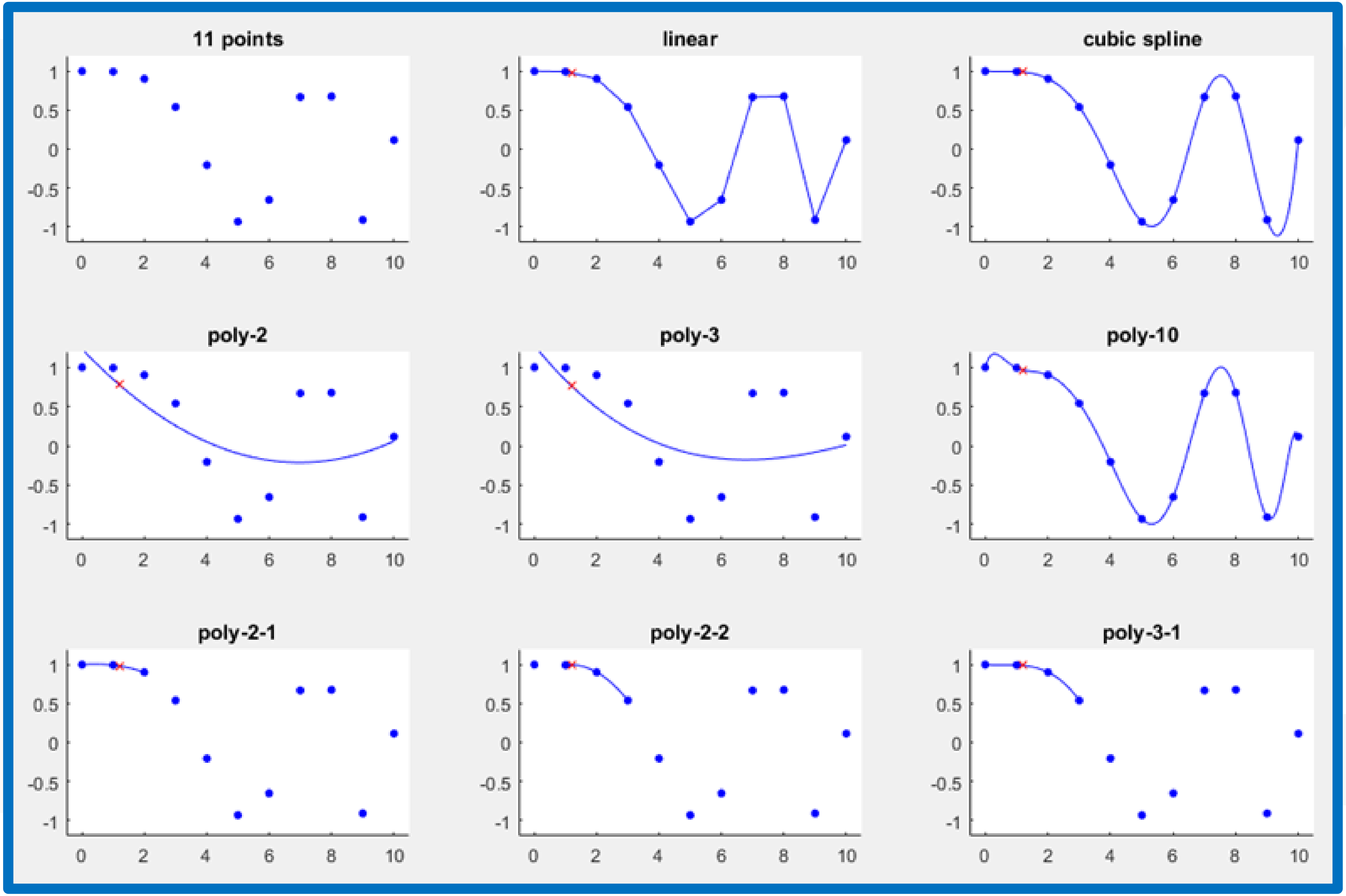
$$p(x) = p_1x^n + p_2x^{n-1} + ... + p_nx + p_{n+1}.$$

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance

Slide 093/096



SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer Of Quantitative Finance

Slide 094/096

1

```
%subplots
subplot(3,3,1);
x=linspace(0,10,11);
y=cos(-x.^2/9);
scatter(x,y,20,'blue','filled')
axis(ax)
title('11 points')
```

2

```
%subplots
subplot(3,3,2);
scatter(x,y,20,'blue','filled')
hold on;
xq=linspace(0,10,101);
yq=interp1(x,y,xq);
y0=interp1(x,y,x0);
plot(xq,yq,'b-')
plot(x0,y0,'rx')
axis(ax)
title('linear')
```

3

```
% subplots
subplot(3,3,3);
scatter(x,y,20,'blue','filled')
hold on;
xq=linspace(0,10,101);
yq=interp1(x,y,xq,'spline');
y0=interp1(x,y,x0,'spline');
plot(xq,yq,'b-')
plot(x0,y0,'rx')
axis(ax)
title('cubic spline')
```

4

```
% subplots
subplot(3,3,4);
scatter(x,y,20,'blue','filled')
hold on;
xq=linspace(0,10,101);
yq=polyval(polyfit(x,y,2),xq);
y0=polyval(polyfit(x,y,2),x0);
plot(xq,yq,'b-')
plot(x0,y0,'rx')
axis(ax)
title('poly-2')
```

5

```
% subplots
subplot(3,3,5);
scatter(x,y,20,'blue','filled')
hold on;
xq=linspace(0,10,101);
yq=polyval(polyfit(x,y,3),xq);
y0=polyval(polyfit(x,y,3),x0);
plot(xq,yq,'b-')
plot(x0,y0,'rx')
axis(ax)
title('poly-3')
```

6

```
% subplots
subplot(3,3,6);
scatter(x,y,20,'blue','filled')
hold on;
xq=linspace(0,10,101);
yq=polyval(polyfit(x,y,10),xq);
y0=polyval(polyfit(x,y,10),x0);
plot(xq,yq,'b-')
plot(x0,y0,'rx')
axis(ax)
title('poly-10')
```

7

```
% subplots
subplot(3,3,7);
scatter(x,y,20,'blue','filled')
hold on;
xq=linspace(0,2,21);
yq=polyval(polyfit(x(1:3),y(1:3),2),xq);
y0=polyval(polyfit(x(1:3),y(1:3),2),x0);
plot(xq,yq,'b-')
plot(x0,y0,'rx')
axis(ax)
title('poly-2-1')
```

8

```
% subplots
subplot(3,3,8);
scatter(x,y,20,'blue','filled')
hold on;
xq=linspace(1,3,21);
yq=polyval(polyfit(x(2:4),y(2:4),2),xq);
y0=polyval(polyfit(x(2:4),y(2:4),2),x0);
plot(xq,yq,'b-')
plot(x0,y0,'rx')
axis(ax)
title('poly-2-2')
```

9

```
% subplots
subplot(3,3,9);
scatter(x,y,20,'blue','filled')
hold on;
xq=linspace(0,3,31);
yq=polyval(polyfit(x(1:4),y(1:4),3),xq);
y0=polyval(polyfit(x(1:4),y(1:4),3),x0);
plot(xq,yq,'b-')
plot(x0,y0,'rx')
axis(ax)
title('poly-3-1')
```


SMU SINGAPORE MANAGEMENT UNIVERSITY

QF666 Programming and Computational Finance

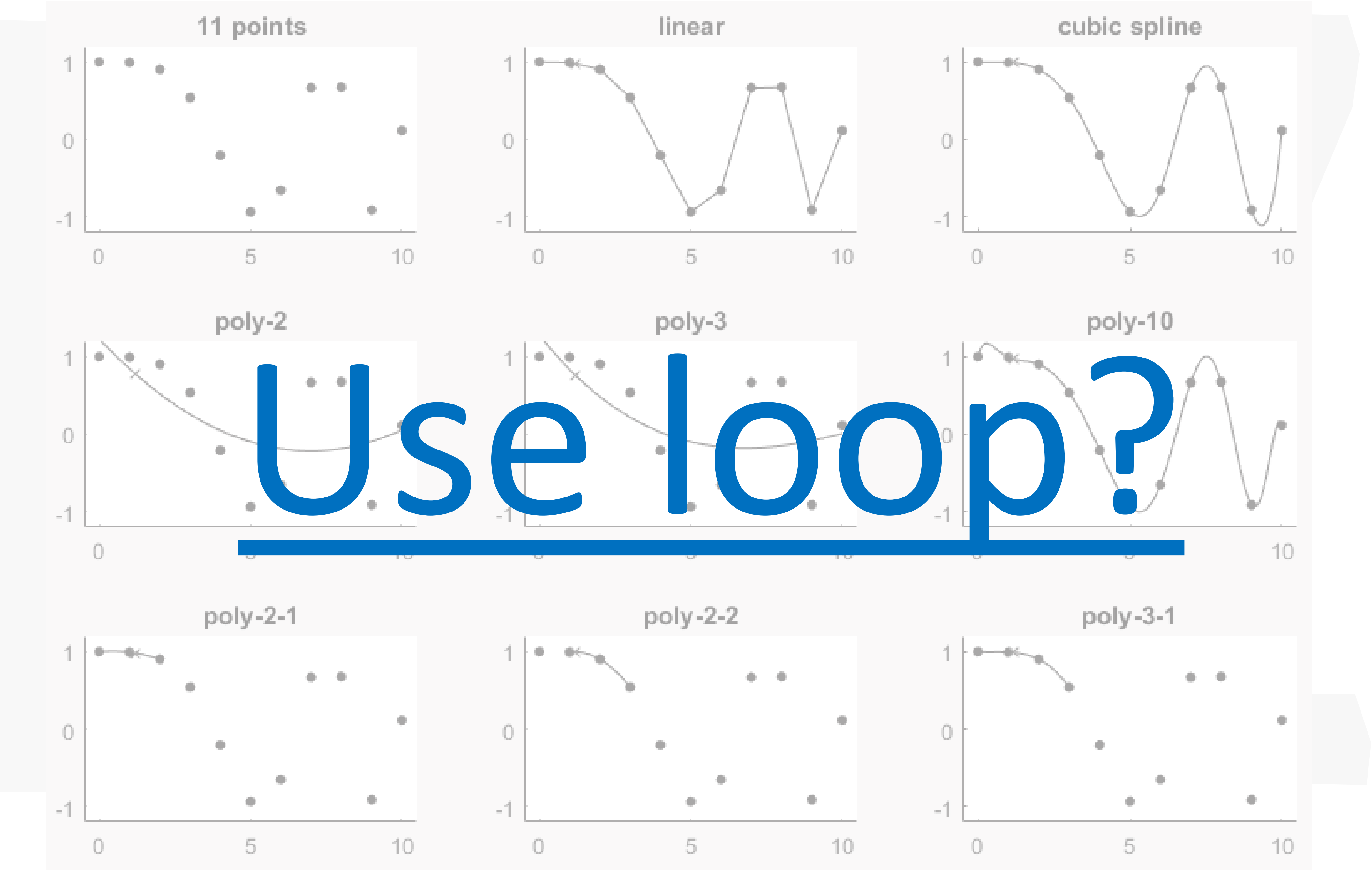
Dr. Zhao Yibao
Senior Lecturer
Of Quantitative
Finance

Slide
095/096

SMU
SINGAPORE MANAGEMENT UNIVERSITY

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance



Use loop?

end

QF666
Programming and Computational Finance

Dr. Zhao Yibao
Senior Lecturer
Of Quantitative Finance