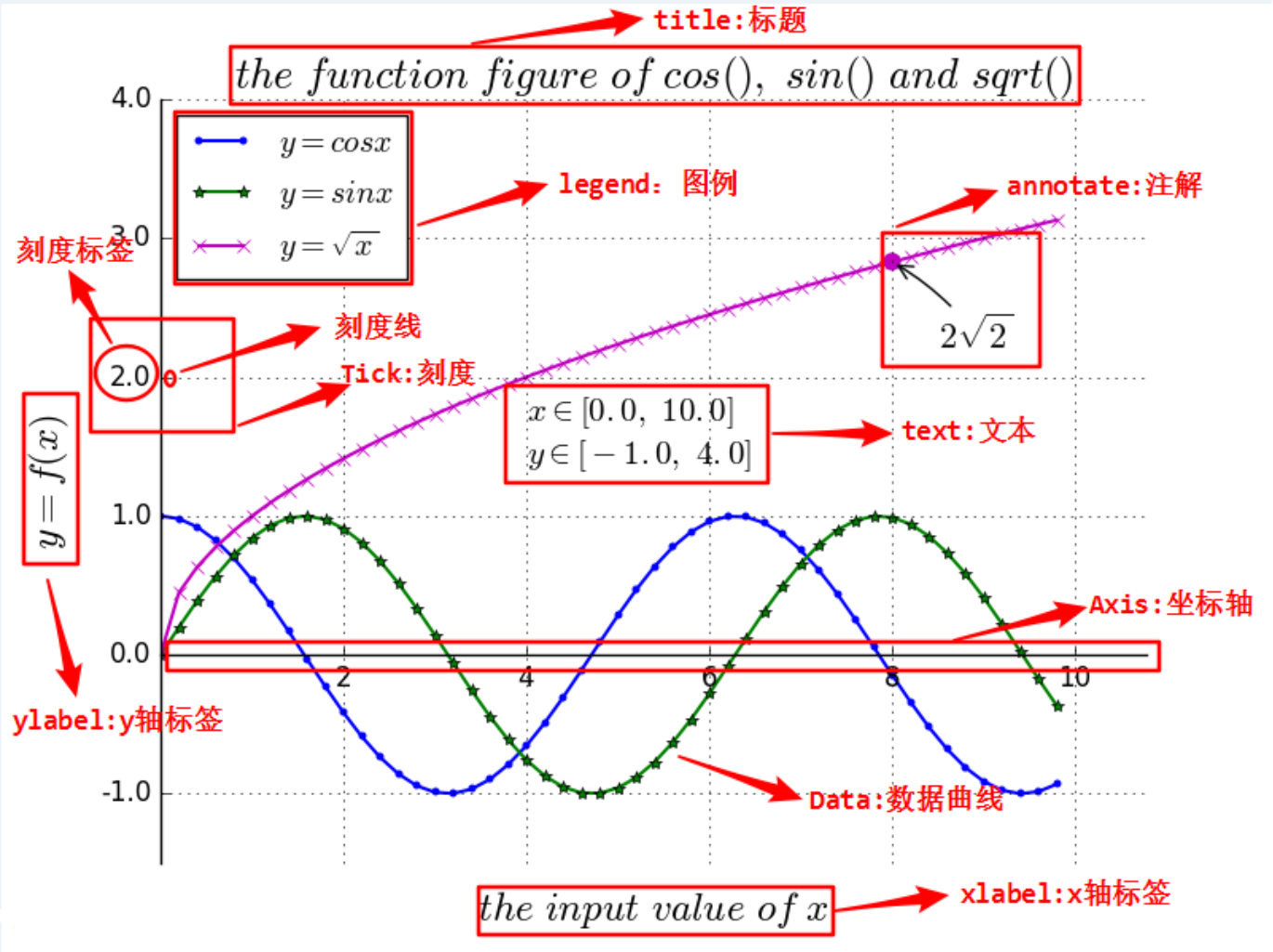
In [12]:

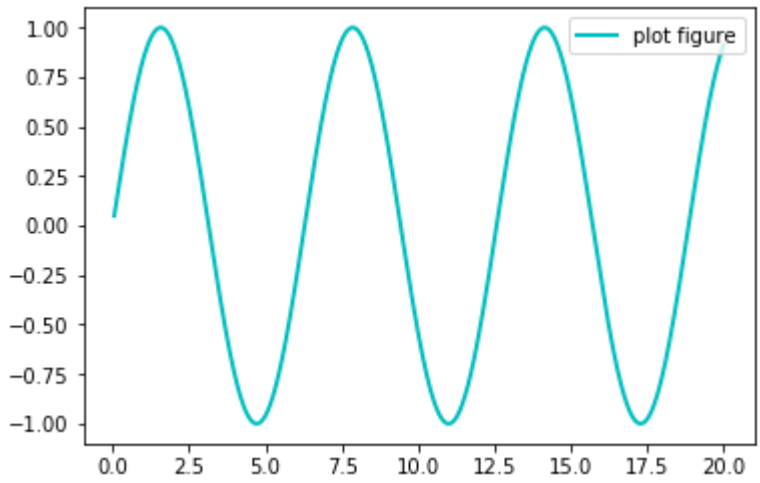
```
import matplotlib as mpl
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

# 解决坐标轴刻度负号乱码
plt.rcParams['axes.unicode_minus'] = False

# 解决中文乱码问题
plt.rcParams['font.sans-serif'] = ['Simhei']
```

1 折线图—plot()



- 函数功能：展现变量的趋势变化。
- 调用签名：plt.plot(x,y,ls="-",lw=2,label="plot figure")
- 参数说明：
  - x：x轴上的数值
  - y：y轴上的数值
  - ls：折线图的线条风格(linestyle)
  - lw：折线图的线条宽度(lineweight)
  - label：标记图形内容的标签文本
  - color：图形颜色

In [5]:
```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(0.05,20,2000)
y=np.sin(x)

fig = plt.figure()
plt.plot(x,y,ls="-",lw=2,label="plot figure",color="c")

plt.legend();        #将标签显示出来
```



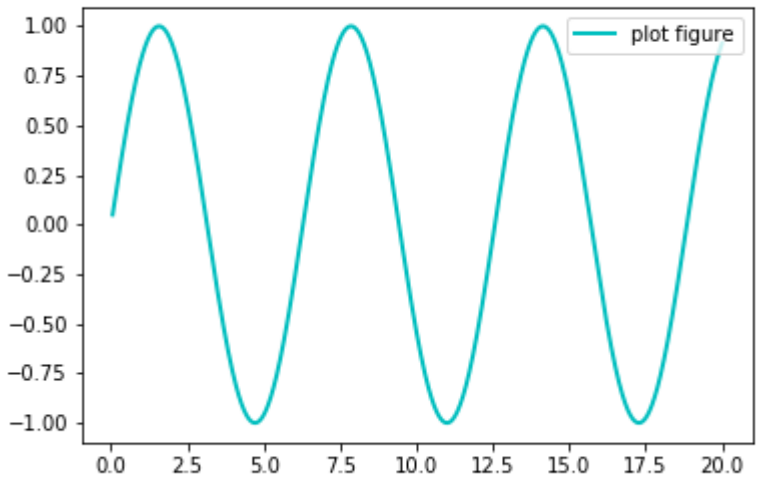
2 图形的保存

如何保存生成的图形为不同的数据格式？ 可以用**savefig()**命令将图形保存为文件。

In [6]:
```
fig.savefig('figure_01.png')
```

In [7]:
```
from IPython.display import Image
Image('figure_01.png')     #用来查看文件中是否保存了我们需要的内容
```

Out[7]:

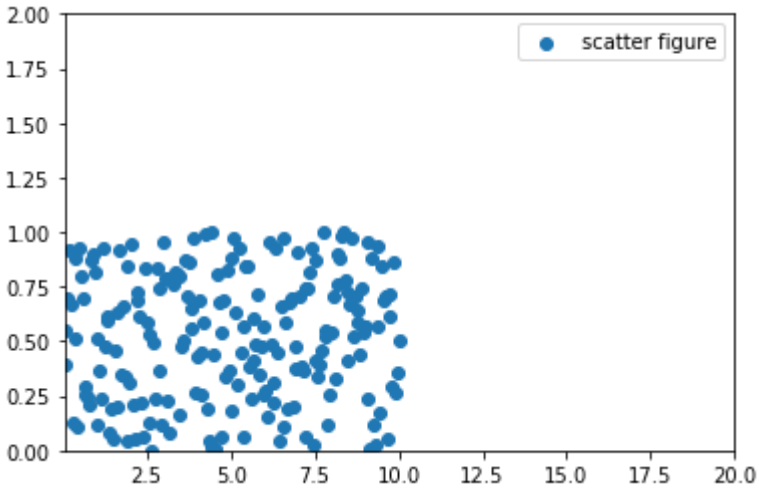


3 设置坐标轴上下限

3.1 xlim()、ylim()

- 函数功能：设置x轴的数值显示范围。
- 调用签名：plt.xlim(xmin,xmax)
- 参数说明：
  - xmin：x轴上的最小值
  - xmax：x轴上的最大值 (注意，xmin和xmax的参数位置可以调换）
  - 平移性：上面的函数功能，调用签名和参数说明同样可以平移到函数ylim()上。

In [8]:
```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(0.05,10,200)
y=np.random.rand(200)

plt.scatter(x,y,label="scatter figure")

plt.legend()

plt.xlim(0.05,20)  #设置图形显示的x轴的范围
plt.ylim(0,2)    #设置图形显示的y轴的范围

plt.show();
```

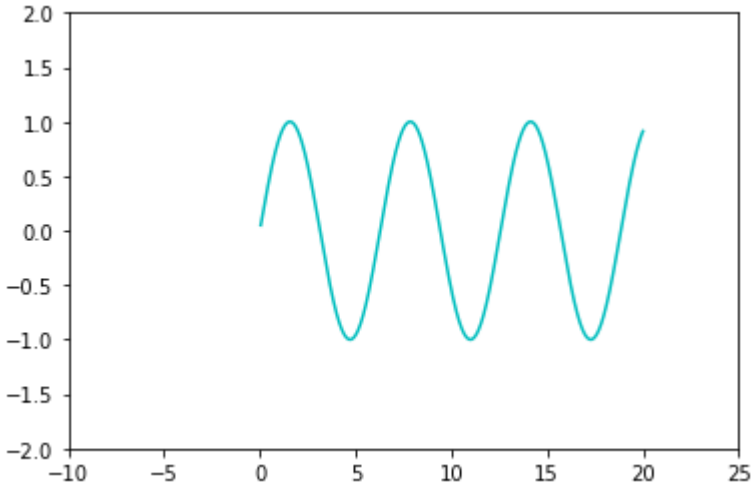


3.2 axis()

- 第一种使用方法： plt.axis([xmin,xmax,ymin,ymax])
  - xmin:x轴的最小范围值
  - xmax:x轴的最大范围值
  - ymin:y轴的最小范围值
  - ymax:y轴的最大范围值
- 第二种使用方法： plt.axis(str)
  - "on"：打开轴线和标签。
  - "off"：关闭轴线和标签。'等于'设置相等的缩放比例（即，使圆形圆形）改变轴限制。
  - "scaled"：设置相等的缩放比例（即，使圆形圆形） 改变绘图框的尺寸。
  - "tight"：设置限制大小足以显示所有数据。
  - "auto"：自动缩放（带数据的填充框）。

In [9]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(0.05,20,2000)
y=np.sin(x)


plt.plot(x,y,color="c")

plt.axis(xmin=-10,xmax=25,ymin=-2,ymax=2)

plt.show();
```



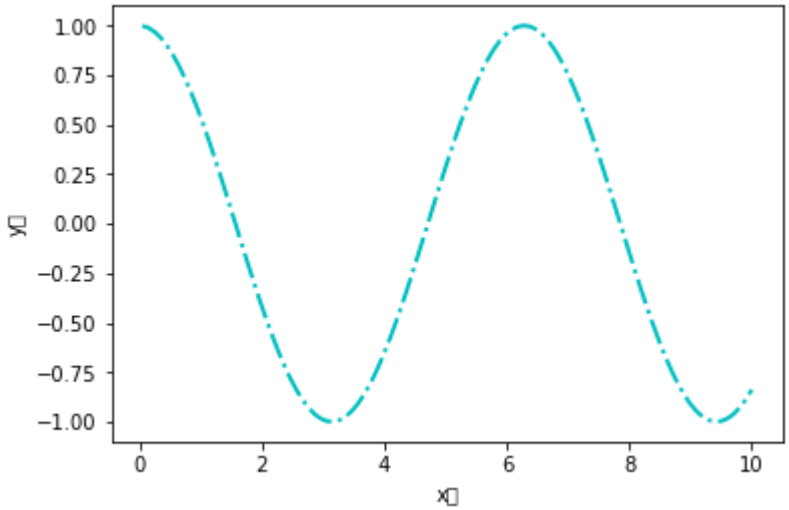
4 设置图形标签

4.1 轴标签xlabel()、ylabel()

- 函数功能：设置x轴的标签文本
- 调用签名：plt.xlabel(string)
- 参数说明：
  - string：标签文本内容
  - 平移性：上面的函数功能，调用签名和参数说明同样可以平移到函数ylabel()上。

In [10]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(0.05,10,100)
y=np.cos(x)

plt.plot(x,y,ls="-.",lw=2,c="c")

plt.xlabel("x轴")      #设置x轴的文本标签
plt.ylabel("y轴")      #设置y轴的文本标签

plt.show();
```

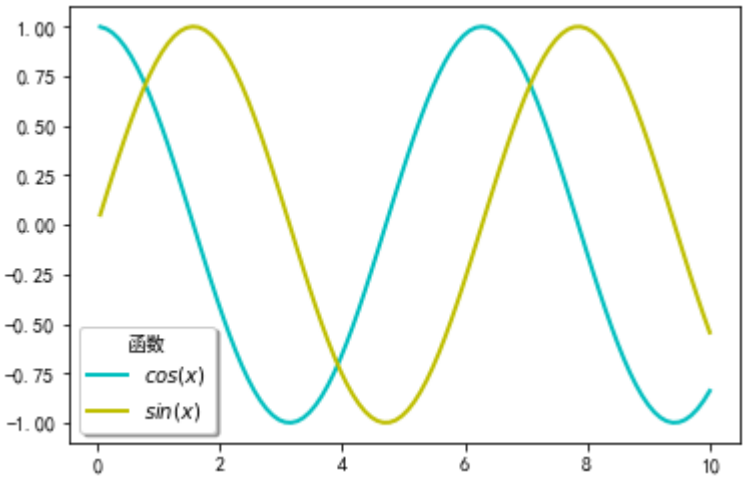


4.2 文本标签图例legend()

- 函数功能：标识不同图形的文本标签图例
- 调用签名：plt.legend(loc="lower left")
- 主要参数：
  - loc：图例在图中的地理位置。
    - upper right/1
    - upper left/2
    - lower left/3
    - lower right/4
    - center left/6
    - center right/7/5
    - lower center/8
    - upper center/9
    - center/10
- 其他参数：
  - bbox_to_anchor：线框位置参数。
    - 第一个元素：距离画布左侧x轴长度的倍数距离。
    - 第二个元素：距离画布底部y轴长度的倍数距离。
    - 第三个元素：x轴长度倍数的线框长度。
    - 第四个元素：y轴长度倍数的线框宽度。
  - title：图例标签内容标题参数。
  - shadow：线框阴影，True或者是False。
  - fancybox：线框圆角处理参数，True或者是False。

In [15]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(0.05,10,100)
y_01=np.cos(x)
y_02=np.sin(x)


#label如果想用印刷级别的文档的效果，可以使用r"$text$"模式(如果里面文本之前不加“\”，非数学文
plt.plot(x,y_01,ls="-",lw=2,c="c",label="$cos(x)$")
plt.plot(x,y_02,ls="-",lw=2,c="y",label="$sin(x)$")

plt.legend(loc="lower left",title="函数",shadow=True,fancybox=True ) #或者写loc=3

plt.show();
```

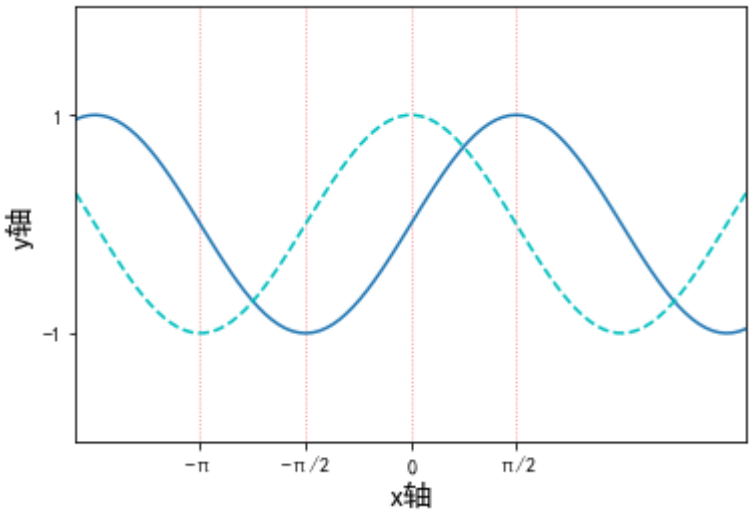


5 设置轴刻度

通过plt.xticks()和plt.yticks()两个函数来给x轴和y轴设定刻度，同时通过这两个函数中的label参数来给这些刻度贴上标签。

In [50]:

```
#导入包
import matplotlib.pyplot as plt
import numpy as np

#创建数据
x = np.linspace(-5, 5, 100)
y1 = np.sin(x)
y2 = np.cos(x)

#创建figure窗口
# plt.figure(num=1, figsize=(8, 5))

#画曲线1
plt.plot(x, y1)
#画曲线2
plt.plot(x, y2, color='c',linestyle='--')

#设置坐标轴范围
plt.xlim((-5, 5))
plt.ylim((-2, 2))

#设置坐标轴名称
plt.xlabel('x轴',size=15)
plt.ylabel('y轴',size=15)

#设置坐标轴刻度
x_ticks = np.arange(-np.pi, np.pi, np.pi/2)       #这里生成x轴刻度所需的数
y_ticks = [-1,1]                                  #这里生成y轴刻度所需的数
plt.xticks(x_ticks,labels=["-π","-π/2","0","π/2","-π"])      #这里给x轴的刻度贴上标签
plt.yticks(y_ticks)

plt.grid(linestyle=":",color="r",axis="x",alpha=0.5)
#显示出所有设置
plt.show();
```



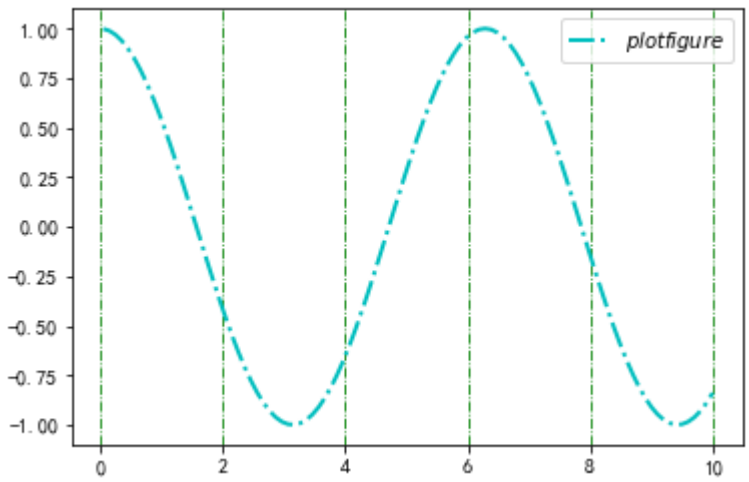
6 网络线grid()

- 函数功能：绘制刻度线的网络线
- 调用签名：plt.grid(linestyle=":",color="r")
- 参数说明

- linestyle：网格线的线条风格。
- color：网格线的线条颜色

In [24]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(0.05,10,100)
y=np.cos(x)

plt.plot(x,y,ls="-.",lw=2,c="c",label="$plot figure$")
plt.legend()

plt.grid(linestyle="-.",color="g",axis="x")    #设置绿色的、虚线的网格线，如果只想画垂直于y轴
plt.show();
```

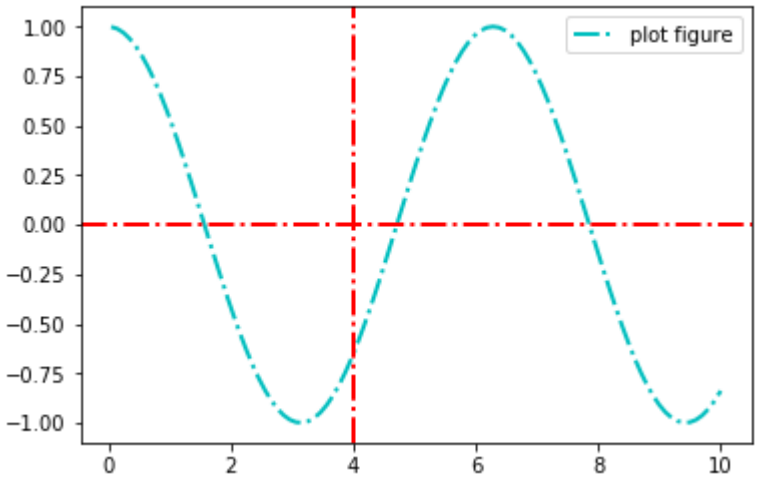


7 参考线axhline()

- 函数功能：绘制平行于x轴的水平参考线
- 调用签名：plt.axhline(y=0,c="r",ls="--",lw=2)
- 参数说明：
  - y：水平参考线的出发点
  - c：参考线的现调颜色
  - ls：参考线的线条风格
  - lw：参考线的线条宽度
  - 平移性：上面的函数功能，调用签名和参数说明同样可以平移到函数axvline()上。
- 一般函数名和方法名中，与轴相关的，带字母“h”的一般意味着是“horizontal”，带“v”一般意味着是vertical。

In [35]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(0.05,10,100)
y=np.cos(x)

plt.plot(x,y,ls="-.",lw=2,c="c",label="plot figure")

plt.legend()

plt.axhline(y=0,c="r",ls="-.",lw=2)     #画一条平行于y轴的红色虚线
plt.axvline(x=4,c="r",ls="-.",lw=2)     #画一条平行于x轴的红色虚线

plt.show();
```

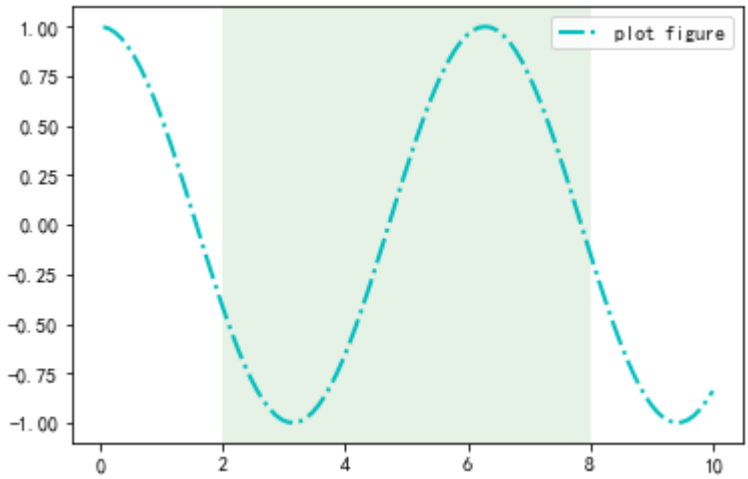


8 参考区域axvspan()

- 函数功能：绘制垂直于x轴的参考区域。
- 调用签名：plt.axvspan(xmin=1.0,xmax=2.0,facecolor="y",alpha=0.3)
- 参数说明
  - xmin：参考区域的起始位置。
  - xmax：参考区域的终止位置。
  - facecolor：参考区域的填充颜色。
  - alpha：参考区域的填充颜色的透明度。
  - 平移性：上面的函数功能，调用签名和参数说明同样可以平移到函数axvspan()上。

In [27]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(0.05,10,100)
y=np.cos(x)

plt.plot(x,y,ls="-.",lw=2,c="c",label="plot figure")

plt.legend()

plt.axvspan(xmin=2.0,xmax=8.0,facecolor="g",alpha=0.1)
# plt.axhspan(ymin=0.0,ymax=0.5,facecolor="g",alpha=0.1)

plt.show;
```



9 指向型注释annotate()

- 函数功能：添加图形内容细节的指向型注释文本
- 调用签名：

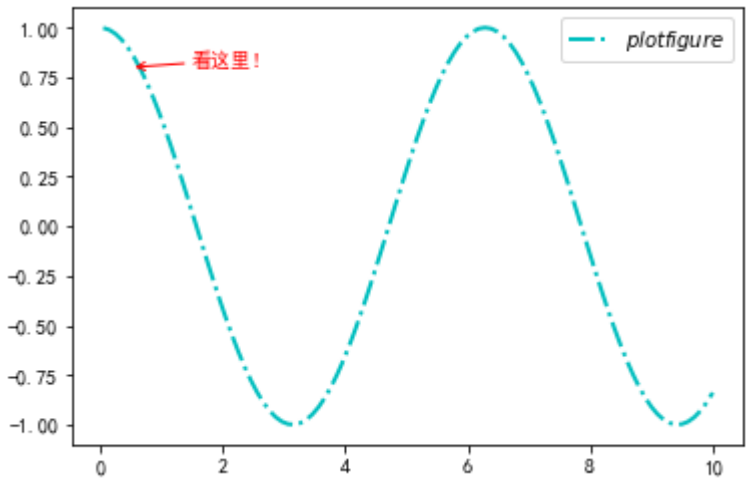
```
plt.annotate(string.
             xy=(0.5,0.8),
             xytext=(1.5,0.8),
             color="b"
             arrowprops=dict(arrowstyle="->",color="b"))
```

- 参数说明：
  - string：图形内容的注释文本。
  - xy：被注释图形内容的位置坐标。
  - xytext：注释文本的位置坐标。
  - weight：注释文本的字体粗细风格。
  - color：注释文本的字体颜色。
  - arrowprops：指示被注释内容的箭头的属性字典。
    - arrowstyle:
      - '-'
      - '->'

- '-['
- '|-|'
- '-|>'
- '<-'
- '<->'
- '<|-'
- '<|-|>'
- 'fancy'
- 'simple'
- 'wedge'

In [32]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(0.05,10,100)
y=np.cos(x)

plt.plot(x,y,ls="-.",lw=2,c="c",label="$plot figure$")

plt.legend()

plt.annotate("看这里！",          #图形内容的注释文本
             xy=(0.5,0.8),        #被注释图形内容的位置坐标
             xytext=(1.5,0.8),    #注释文本的位置坐标
             weight="bold",       #注释文本的字体粗细风格
             color="r",           #注释文本的字体颜色
             arrowprops=dict(arrowstyle="->",color="r"))

plt.show();
```



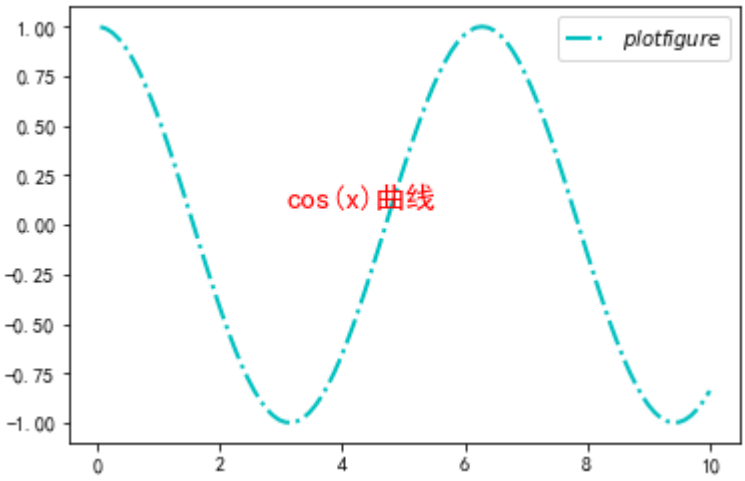
10 函数text()

- 函数功能：添加图形内容细节的无指向型注释文本
- 调用签名：plt.text(x,y,string,weight="bold",color="b)
- 参数说明：
  - x：注释文本内容所在位置的横坐标
  - y：注释文本内容所在位置的纵坐标
  - string：注释文本内容
  - fontsize：字体大小

- weight：注释文本内容的粗细风格
- color：注释文本内容的字体颜色

In [37]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(0.05,10,100)
y=np.cos(x)

plt.plot(x,y,ls="-.",lw=2,c="c",label="$plot figure$")

plt.legend()

plt.text(3.1,0.09,"cos(x)曲线",fontsize=15,weight="bold",color="r")

plt.show();
```



11 函数title()

- 函数功能：添加图形内容的标题
- 调用签名：plt.title(string)
- 参数说明：
  - string：图形内容的标题文本
  - loc：center/left/right
  - fontdict：可以用字典来存储，包含以下参数（下面的参数也可以单独在plt.title()中存在）
    - family：字体类别
    - size：字体大小
    - color：字体颜色
    - style：字体风格

In [43]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(0.05,10,100)
y=x**3

plt.plot(x,y,ls="-",lw=2,c="c",label="plot figure")

plt.legend()

fontdict={"family":"Comic Sans MS","size":18,"style":"oblique"}
plt.title("y=con(x)",loc="center",fontdict=fontdict);
```