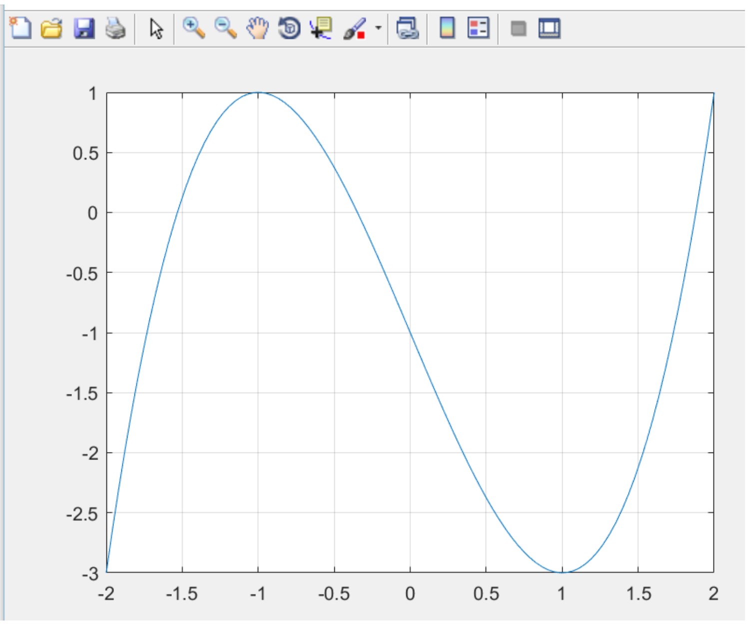
1. 牛顿切线法的迭代公式为 $x_{n+1}=x_n-\dfrac{f(x_n)}{f'(x_n)}$，如图得到对应的程序，计算结果在精度要求下与精确解 $x^*=1.87938524\cdots\cdots$ 一致，作图可以看出该方程的大致情况。

```matlab
function root = Newton( x0 )

f=@(x)x^3-3*x-1;
f1=@(x)3*x^2-3;
X(1)=x0;
X(2)=X(1)-feval(f,X(1))/feval(f1,X(1));
i=2;
while abs(X(i)-X(i-1))>10^-5
    X(i+1)=X(i)-feval(f,X(i))/feval(f1,X(i));
    i=i+1;
end
root=X(i);

end
```

```
>> f=@(x)x.^3-3.*x-1;
>> x=linspace(-2,2);
>> y=f(x);
>> plot(x,y)
>> grid on
>> Newton(2)

ans =

    1.8794

>> fprintf('10182499周天翊\n')
10182499周天翊
>>
```




2. Aitken 方法增加一步迭代来消除求导的要求，方程如下：

$$\begin{cases} \bar{x}_{k+1}=\varphi(x_k) \\ \tilde{x}_{k+1}=\varphi(\bar{x}_{k+1}) \\ x_{k+1}=\tilde{x}_{k+1}-\dfrac{(\tilde{x}_{k+1}-\bar{x}_{k+1})^2}{\tilde{x}_{k+1}-2\bar{x}_{k+1}+x_k} \end{cases}$$

程序如图，用两个中间变量 $x_1$ 和 $x_2$ 进行迭代，在满足精度要求 $10^{-4}$ 时停止迭代输出方程根 $x_0$，解得方程根约为 0.2455，需要注意的是选择迭代函数 $\varphi(x)$ 的问题，采用 $\varphi(x)=\dfrac{x^3-e^x+2}{3}$ 可以满足收敛条件。

```
>> f=@(x) (x^3-exp(x)+2)/3;
>> Aitken(f,1)

ans =

    0.2455

>>
```

```
function x0 = Aitken( f,x0 )
temp=x0;
x1=feval(f,x0);
x2=feval(f,x1);
x0=x2-(x1-x2)^2/(x2-2*x1+x0);
while abs(temp-x0)>10^-4
    temp=x0;
    x1=feval(f,x0);
    x2=feval(f,x1);
    x0=x2-(x1-x2)^2/(x2-2*x1+x0);
end
end
```

3. 弦截法和快速弦截法原理相似，迭代方程分别为：

$$x_{k+1}=x_k-\frac{f(x_k)}{f(x_k)-f(x_0)}(x_k-x_0) \qquad x_{k+1}=x_k-\frac{f(x_k)}{f(x_k)-f(x_{k-1})}(x_k-x_{k-1})$$

但在程序设计时快速弦截法要略比弦截法复杂，因为弦截法每次迭代中只有 $x_k$ 一个需要记录的变量，但快速弦截法在每次循环中需要记录两个变量 $x_k$ 和 $x_{k-1}$，原理上微妙的差别在设计程序逻辑时体现出较大的不同，如图：

```
function root = Chord( f,x0,x1,emg )
y0=feval(f,x0);
y=feval(f,x1);
while abs((x1-x0)*y/(y-y0))>emg
    y=feval(f,x1);
    x1=x1-(x1-x0)*y/(y-y0);
end
root=x1;
end
```

```
function [root,k] = Fast_chord( f,x1,x2,emg )
k=1;
y1=feval(f,x1);
y2=feval(f,x2);
x(k)=x2-(x2-x1)*y2/(y2-y1);
y(k)=feval(f,x(k));
k=k+1;
x(k)=x(k-1)-(x(k-1)-x2)*y(k-1)/(y(k-1)-y2);
while abs(x(k)-x(k-1))>emg
    y(k)=feval(f,x(k));
    x(k+1)=x(k)-(x(k)-x(k-1))*y(k)/(y(k)-y(k-1));
    k=k+1;
end
root=x(k);
end
```

最后计算结果如下，解得 $x=0.5671$，经作图验证求得的解应该比较准确。

```
>> f=@(x)x.*exp(x)-1;
>> Chord(f,0.5,0.6,10^-6)

ans =

    0.5671

>> Fast_chord(f,0.5,0.6,10^-6)

ans =

    0.5671

>> x=linspace(-2,2);
>> y=f(x);
>> plot(x,y)
>> grid on
>>
```