Méthode de Newton

Principe: Construire une suite de $(x_n)^n$, tel que x_{n+1} soit l'intersection de la tangente à la courbe de f au point (x_n , $f(x_n)$) avec l'axe des abscisses.

Conditions sur f : dans l'intervalle [a b]

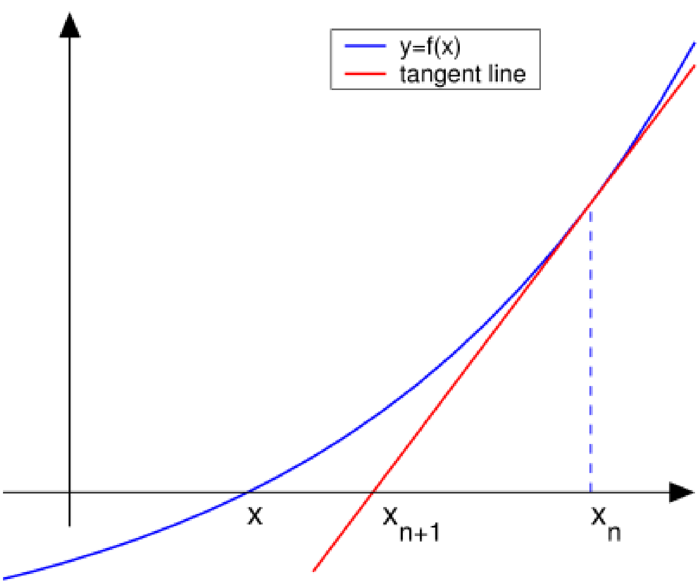
- Vérifier si f est continue
- $f(a)*f(b) < 0$
- f dérivable sur [a,b]
- $\forall$ x $\in$ [a,b] , $f'(x) \neq 0$
- $\forall$ x $\in$ [a,b] , $f''(x) \neq 0$
- f' et f'' gardent le même signe sur [a,b]

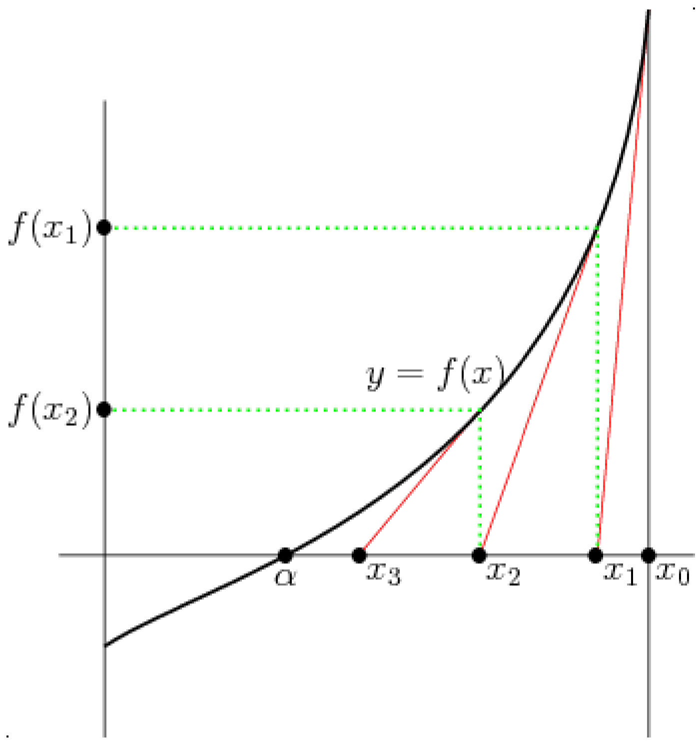
Déroulement :

1) Choisir x_n très proche de la racine de f
2) Dessiner la tangente à la courbe de f au point $(x_n, f(x_n))$
   - Etant le point d'intersection de la tangente avec l'axe des abscisses x_{n+1}, tq :

$$x_{n+1} = x_n - \frac{f(xn)}{f'(xn)}$$

3) Dessiner la tangente à la courbe de f au point $(x_{n+1}, f(x_{n+1}))$
4) Répéter l'opération jusqu'à ce que $|x_{n+1} - x_n| \leq \varepsilon$





TP1 : DICHOTOMIE

1. <u>Etude préliminaire :</u>

- Ecriture et création de la fonction f

```
function  y = f(x)

y = x/2 - sin(x) + pi/6 - sqrt(3)/2 ;
```

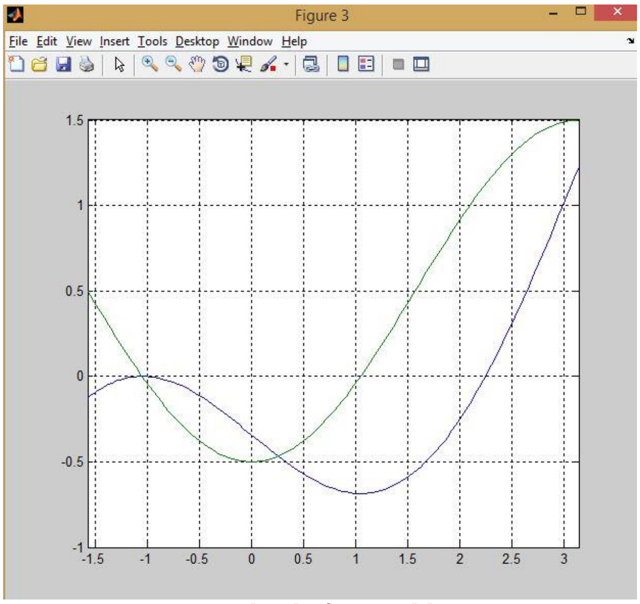
- Ecriture et création de la dérivé de f

```
function  y = df(x)

y = 1/2 - cos(x) ;
```

- Tracer les graphes de f et df

```
>> fplot ( '[ f(x),df(x)] ' , [-pi/2  pi] )
>> grid
```



Le graphe de f est en bleu

Le graphe de df est en vert

- Possibilité d'appliquer la méthode de Newton dans cet intervalle :

  La Fonction f est continue et dérivable mais la dérivé s'annule en un point donné

  *nous devons changer d'intervalle

- Proposer un intervalle qui encadre la racine et qui répond aux conditions de la méthode de Newton : I=[1.5 3]

*P .S : la racine de f est : 2.2460 (trouvé après l'exécution du script suivant)

2. <u>Script :</u>

```
x0=input('Donnez le terme initial: ');
eps=10^-10;
xr=x0-fn(x0)/dfn(x0);
Err=abs(xr-x0);
E=Err;
X=zeros(1,5);  %déclarer un vecteur à zeros qui va prendre les valeurs de xr
iter=0;
for i=10:10:50  %boucle pour avec i variant en fonction du nombre d'itérations
    j=1;
       while ((abs(xr-x0)>eps) && (j<=i))
            iter=iter+1;
            x0=xr;
            xr=x0-fn(x0)/dfn(x0);
            Err=abs(xr-x0);
            E=[E,Err]; %vecteur d'erreur
```

```
            j=j+1;
        end
         X(j)=xr;
       end

     disp(['Le nombre d''iterations = ',num2str(iter)])

     figure(1),plot(0 : iter , E), grid %graphe d'erreur en fonction de l'iter

     figure(2),plot(10:10:50,X) , grid  %graphe de xr en fonction de l'iter
```