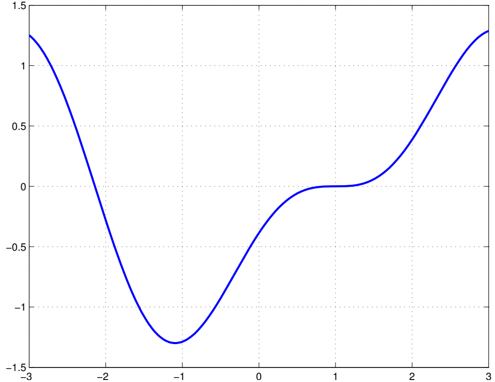
RADICI MULTIPLE

Esercizio

(esmultiple)
Calcolare con il metodo di Newton e con il metodo delle secanti le radici di $f(x) = \sin(x-1) - 0.5\sin(2(x-1))$ nell'intervallo $[-3, 3]$ e commentare i risultati ottenuti.

1. Definire la funzione e disegnarla.
2. Localizzare le radici.
3. Scegliere due dati iniziali opportuni per calcolare entrambe le radici con Newton (porre `tol=1.e-8, kmax=100`).
4. Confrontare il n. di iterazioni che sono servite per approssimare le due radici. Perchè sono così diversi?
5. Ripetere il passo 4 con metodo delle secanti. I risultati ottenuti riflettono quanto visto in teoria?



Scegliendo `x0=0.1`, Newton produce:

```
z =
    1.0000
res =
  -3.3087e-24
it =
    43
```

Scegliendo `x0=-1.5` Newton produce:

```
z =
   -2.1416
res =
  -2.4493e-16
it =
     5
```

Con x0=0.1,x1=0.4, secanti produce:

```
z =
    1.0000
res =
   -6.6174e-24
it =
    61
```

Con x0=-1.5,x1=-1.6, secanti produce:

```
z =
   -2.1416
res =
  -2.4493e-16
it =
     6
```

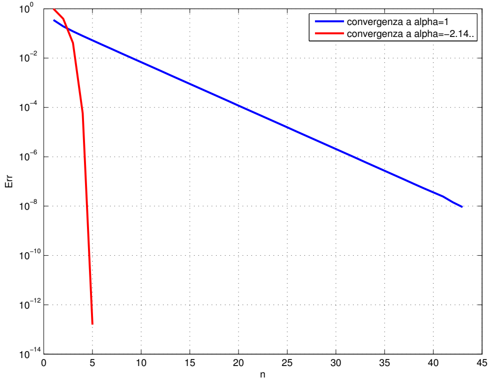
I risultati numerici riflettono quanto detto dalla teoria: sia Newton che secanti convergono linearmente verso la radice doppia $\alpha = 1$.
Per quanto riguarda la convergenza alla radice semplice, sia Newton che secanti convergono in molte meno iterazioni di prima, quindi la convergenza è più che lineare, ordine 2 per Newton e ordine $p \simeq 1.618$ per secanti. Secanti richiede una sola iterazione in più di Newton

Analisi dell'ordine di convergenza

Modifichiamo la function newton.m:
- generiamo un vettore `errore` che contenga tutti i valori $|x^{(k+1)} - x^{(k)}|$ al variare di k,
- mettiamo `errore` nella lista dei parametri di output,
- dopo aver chiamato `newton.m` rappresentiamo in scala semi-logaritmica il vettore `errore`:

```
figure;
semilogy(errore);
grid on
xlabel('k')
ylabel('errore')
```

Quando `x0=0.1` e la succ. converge a $\alpha_1 = 1$
Quando `x0=-1.5` e la succ. converge a $\alpha_2 = 1 - \pi$



Si osserva che nel primo caso, la radice è doppia e la convergenza è lenta e lineare (la curva blue è una retta)
Nel secondo caso, la radice è semplice e la convergenza è veloce e quadratica (la curva rossa è un arco di parabola)

Function `fzero` di Matlab

```
x0=0; f=@(x)x^2-3*x-4
[x,fval,exitflag] = fzero(f,x0)
```

`x` è la radice approssimata
`fval` è il residuo
`exitflag`: se è minore di zero non si è raggiunta convergenza
ATTENZIONE: `fzero` trova solo radici reali semplici, punti in cui la funzione continua cambia di segno.
Le chiamate

```
x0=-1; f=@(x)x^2-3*x+4
[x,fval,exitflag] = fzero(f,x0)

x0=-1; f=@(x)(x-3)^2
[x,fval,exitflag] = fzero(f,x0)
```

non producono risultati significativi.

Ricerca di radici complesse con Newton

Al contrario il metodo di Newton ed il metodo di secanti convergono a soluzioni complesse, se si sceglie $x_0 \in \mathbb{C}$ (per Newton) e almeno uno tra x_0 e x_1 in $\mathbb{C}$ (per secanti). Ad es:

```
f=@(x)x^2-3*x+4;
df=@(x)2*x-3;
tol=1.e-8;kmax=100;
x0=-1+i;   % x0 complessa
[z,res,it]=newton(f,df,x0,tol,kmax)
x1=-1;
[z,res,it]=secant(f,x0,x1,tol,kmax)
```

Il metodo di Newton (secanti) converge in 8 (12) iterazioni a `z=1.5000e+00 + 1.3229e+00i`.