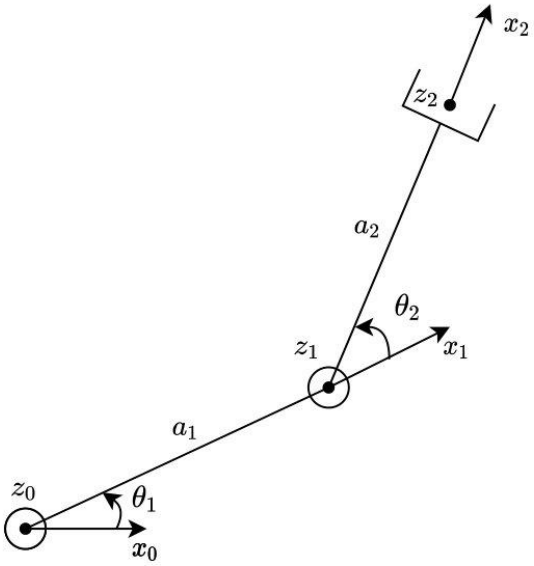
Algoritmo di Newton e del gradiente per il robot RR planare

Robot RR planare




```
clear
clc
syms l1 l2 q1 q2
DHRRplanare = [1, 0, 0, q1;
               1, 0, 0, q2]
```

DHRRplanare =

$$\begin{pmatrix} 1 & 0 & 0 & q_1 \\ 1 & 0 & 0 & q_2 \end{pmatrix}$$

```
tList = cinDirDH(DHRRplanare);
T02 = tList{3}
```

T02 =

$$\begin{pmatrix} \cos(q_1+q_2) & -\sin(q_1+q_2) & 0 & \cos(q_1+q_2)+\cos(q_1) \\ \sin(q_1+q_2) & \cos(q_1+q_2) & 0 & \sin(q_1+q_2)+\sin(q_1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
r = T02(1:2, 4)
```

r =

$$\begin{pmatrix} \cos(q_1+q_2)+\cos(q_1) \\ \sin(q_1+q_2)+\sin(q_1) \end{pmatrix}$$

```
Jgeom = jacGeom(DHRRplanare, [1 1])
```

Jgeom =

$$\begin{pmatrix} -\sin(q_1+q_2)-\sin(q_1) & -\sin(q_1+q_2) \\ \cos(q_1+q_2)+\cos(q_1) & \cos(q_1+q_2) \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{pmatrix}$$

```
Jr = Jgeom(1:2, 1:2)
```

```
Jr =
```

$$\begin{pmatrix} -\sin(q_1+q_2)-\sin(q_1) & -\sin(q_1+q_2) \\ \cos(q_1+q_2)+\cos(q_1) & \cos(q_1+q_2) \end{pmatrix}$$

```
Jinv = simplify(inv(Jr))
```

```
Jinv =
```

$$\begin{pmatrix} \frac{\cos(q_1+q_2)}{\sin(q_2)} & \frac{\sin(q_1+q_2)}{\sin(q_2)} \\ -\frac{\cos(q_1+q_2)+\cos(q_1)}{\sin(q_2)} & -\frac{\sin(q_1+q_2)+\sin(q_1)}{\sin(q_2)} \end{pmatrix}$$

Algoritmo del gradiente

```
q_i = [-1; -1]; % condizioni iniziali
rRif = [1; 1]; % riferimento
[q_f_grad, iter_grad, deltaQ_grad, currentError_grad] = algGradiente(q_i, rRif);
qEnd = q_f_grad(:, end)
```

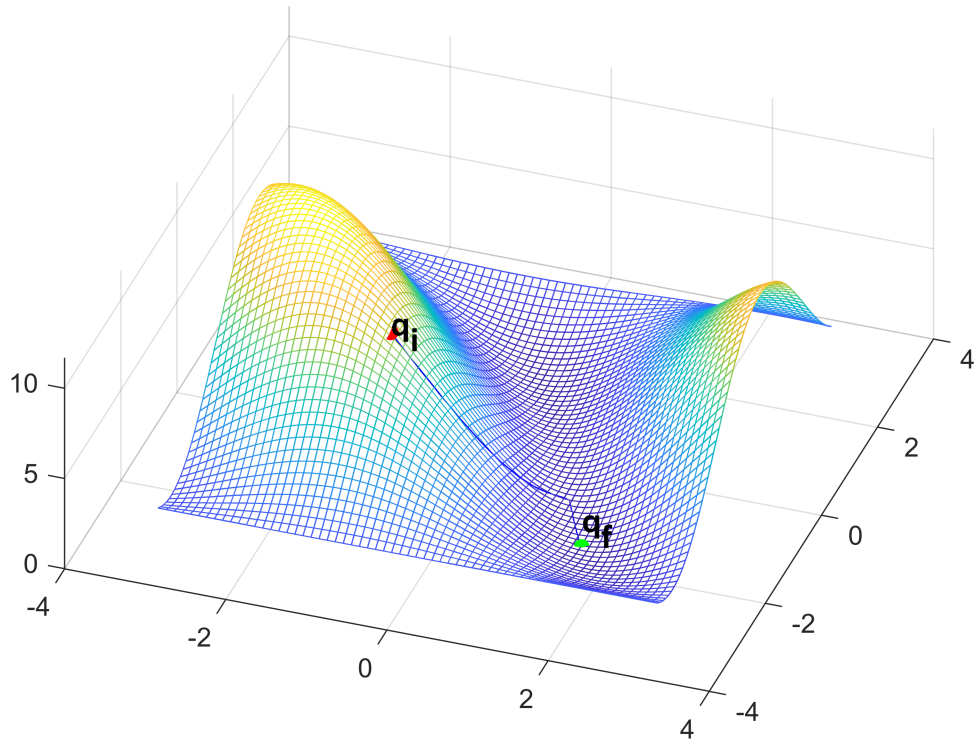
```
qEnd = 2×1
    1.5707
   -1.5706
```

```
fr = [cos(qEnd(1))+cos(qEnd(1)+qEnd(2)); +sin(qEnd(1))+sin(qEnd(1)+qEnd(2))]
```

```
fr = 2×1
    1.0001
    1.0001
```

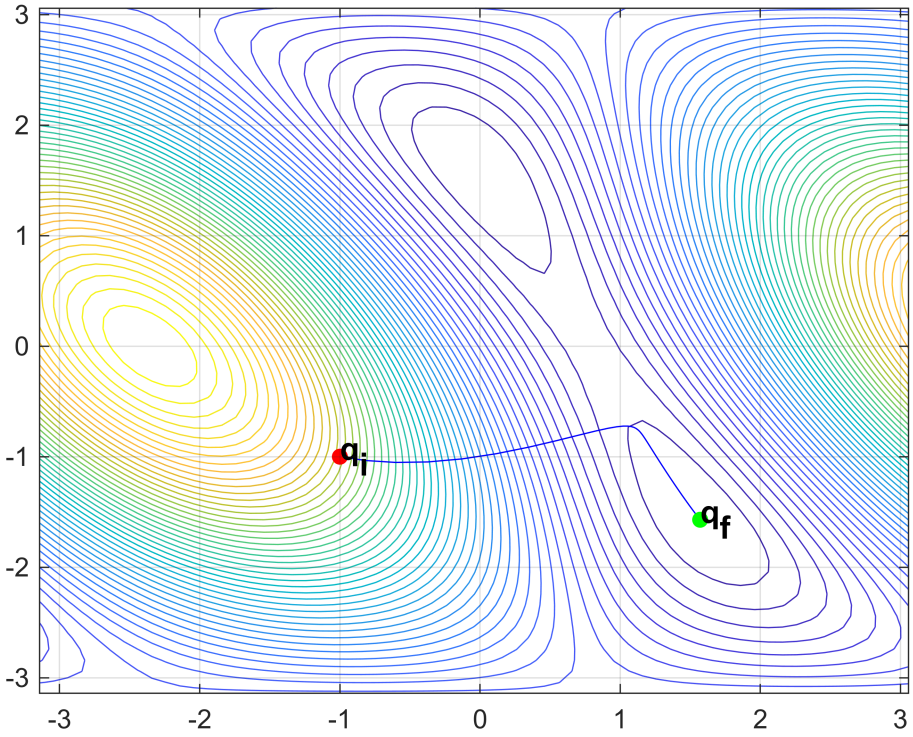
3D plot

```
plot3Dtraj(q_f_grad, rRif, 'b')
```



2D plot

```
plot2Dtraj(q_f_grad, rRif, 'b')
```



Algoritmo di Newton

```
q_i = [-1; -1]; % condizioni iniziali
rRif = [1; 1]; % riferimento
[q_f_newton, iter_newton, deltaQ_newton, currentError_newton] = algNewton(q_i, rRif);
```

```
iter = 103
deltaQ = 9.9043e-06
currentError = 6.8587e-05
```

```
qEnd = q_f_newton(:, end)
```

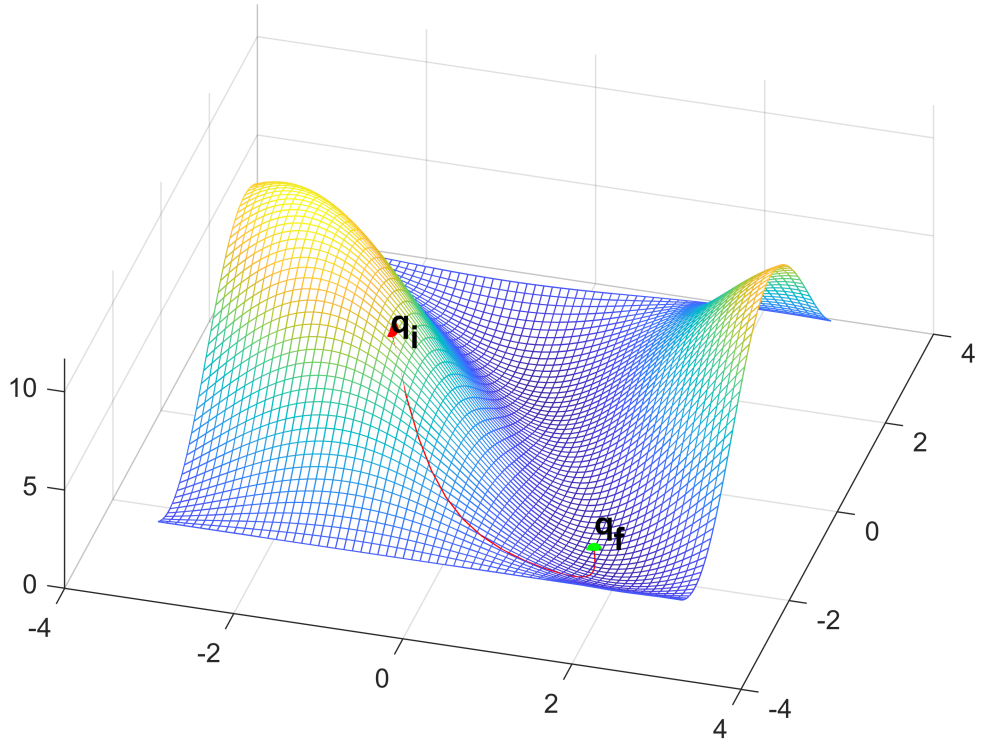
```
qEnd = 2×1
    1.5708
   -1.5709
```

```
fr = [cos(qEnd(1))+cos(qEnd(1)+qEnd(2)); +sin(qEnd(1))+sin(qEnd(1)+qEnd(2))]
```

```
fr = 2×1
    1.0000
    0.9999
```

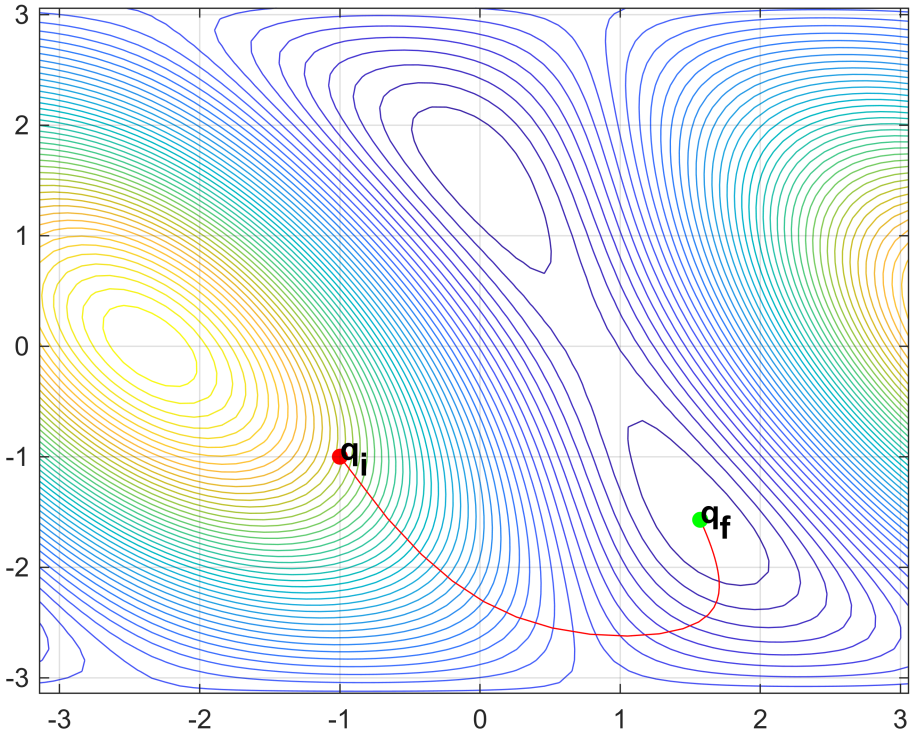
3D plot

```
plot3Dtraj(q_f_newton, rRif, 'r')
```



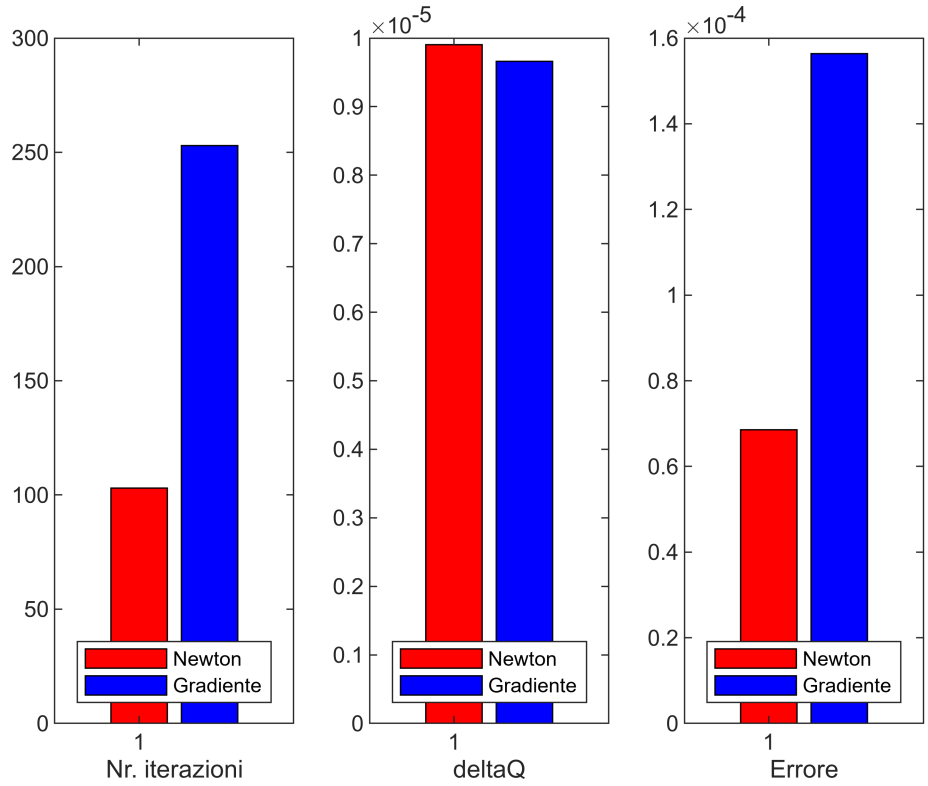
2D plot

```
plot2Dtraj(q_f_newton, rRif, 'r')
```

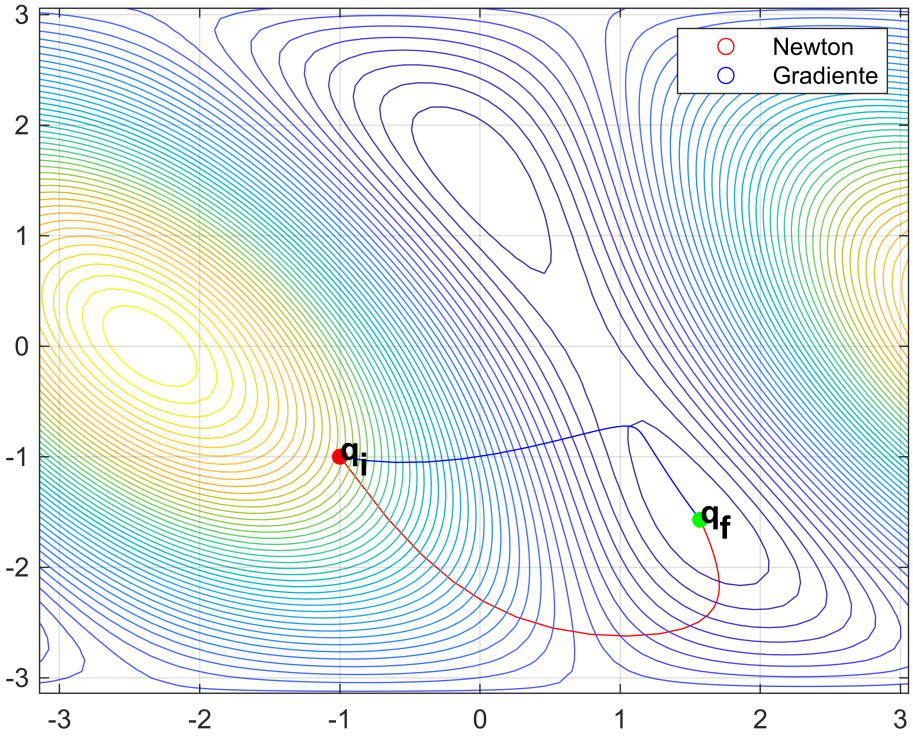


Confronto

```
figure;
subplot(1,3,1);
b = bar(1, iter_newton, 'FaceColor', 'r');
hold on
b = bar(2, iter_grad, 'FaceColor', 'b');
legend('Newton','Gradiente')
xlabel('Nr. iterazioni');
legend('Location', 'south');

subplot(1,3,2);
b = bar(1, deltaQ_newton, 'FaceColor', 'r');
hold on
b = bar(2, deltaQ_grad, 'FaceColor', 'b');
legend('Newton','Gradiente')
xlabel('deltaQ');
legend('Location', 'south');

subplot(1,3,3);
b = bar(1, currentError_newton, 'FaceColor', 'r');
hold on
b = bar(2, currentError_grad, 'FaceColor', 'b');
legend('Newton','Gradiente')
xlabel('Errore');
legend('Location', 'south');
```




```
figure
plot2Dtraj(q_f_newton, rRif, 'r')
hold on
plot2Dtraj(q_f_grad, rRif, 'b')
hold on
h = zeros(2, 1);
h(1) = plot(NaN,NaN,'or');
h(2) = plot(NaN,NaN,'ob');
legend(h,'Newton','Gradiente')
hold off
```



Studio punti estremanti

```
e2v = (rRif-r).^2;
e2 = e2v(1) + e2v(2)
```

e2 = $(\cos(q_1+q_2)+\cos(q_1)-1)^2+(\sin(q_1+q_2)+\sin(q_1)-1)^2$

```
g = gradient(e2, [q1, q2])
```

g =

$$\begin{pmatrix} 2\,(\cos(q_1+q_2)+\cos(q_1))\,\sigma_1-2\,(\sin(q_1+q_2)+\sin(q_1))\,\sigma_2 \\ 2\cos(q_1+q_2)\,\sigma_1-2\sin(q_1+q_2)\,\sigma_2 \end{pmatrix}$$

where

$$\sigma_1 = \sin(q_1+q_2)+\sin(q_1)-1$$

$$\sigma_2 = \cos(q_1+q_2)+\cos(q_1)-1$$

```
sol = solve(g==0, [q1, q2])
```

```
sol = struct with fields:
    q1: [5×1 sym]
    q2: [5×1 sym]
```

```
estr = [sol.q1, sol.q2]
```

estr =

$$\begin{pmatrix} 0 & \frac{\pi}{2} \\ \frac{\pi}{4} & 0 \\ -\frac{3\,\pi}{4} & 0 \\ \frac{\pi}{2} & -\frac{\pi}{2} \\ \frac{\pi}{4} & \pi \end{pmatrix}$$

```
H = hessian(e2, [q1, q2])
```

H =

$$\begin{pmatrix} 2\,{\sigma_3}^2 - 2\,\sigma_3\,\left(\cos(q_1+q_2)+\cos(q_1)-1\right) - 2\,\sigma_2\,\left(\sin(q_1+q_2)+\sin(q_1)-1\right) + 2\,{\sigma_2}^2 & \\ \sigma_1 & 2\cos(q_1+q_2)^2 - \sigma_4 \end{pmatrix}$$

where

$$\sigma_1 = 2\sin(q_1+q_2)\,\sigma_2 - \sigma_4 + 2\cos(q_1+q_2)\,\sigma_3 - \sigma_5$$

$$\sigma_2 = \sin(q_1+q_2)+\sin(q_1)$$

$$\sigma_3 = \cos(q_1+q_2)+\cos(q_1)$$

$$\sigma_4 = 2\sin(q_1+q_2)\,\left(\sin(q_1+q_2)+\sin(q_1)-1\right)$$

$$\sigma_5 = 2\cos(q_1+q_2)\,\left(\cos(q_1+q_2)+\cos(q_1)-1\right)$$

```
nrSol = size(estr, 1)
```

nrSol = 5

```
for i=1:nrSol
    Hp = subs(H, [q1, q2], estr(i, :));
    %disp("Autovalori soluzione " + i)
    autovalori = simplify(eig(Hp));

    if(autovalori > 0)
        % hessiana definita positiva => punto di minimo
        estr(i, :)
        disp(" è punto di minimo")
    elseif(autovalori < 0)
        % hessiana definita negativa => punto di massimo
```

```
        estr(i, :)
        disp(" è punto di massimo")
    elseif(any(autovalori > 0) && any(autovalori < 0))
        % hessiana indefinita => punto di sella
        estr(i, :)
        disp(" è punto di sella")
    else
        % hessiana semidefinita positiva/negativa => punto estremante
        % indefinito
        estr(i, :)
        disp(" è punto estremante indefinito")
    end
end
```

```
ans =
```

$$\left(0 \quad \frac{\pi}{2}\right)$$

```
 è punto di minimo
ans =
```

$$\left(\frac{\pi}{4} \quad 0\right)$$

```
 è punto di sella
ans =
```

$$\left(-\frac{3\pi}{4} \quad 0\right)$$

```
 è punto di massimo
ans =
```

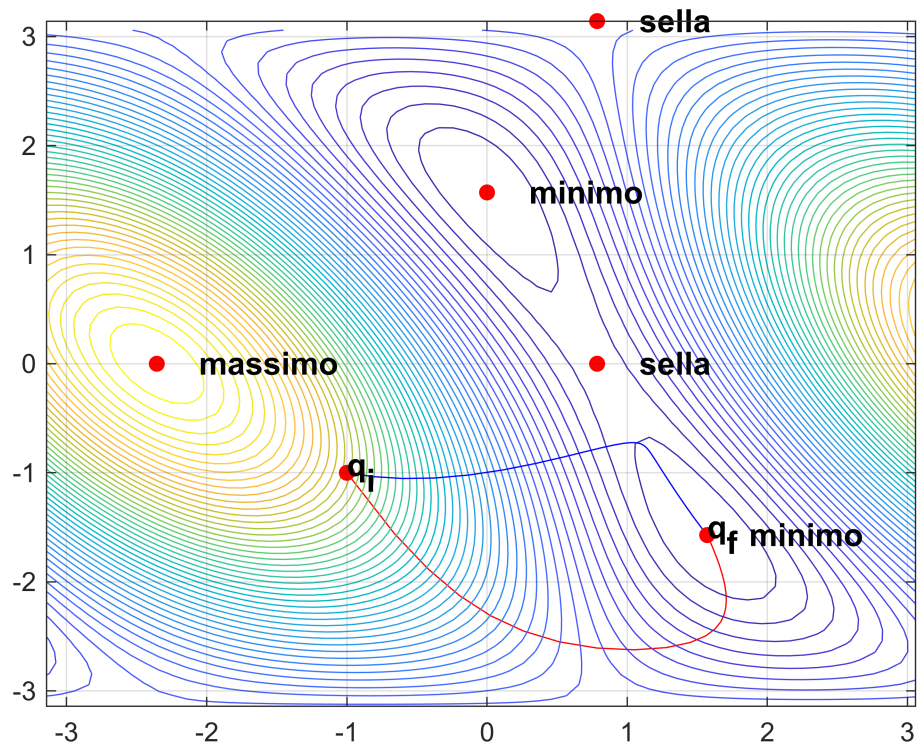
$$\left(\frac{\pi}{2} \quad -\frac{\pi}{2}\right)$$

```
 è punto di minimo
ans =
```

$$\left(\frac{\pi}{4} \quad \pi\right)$$

```
 è punto di sella
```

```
figure
plot2Dtraj(q_f_newton, rRif, 'r')
hold on
plot2Dtraj(q_f_grad, rRif, 'b')
hold on
mappa = ["minimo", "sella", "massimo", "minimo", "sella"];

for i=1:nrSol
    q_estr = estr(i, :);
    scatter(q_estr(1), q_estr(2), 'filled', 'MarkerFaceColor', 'r');
    text(q_estr(1)+0.3, q_estr(2), mappa(i), 'FontSize', 12, 'FontWeight', 'bold');
end
legend off
hold off
```



Le singolarità si ottengono per $\det(\mathbf{J}) = 0$, e dato che

```
detJr = simplify(det(Jr))
```

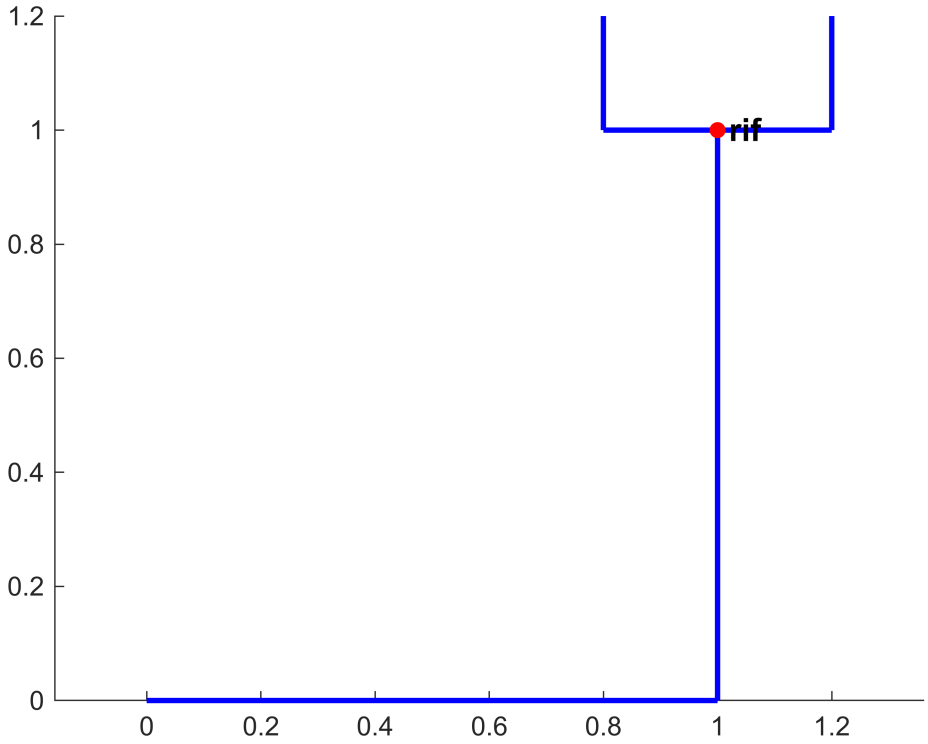
```
detJr = sin(q2)
```

si hanno singolarità per $q_2 = 0 \vee q_2 = \pi$. I punti di sella e di massimo trovati sono tutti in singolarità.

```
for i=1:nrSol
    Jrq = subs(Jr, [q1, q2], estr(i, :));
    disp("Nucleo di  Jr: ")
    nucleo = null(Jrq)
    display("Rango di Jr per il punto estremante " + mat2str(string(estr(i, :))) ...
        + " di " + mappa(i) + ": rank(Jr)=" + rank(Jrq))
    figure
    drawPlanarRobot([1,1],estr(i, :))
    scatter(rRif(1), rRif(2), 'filled', 'MarkerFaceColor', 'r');
    text(rRif(1)+0.02, rRif(2), "rif", 'FontSize', 12, 'FontWeight', 'bold');
    axis equal
end
```

```
Nucleo di  Jr:

nucleo =

Empty sym: 2-by-0
```

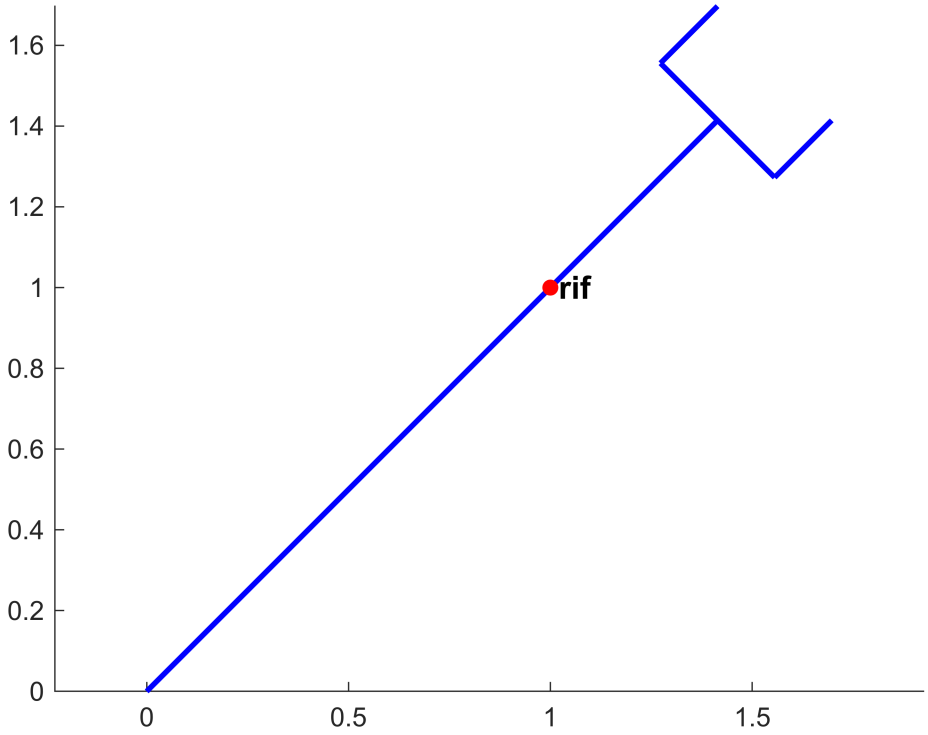
"Rango di Jr per il punto estremante ["0" "pi/2"] di minimo: rank(Jr)=2"



Nucleo di Jr:
nucleo =

$$\begin{pmatrix} -\frac{1}{2} \\ 1 \end{pmatrix}$$

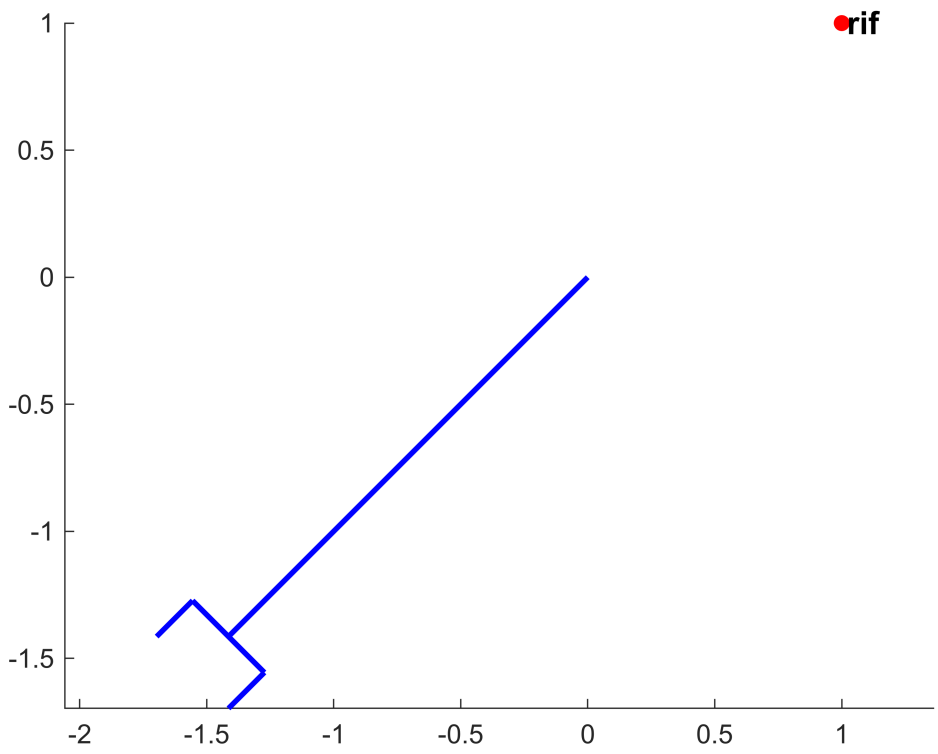
"Rango di Jr per il punto estremante ["pi/4" "0"] di sella: rank(Jr)=1"



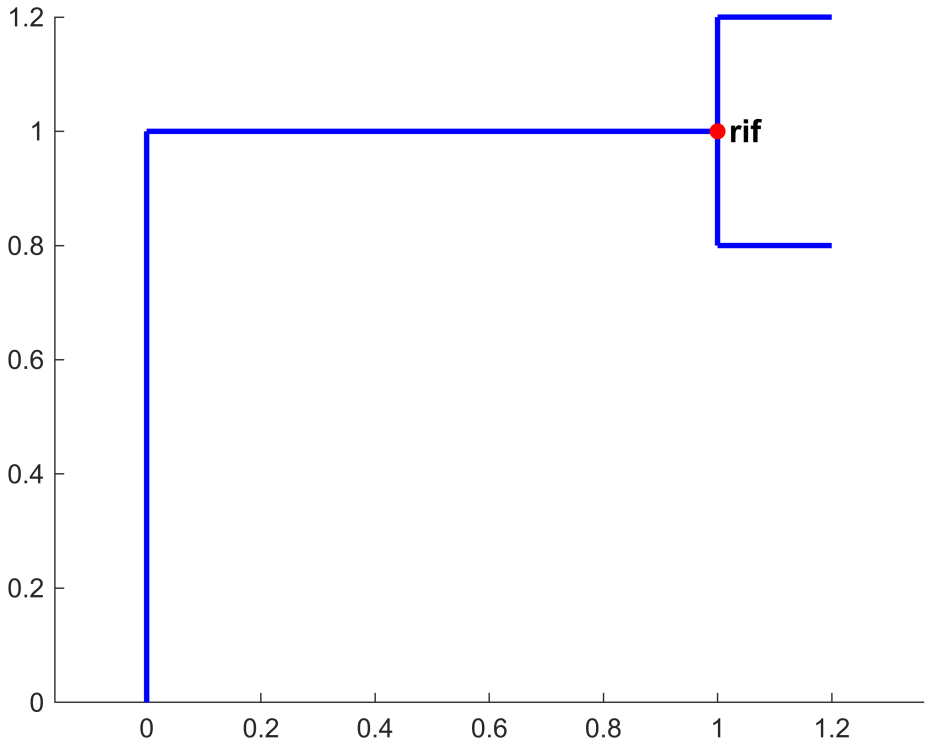

```
Nucleo di  Jr:
nucleo =
```

$$\begin{pmatrix} -\frac{1}{2} \\ 1 \end{pmatrix}$$

```
    "Rango di Jr per il punto estremante ["-(3*pi)/4" "0"] di massimo: rank(Jr)=1"
```



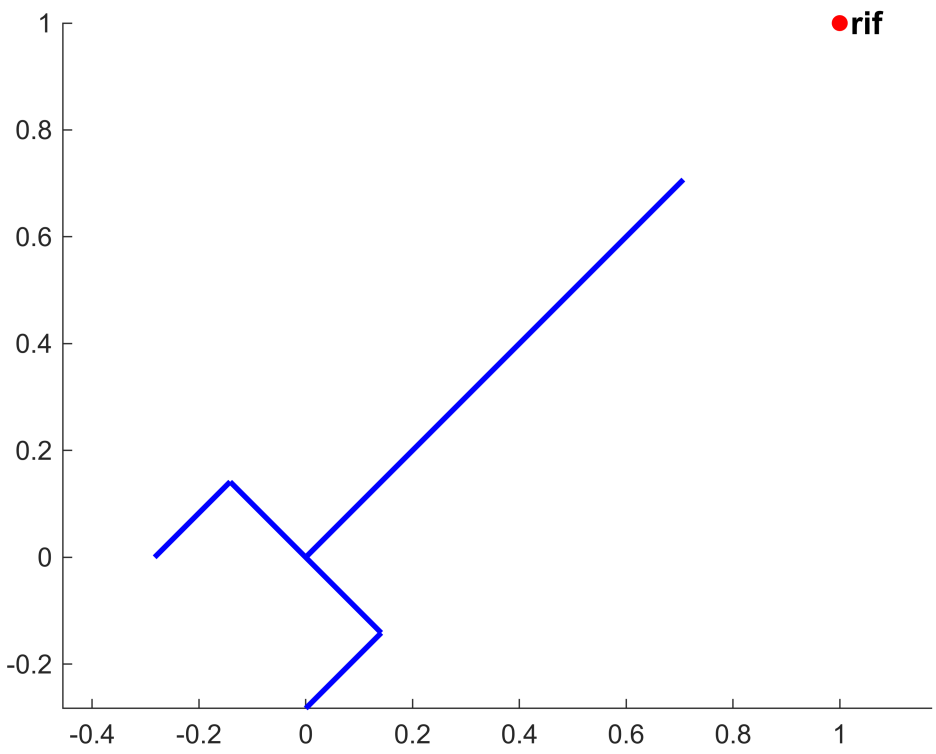

```
Nucleo di  Jr:

nucleo =

Empty sym: 2-by-0

    "Rango di Jr per il punto estremante ["pi/2" "-pi/2"] di minimo: rank(Jr)=2"
```




```
Nucleo di  Jr:
nucleo =
```

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

```
    "Rango di Jr per il punto estremante ["pi/4" "pi"] di sella: rank(Jr)=1"
```



Test con punti estremanti come condizione iniziale

Algoritmo del gradiente

```
rRif = [1; 1]; % riferimento
for i=1:nrSol
    q_i = estr(i, :).'; % condizioni iniziali
    display("Condizione iniziale: q_i=" + mat2str(string(estr(i, :))))
    [q_f_grad, iter_grad, deltaQ_grad, currentError_grad] = algGradiente(q_i, rRif)
    qEnd = q_f_grad(:, end)
    fr = [cos(qEnd(1))+cos(qEnd(1)+qEnd(2)); +sin(qEnd(1))+sin(qEnd(1)+qEnd(2))]
    %plot3Dtraj(q_f_grad, rRif)
    %figure(i)
    %plot2Dtraj(q_f_grad, rRif)
    %hold off
end
```

```
    "Condizione iniziale: q_i=["0" "pi/2"]"
q_f_grad =
```

$$\begin{pmatrix} 0 & 0 \\ \frac{\pi}{2} & \frac{\pi}{2} \end{pmatrix}$$

```
iter_grad = 2
deltaQ_grad = 0
currentError_grad = 0
qEnd =
```

$$\begin{pmatrix} 0 \\ \frac{\pi}{2} \end{pmatrix}$$

fr =

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

"Condizione iniziale: q_i=["pi/4" "0"]"

q_f_grad =

$$\begin{pmatrix} \frac{\pi}{4} & \frac{\pi}{4} \\ 0 & 0 \end{pmatrix}$$

iter_grad = 2

deltaQ_grad = 0

currentError_grad =

$$\sqrt{2\,\left(\sqrt{2}-1\right)^2}$$

qEnd =

$$\begin{pmatrix} \frac{\pi}{4} \\ 0 \end{pmatrix}$$

fr =

$$\begin{pmatrix} \sqrt{2} \\ \sqrt{2} \end{pmatrix}$$

"Condizione iniziale: q_i=["-(3*pi)/4" "0"]"

q_f_grad =

$$\begin{pmatrix} -\frac{3\,\pi}{4} & -\frac{3\,\pi}{4} \\ 0 & 0 \end{pmatrix}$$

iter_grad = 2

deltaQ_grad = 0

currentError_grad =

$$\sqrt{2\,\left(\sqrt{2}+1\right)^2}$$

qEnd =

$$\begin{pmatrix} -\frac{3\,\pi}{4} \\ 0 \end{pmatrix}$$

fr =

$$\begin{pmatrix} -\sqrt{2} \\ -\sqrt{2} \end{pmatrix}$$

"Condizione iniziale: q_i=["pi/2" "-pi/2"]"

q_f_grad =

$$\begin{pmatrix} \frac{\pi}{2} & \frac{\pi}{2} \\ -\frac{\pi}{2} & -\frac{\pi}{2} \end{pmatrix}$$

iter_grad = 2

deltaQ_grad = 0

currentError_grad = 0

qEnd =

$$\begin{pmatrix} \frac{\pi}{2} \\ -\frac{\pi}{2} \end{pmatrix}$$

fr =

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

"Condizione iniziale: q_i=["pi/4" "pi"]"

q_f_grad =

$$\begin{pmatrix} \frac{\pi}{4} & \frac{\pi}{4} \\ \pi & \pi \end{pmatrix}$$

iter_grad = 2

deltaQ_grad = 0

currentError_grad = $\sqrt{2}$

qEnd =

$$\begin{pmatrix} \frac{\pi}{4} \\ \pi \end{pmatrix}$$

fr =

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Agoritmo di Newton

```
rRif = [1; 1]; % riferimento
for i=1:nrSol
    q_i = estr(i, :).'; % condizioni iniziali
    display("Condizione iniziale: q_i=" + mat2str(string(estr(i, :))))
    [q_f_newton, iter_newton, deltaQ_newton, currentError_newton] = algNewton(q_i, rRif);
    qEnd = q_f_newton(:, end)
    fr = [cos(qEnd(1))+cos(qEnd(1)+qEnd(2)); +sin(qEnd(1))+sin(qEnd(1)+qEnd(2))]
    %plot3Dtraj(q_f_newton, rRif)
    %figure
    %plot2Dtraj(q_f_newton, rRif)
    %hold off
end
```

"Condizione iniziale: q_i=["0" "pi/2"]"

iter = 2

deltaQ = 0

currentError = 0

qEnd =

$$\begin{pmatrix} 0 \\ \frac{\pi}{2} \end{pmatrix}$$

fr =

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

"Condizione iniziale: q_i=["pi/4" "0"]"

iter = 3

deltaQ = NaN

currentError = NaN

qEnd =

$$\begin{pmatrix} \mathrm{NaN} \\ \mathrm{NaN} \end{pmatrix}$$

fr =

$$\begin{pmatrix} \mathrm{NaN} \\ \mathrm{NaN} \end{pmatrix}$$

"Condizione iniziale: q_i=["-(3*pi)/4" "0"]"

iter = 3

deltaQ = NaN

currentError = NaN

qEnd =

$$\begin{pmatrix} \mathrm{NaN} \\ \mathrm{NaN} \end{pmatrix}$$

fr =

$$\begin{pmatrix} \mathrm{NaN} \\ \mathrm{NaN} \end{pmatrix}$$

"Condizione iniziale: q_i=["pi/2" "-pi/2"]"

iter = 2

deltaQ = 0

currentError = 0

qEnd =

$$\begin{pmatrix} \frac{\pi}{2} \\ -\frac{\pi}{2} \end{pmatrix}$$

fr =

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

"Condizione iniziale: q_i=["pi/4" "pi"]"

iter = 2

deltaQ = NaN

currentError = $\sqrt{2}$

qEnd =

$$\begin{pmatrix} -\infty \\ \mathrm{NaN} \end{pmatrix}$$

```
fr =
```

$$\begin{pmatrix} \mathrm{NaN} \\ \mathrm{NaN} \end{pmatrix}$$