```
function [z, v, info, StimaErr, succ]=newtonNd(f, x0, d1f, E, kmax)
//
// Metodo di Newton per funzioni da R^n in R^n.
//
// Approssima uno zero della funzione f: R^n -> R^n con differenziale
// d1f costruendo una successione x(k) a partire da x0.
//
// La costruzione della successione si arresta quando:
//    (*) la funzione f si annulla nel punto attuale x(k);
//    (*) || x(k+1) - x(k) || risulta minore di E (in tal
//         caso, se la successione fosse convergente ad uno zero semplice
//         di f, l'errore assoluto tra z = x(k) e tale zero sarebbe
//         circa pari ad E);
//    (*) dopo kmax iterazioni (valore predefinito: kmax = 20).
//
//
// z: approssimazione finale (zero di f oppure una sua approssimazione);
// v: valore di f in z;
// info = 0: individuato valore in cui f si annulla;
//      = 1: criterio di arresto soddisfatto;
//      = 2: d1f(x(k)) non invertibile;
//      = 3: superato il numero massimo consentito di iterazioni.
// StimaErr: vettore riga contenente le stime dell'errore calcolate
//           a ciascuna iterazione.
// succ: matrice di colonne x(0), ... ,x(k).
//
//
if ~exists('kmax','l') then kmax = 20; end;
succ = x0;
StimaErr = [];
v_f = f(succ(:,$));
v_d = d1f(succ(:,$));
[S,D,P] = lu(v_d); // fattorizzazione EGPP di d1f(x(k))
if (norm(v_f) ~= 0 & det(D) ~= 0) then passo = -D \ (S \ (P * v_f)); StimaErr = norm(passo); end;
//
// ****************** Iterazione
//
while (norm(v_f) ~= 0 & det(D) ~= 0 & StimaErr($) >= E & (size(succ,'c') - 1) < kmax),
   succ(:,$+1) = succ(:,$) + passo;
   v_f = f(succ(:,$));
   v_d = d1f(succ(:,$));
   [S,D,P] = lu(v_d);
   if det(D) ~= 0 then passo = -D \ (S \ (P * v_f)); StimaErr($+1) = norm(passo); end;
end;
//
// ****************** Fine iterazione
//
z = succ(:,$); v = v_f; info = 1;
if norm(v_f) == 0 then info = 0;
   else if det(D) == 0 then info = 2;
           else if (size(succ,'c') - 1) == kmax then info = 3; end;
        end;
end;
endfunction
//
```

```
//
// Test per newtonNd: zeri della funzione f_test:
//
//     f_test(x) = [ x(1)^2 - x(2) + g;
//                  -x(1) + x(2)^2 + g]
//
// (intersezione di due parabole).
//
// Parte 1: assegnare g = 0. Soluzioni esatte: (0,0) e (1,1).
// (a) Assegnare x0 = [1;-1]. Disegnare il grafico per la stima dell'ordine di
//     convergenza, ovvero il grafico di log(stima(k+1)) / log(stima(k)) in
//     funzione di k. Calcolare poi la matrice jacobiana di f_test in (0,0)
//     e verificare che risulta non singolare.
// (b) Assegnare x0 = [1;0]. Verificare che il metodo supera il numero di
//     iterazioni massimo e che la successione calcolata risulta periodica
//     (quindi non serve aumentare il numero massimo di iterazioni...).
// (c) Cercare x0 in modo che la successione abbia limite (1,1),
//     costruire il grafico della stima dell'ordine di convergenza, calcolare
//     la matrice jacobiana di f_test in (1,1) e verificare che risulta non
//     singolare.
//
// Parte 2: assegnare g = 1/4. Soluzione esatta: (1/2,1/2).
//     Assegnare x0 = [1;-1]. Costruire il grafico per la stima dell'ordine di
//     convergenza, calcolare la matrice jacobiana di f_test in (1/2,1/2) e
//     verificare che risulta singolare. Calcolare il determinante della
//     matrice jacobiana di f_test nel punto z (ovvero det(d1f(z))).
//
```

<u>Oss</u> :

- $n=1$, h regolare, α p.u. di h :

$$h(x) = h(\alpha) + h'(\alpha)(x-\alpha) + \frac{1}{2}h''(\alpha)(x-\alpha)^2 + \dots$$

<u>SE</u> $h'(\alpha) \neq 0$ (odc $= 1$) : per k grande

$$\boxed{|x_{k+1} - \alpha| \simeq |h'(\alpha)|\,|x-\alpha|}$$

<u>SE</u> $h'(\alpha) = 0$ e $h''(\alpha) \neq 0$ (odc $= 2$) :

per k grande $\boxed{|x_{k+1} - \alpha| \simeq \left|\frac{1}{2}h''(\alpha)\right| |x_k - \alpha|^2}$

- d_k success. di numeri reali > 0 t.c.

(1) $d_k \to 0$

(2) $\exists\, m > 0$ t.c. $\lim_{k \to \infty} \frac{d_{k+1}}{d_k^{\,m}} = c > 0$

<u>Allora</u> : per k grande $d_{k+1} \simeq c\, d_k^{\,m}$

e $$\lim_{k\to\infty} \frac{\ln d_{k+1}}{\ln d_k} = \lim_{k\to\infty}\Big(\underbrace{\frac{\ln c}{\ln d_k}}_{\to 0} + m\Big) = m$$

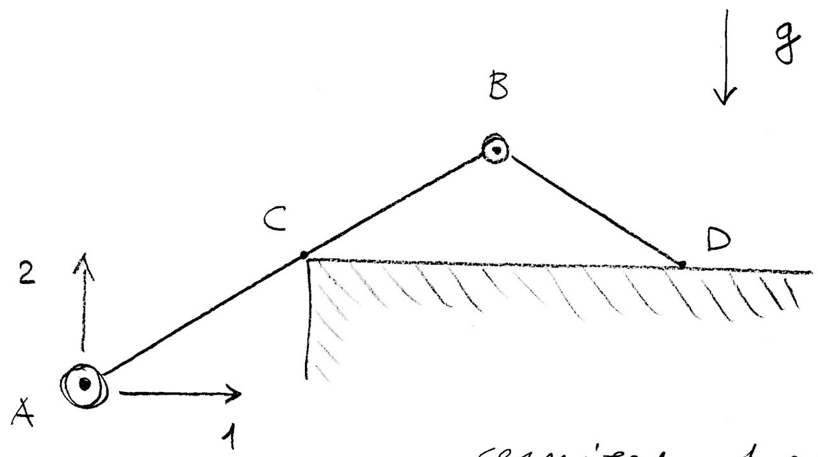
- $\boxed{m = \text{odc}}$

* SISTEMI di EQUAZIONI LINEARI *

Es :

- In ciascuna iterazione del m di Newton per $F : \mathbb{R}^n \to \mathbb{R}^n$
- Pb. statica lineare (circuiti elettrici lineari, sistemi di corpi rigidi: travature reticolari, ...)
- Pb dinamica lineare: regime sinusoidale (circuiti elettrici lineari, oscillazioni forzate di sistemi di corpi rigidi ...)

Es 1

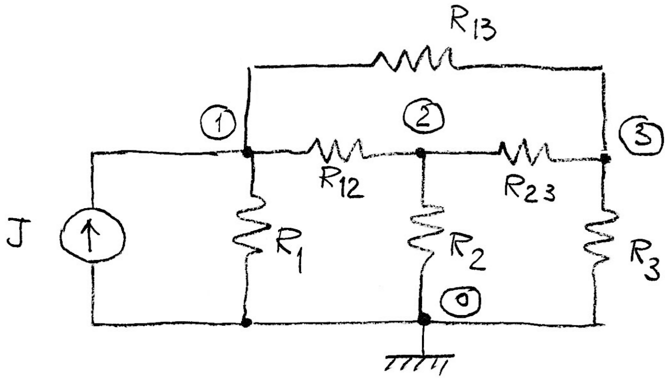


- cerniere ed appoggi lisci
- aste pesanti
- geometria nota

- Incognite : reaz vincolari φ_A (2 comp)
  φ_C (intensita'), φ_B (2 comp), φ_D (intensita')

- Equazioni : cardinali della statica
  (2+1 asta AB, 2+1 asta BD)

Es 2

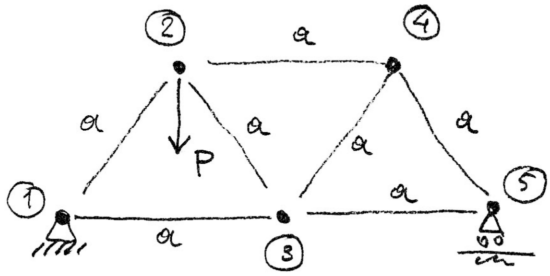


- Rete elettrica lineare
- valori R_K, R_{ij} noti e positivi
- J assegnato

- Incognite: tensioni E_1, E_2, E_3 dei nodi ①, ②, ③ risp al rif ⓪
- Equazioni: LKC ($\Sigma i_k = 0$) e costitutive (legge di Ohm) — tre eq.ni

---

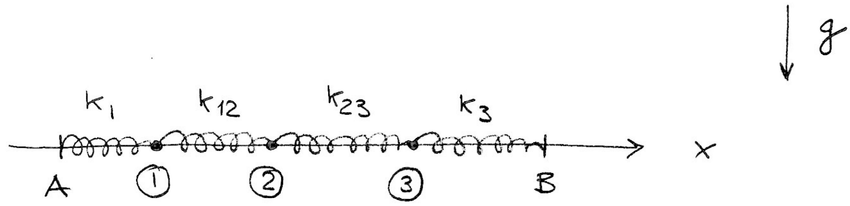
Es 3



- travatura reticolare (aste rigide, giunti lisci)
- geom nota e carico P assegnato

- Incognite: reaz vincolari in ① e ⑤, tens delle aste (3 + 7 inc)
- Equazioni: equilibrio di ciascun nodo (10 eq)

Es 4



– molle lineari (Hooke + lungh riposo nulla)

– x_A, x_B noti

– m_1, m_2, m_3 note, moto unidim.

- Incognite: x_1, x_2, x_3
- Equaz: eq di Newton

(1) $m_1 \ddot{x}_1 = -k_1(x_1 - x_A) - k_{12}(x_1 - x_2)$

(2) $m_2 \ddot{x}_2 = -k_{12}(x_2 - x_1) - k_{23}(x_2 - x_3)$

(3) $m_3 \ddot{x}_3 = -k_{23}(x_3 - x_2) - k_3(x_3 - x_B)$

$x_A = \operatorname{sen} \omega t$

$x_B = 100 + \operatorname{sen} \omega t$

Conf eq: x_1^0, x_2^0, x_3^0

Spostam: $u_1 = x_1 - x_1^0$ ecc

Eq. nello spostam: ...

Metodo FASORIALE: ...