El esquema de Newton-Raphson para problemas unidimensionales

Dr. Alejo O. Sfriso
Universidad de Buenos Aires — materias.fi.uba.ar/6408 — asfriso@fi.uba.ar
SRK Consulting (Argentina) — latam.srk.com — asfriso@srk.com.ar
AOSA — www.aosa.com.ar — asfriso@aosa.com.ar

El esquema de Newton-Raphson

Sea el problema no lineal $p = K[p] \cdot \epsilon_v$

Se parte de un estado convergido $p^n = K[p^n] \cdot \epsilon_v^n$

Se impone un incremento de def. $\epsilon_v^{n+1} = \epsilon_v^n + \Delta\epsilon_v$

El problema es hallar p^{n+1} que cumpla $p^{n+1} = K[p^{n+1}] \cdot \epsilon_v^{n+1}$

Se plantea la forma implícita $F[p] = p - K[p] \cdot \epsilon_v^{n+1} = 0$

Por teorema de Taylor $F[p^{i+1}] = F[p^i] + \left.\frac{\partial F}{\partial p}\right|_i \Delta p = 0$

De donde se despeja $\Delta p = \frac{-F[p^i]}{(\partial F/\partial p)^i}$

Se actualiza $p^{i+1} = p^i + \Delta p$ y se itera hasta convergencia

El esquema de Newton-Rapshon

Cómo funciona

```
Clear[K, F, dF, Kref, m, pref, p0, εv0,
    Δεv, εv, p0, p, i, dp]
K[p_] := Kref * (-p / pref)^m (* Rigidez *)
F[p_] := Re[p - K[p] εv ] (* Función error *)
dF[p_] := Re[1 - εv m K[p] / p ] (* Derivada de F *)
Kref = 2000; m = 0.5; pref = 100; (* Constantes *)
p0 = -10; εv0 = p0 / K[p0]; (* Estado inicial *)
Δεv = -0.1; εv = εv0 + Δεv; (* Incr. deformación *
p = -134; (* Presión de partida *)
(* ---------------------------------- Código *
i = 1;
Print["iter=0 p0=", p];
While[Abs[F[p]] > 0.1 ⋀ i < 100,
 dp = -F[p] / dF[p];
 p = p + dp;
 Print["iter=", i, " p=", p];
 i += 1;
]
```

```
iter=0 p0=-500
iter=1 p=-537.181
iter=2 p=-536.491
```

```
iter=0 p0=-300
iter=1 p=-605.353
iter=2 p=-538.339
iter=3 p=-536.493
```

```
iter=0 p0=-150
iter=1 p=-2607.15
iter=2 p=-764.804
iter=3 p=-551.034
iter=4 p=-536.586
```

```
iter=0 p0=-135
iter=1 p=-41 348.9
iter=2 p=-2497.18
iter=3 p=-753.314
iter=4 p=-549.89
iter=5 p=-536.572
```

```
iter=0 p0=-134
iter=1 p=292 700.
iter=2 p=0.
```

3

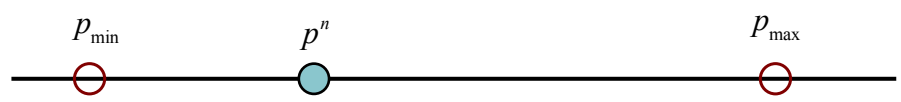
El esquema de Newton-Rapshon

Cómo se arregla (bisección)

srk

- Se eligen topes ridículamente lejanos

```
pmin = -10.^6; pmax = -10.^-6;
(* ---------------------- Código *)
```



4

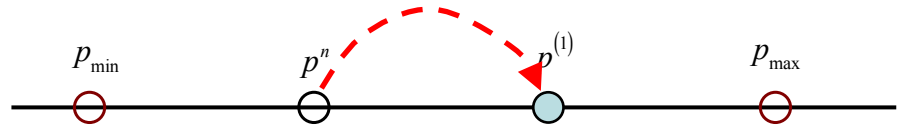
El esquema de Newton-Rapshon

Cómo se arregla (bisección)

srk

- Se eligen topes ridículamente lejanos
- Se resuelve N-R

```
pmin = -10.^6; pmax = -10.^-6;
(* ---------------------- Código *)
i = 1;
While[Abs[F[p]] > 0.1 ∧ i < 100,
    dp = -F[p] / dF[p];
    pNR = p + dp;
```

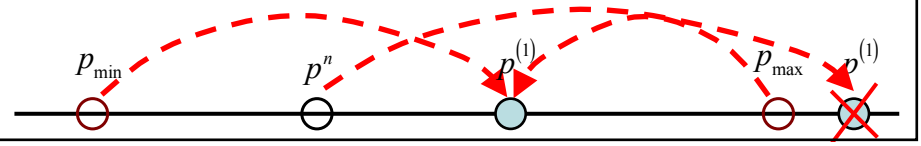


Cómo se arregla (bisección)

srk

- Se eligen topes ridículamente lejanos
- Se resuelve N-R
- Si la presión cae fuera de los topes, se ignora el resultado y se adopta un promedio de los topes

```
pmin = -10.^6; pmax = -10.^-6;
(* ---------------------- Código *)
i = 1;
While[Abs[F[p]] > 0.1 ∧ i < 100,
    dp = -F[p] / dF[p];
    pNR = p + dp;
    If[pNR > pmin ∧ pNR < pmax,
        p = pNR,
        p = -√(pmin pmax);
    ];
```



Cómo se arregla (bisección)

srk

- Se eligen topes ridículamente lejanos
- Se resuelve N-R
- Si la presión cae fuera de los topes, se ignora el resultado y se adopta un promedio de los topes
- En función del valor de *F* se corrigen los topes

```
pmin = -10.^6; pmax = -10.^-6;
(* -------------------- Código *)
i = 1;
While[Abs[F[p]] > 0.1 ∧ i < 100,
    dp = -F[p] / dF[p];
    pNR = p + dp;
    If[pNR > pmin ∧ pNR < pmax,
        p = pNR,
        p = -√(pmin pmax) ;
    ];
    If[F[p] > 0, pmax = p, pmin = p];
    i += 1;
]
```

$p_{\min}$ $p^{(1)} = p_{\max}$

Cómo se arregla (bisección)

srk

El reducido radio de convergencia de NR originó el desarrollo de decenas de métodos de integración

La combinación **N-R con bisección** es **muy robusta**, aunque no es la más eficiente en la mayoría de los casos

Sin bisección no converge aún para un p_0 razonable

```
iter=0  p0=-134
iter=1  p=292 700.
iter=2  p=0.
```

Con bisección converge en pocos pasos aunque p_0 sea extremo

```
iter=0  p0=-10
iter=1  p=-1.
iter=2  p=-1000.
iter=3  p=-577.854
iter=4  p=-537.205
iter=5  p=-536.491
```

```
iter=0  p0=-100 000
iter=1  p=-3801.5
iter=2  p=-879.192
iter=3  p=-563.477
iter=4  p=-536.807
iter=5  p=-536.491
```