Sistemas de Ecuaciones No Lineales

Profesor: Dr. Alejandro S. M. Santa Cruz
JTP: Dr. Juan Ignacio Manassaldi
Aux. 1ra: Ing. Amalia Rueda

Introducción

Podemos decir que los sistemas de ecuaciones no-lineales son un conjunto de ecuaciones no-lineales que deben satisfacerse en simultaneo.

$$
\begin{aligned}
& f_1\left(x_1, x_2, \ldots, x_n\right)=0 \\
& f_2\left(x_1, x_2, \ldots, x_n\right)=0 \\
& \vdots \\
& f_m\left(x_1, x_2, \ldots, x_n\right)=0
\end{aligned}
$$

$$f_1(x_1, x_2, \ldots, x_n) = 0$$
$$f_2(x_1, x_2, \ldots, x_n) = 0$$
$$\vdots$$
$$f_m(x_1, x_2, \ldots, x_n) = 0$$

$x_i \in R \qquad \forall\, i = 1, 2, \ldots, n$

$f_j : R^n \to R \qquad \forall\, j = 1, 2, \ldots, m$

n-m: grados de libertad

Sin perdida de generalidad supondremos $m=n$

$$f_1(x_1, x_2, \ldots, x_n) = 0$$
$$f_2(x_1, x_2, \ldots, x_n) = 0$$
$$\vdots$$
$$f_n(x_1, x_2, \ldots, x_n) = 0$$

Se define la función vectorial $\underline{f}$ asociada al sistema de ecuaciones original y el vector de incógnitas $\underline{x}$:

$$\underline{f}\left(\underline{x}\right)=\begin{bmatrix} f_1\left(x_1,x_2,\ldots,x_n\right)\\ f_2\left(x_1,x_2,\ldots,x_n\right)\\ \vdots \\ f_n\left(x_1,x_2,\ldots,x_n\right)\end{bmatrix} \qquad \underline{x}=\begin{bmatrix} x_1\\ x_2\\ \vdots\\ x_n\end{bmatrix}$$

Por lo tanto, la expresión compacta de un sistema de ecuaciones algebraicas no lineal corresponde a:

$$\underline{f}\left(\underline{x}\right)=\underline{0} \qquad \begin{aligned} &\underline{x}\in R^n \\ &\underline{f}:R^n\to R^n\end{aligned}$$

Aproximaciones sucesivas (multivariable)

El valor de arranque o semilla corresponde a un vector: $\underline{x}^{(0)} = \underline{\alpha}^{(0)} = \begin{bmatrix} \alpha_1^{(0)} \\ \alpha_2^{(0)} \\ \vdots \\ \alpha_n^{(0)} \end{bmatrix}$

La primera aproximación se obtiene a partir de la función vectorial $\underline{F}$ asociada al sistema original.

$$\underline{x}^{(1)} = \underline{F}\left(\underline{x}^{(0)}\right)$$

Finalmente se desarrolla el proceso iterativo hasta satisfacer la tolerancia o alcanzar el máximo de iteraciones:

$$\underline{x}^{(k+1)} = \underline{F}\left(\underline{x}^{(k)}\right)$$

$$k = 1,\ 2,\ \ldots,\ k_{\max}$$

$$\left\| \underline{x}^{(k+1)} - \underline{x}^{(k)} \right\| < \varepsilon$$

$$\frac{\left\| \underline{x}^{(k+1)} - \underline{x}^{(k)} \right\|}{\left\| \underline{x}^{(k)} \right\|} < \varepsilon_r$$

Al tratarse de vectores el error corresponde a la norma de la diferencia entre dos aproximaciones sucesivas

Ejemplo Aprox. Sucesivas

$$\begin{cases} x_2 + x_1^2 - x_1 - 0.75 = 0 \\ x_2 + 5x_2x_1 - x_1^2 = 0 \end{cases}$$

$$\underline{f}(\underline{x}) = \begin{bmatrix} x_2 + x_1^2 - x_1 - 0.75 \\ x_2 + 5x_2x_1 - x_1^2 \end{bmatrix} \qquad \underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \underline{f}(\underline{x}) = \underline{0}$$

Sistema equivalente:

$$\underline{F}(\underline{x}) = \begin{bmatrix} x_2 + x_1^2 - 0.75 \\ -5x_2x_1 + x_1^2 \end{bmatrix} \qquad \underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \underline{x} = \underline{F}(\underline{x})$$

Ejemplo Aprox. Sucesivas

$$\underline{F}(\underline{x}) = \begin{bmatrix} x_2 + x_1^{\,2} - 0.75 \\ -5x_2x_1 + x_1^{\,2} \end{bmatrix} \quad \underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\underline{x}^{(0)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \xrightarrow{\underline{F}\left(\underline{x}^{(0)}\right)} \underline{x}^{(1)} = \begin{bmatrix} 1.2500000 \\ -4.0000000 \end{bmatrix} \xrightarrow{\underline{F}\left(\underline{x}^{(1)}\right)} \underline{x}^{(2)} = \begin{bmatrix} -3.1875000 \\ 26.5625000 \end{bmatrix}$$

$$\underline{e} = \left\| \underline{x}^{(2)} - \underline{x}^{(1)} \right\| = 30.8829696$$

El error de la iteración 5 corresponde a:

$$\left\| \underline{e} \right\| = 9.6510395 \times 10^{15}$$

¡El sistema no converge!

Ejemplo Aprox. Sucesivas

$$\underline{f}(\underline{x}) = \begin{bmatrix} x_2 + x_1^2 - x_1 - 0.75 \\ x_2 + 5x_2x_1 - x_1^2 \end{bmatrix} \rightarrow \underline{F}(\underline{x}) = \begin{bmatrix} \sqrt{-x_2 + x_1 + 0.75} \\ \left(-x_2 + x_1^2\right)/\left(5x_1\right) \end{bmatrix}$$

$$\underline{x}^{(0)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \xrightarrow{\underline{F}\left(\underline{x}^{(0)}\right)} \underline{x}^{(1)} = \begin{bmatrix} 0.8660254 \\ 0 \end{bmatrix} \xrightarrow{\underline{F}\left(\underline{x}^{(1)}\right)} \underline{x}^{(2)} = \begin{bmatrix} 1.2712299 \\ 0.1732051 \end{bmatrix}$$

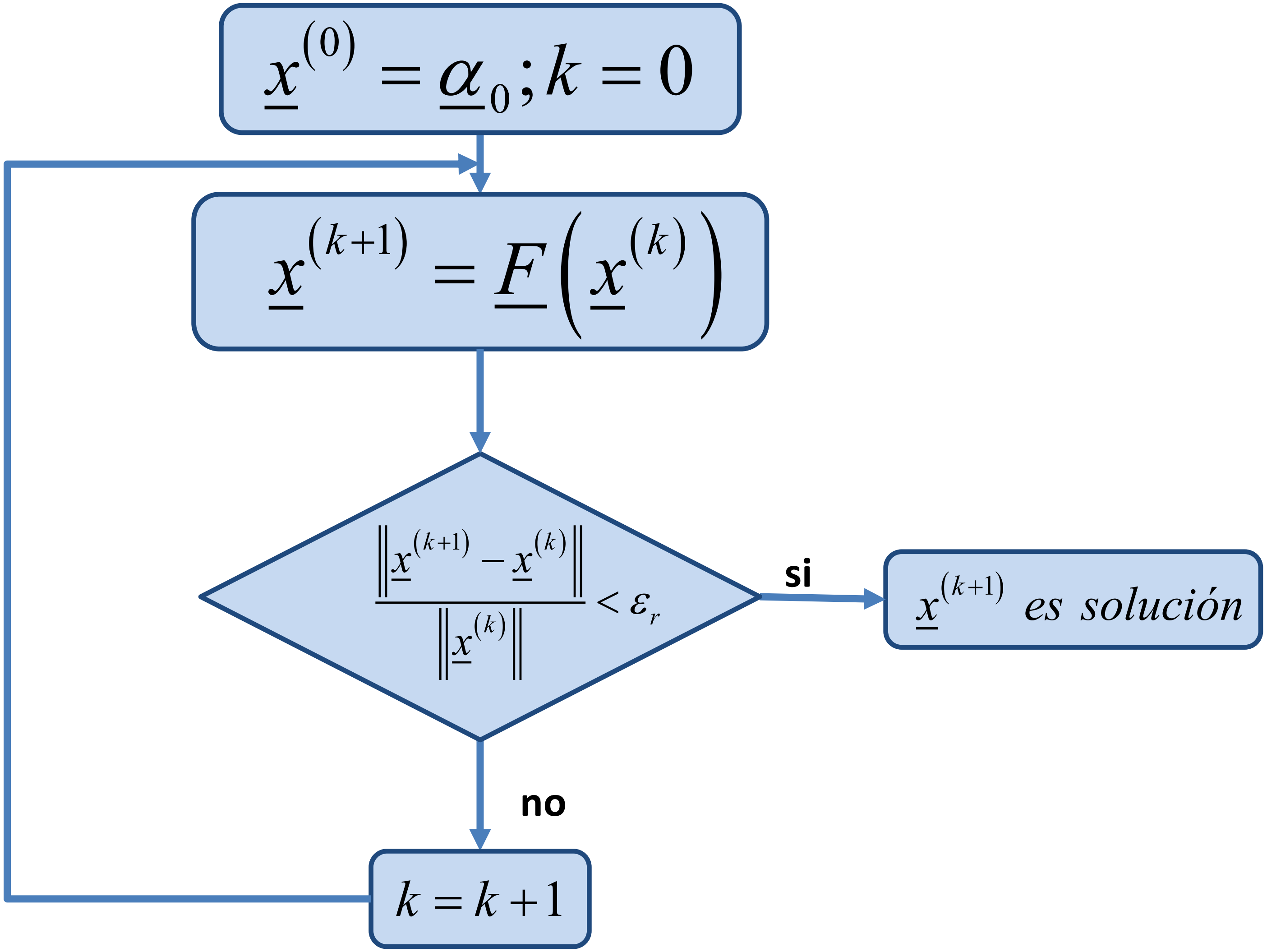
$$\underline{x}^{(9)} = \begin{bmatrix} 1.3720650 \\ 0.2395017 \end{bmatrix} \quad \rightarrow \underline{e} = \left\| \underline{x}^{(9)} - \underline{x}^{(8)} \right\| = 6.0051938 \times 10^{-7}$$

$$F\left(\underline{x}^{(9)}\right) = F\left(\begin{bmatrix} 1.3720650 \\ 0.2395017 \end{bmatrix}\right) = \begin{bmatrix} 1.3720653 \\ 0.2395019 \end{bmatrix}$$

$$f\left(\underline{x}^{(9)}\right) = \begin{bmatrix} -8.4719027 \times 10^{-7} \\ -1.0625308 \times 10^{-6} \end{bmatrix}$$

$$\underline{x}^{*} = \begin{bmatrix} 1.3720650 \\ 0.2395017 \end{bmatrix}$$

Aproximaciones Sucesivas



Aproximaciones Sucesivas (Scilab)

$$\underline{f}(\underline{x}) = \begin{bmatrix} x_2 + x_1^{\,2} - x_1 - 0.75 \\ x_2 + 5x_2x_1 - x_1^{\,2} \end{bmatrix} \rightarrow \underline{F}(\underline{x}) = \begin{bmatrix} \sqrt{-x_2 + x_1 + 0.75} \\ \left(-x_2 + x_1^{\,2}\right)/(5x_1) \end{bmatrix}$$

```
function fx=fsist(x)
   fx = [x(2) + x(1)^2 - x(1)-0.75
        x(2) + 5*x(2)*x(1) - x(1)^2]
endfunction
```

```
function fx=Fsist(x)
   fx = [sqrt(-x(2) + x(1) + 0.75)
        (-x(2) + x(1)^2)/(5*x(1))]
endfunction
```

Aproximaciones Sucesivas (Scilab)

```
function [out,k] =aproxsuc(fun, x0, tol)
x=x0;
for k=1:100
    x(:,k+1)=fun(x(:,k));
    if norm(x(:,k+1)-x(:,k))/norm(x(:,k)) < tol then
        out = x(:,k+1);
        break
    end
end
if k == 100
out=[];
disp('no converge');
end
endfunction
```

Método de Wegstein (multivariable)

Corresponde a una extensión de lo presentado para una variable.

$$\underline{x}^{(0)} = \underline{\alpha}_0$$

$$\underline{x}^{(1)} = \underline{F}\left(\underline{x}^{(0)}\right)$$

$$\underline{\underline{Q}} = \begin{bmatrix} q_1 & 0 & \cdots & 0 \\ 0 & q_2 & \cdots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & q_n \end{bmatrix}$$

$$\omega_i = \frac{F_i\left(\underline{x}^{(k)}\right) - F_i\left(\underline{x}^{(k-1)}\right)}{x_i^{(k)} - x_i^{(k-1)}} \qquad i = 1 \; a \; n$$

$$q_i = \frac{\omega_i}{\omega_i - 1} \quad i = 1 \; a \; n \rightarrow Q_{ii} = q_i \quad i = 1 \; a \; n$$

$$\underline{x}^{(k+1)} = \underline{\underline{Q}}\,\underline{x}^{(k)} + \left(\underline{\underline{I}} - \underline{\underline{Q}}\right)\underline{F}\left(\underline{x}^{(k)}\right)$$

$$k = 1, 2, 3, \ldots, k_{\max}$$

$$\frac{\left\| \underline{x}^{(k+1)} - \underline{x}^{(k)} \right\|}{\left\| \underline{x}^{(k)} \right\|} < \varepsilon_r$$

$$\left\| \underline{x}^{(k+1)} - \underline{F}\left(\underline{x}^{(k+1)}\right) \right\| < \varepsilon$$

Implementación en el ejemplo

$$\underline{F}\left(\underline{x}\right)=\begin{bmatrix} x_2+{x_1}^2-0.75 \\ -5x_2x_1+{x_1}^2 \end{bmatrix} \qquad \underline{x}=\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\underline{x}^{(0)}=\begin{bmatrix} 1 \\ 1 \end{bmatrix} \xrightarrow{\underline{F}\left(\underline{x}^{(0)}\right)} \underline{x}^{(1)}=\begin{bmatrix} 1.2500000 \\ -4.0000000 \end{bmatrix} \xrightarrow{\underline{F}\left(\underline{x}^{(1)}\right)} \begin{bmatrix} -3.1875000 \\ 26.5625000 \end{bmatrix}$$

$$\omega_i=\frac{F_i\left(\underline{x}^{(1)}\right)-F_i\left(\underline{x}^{(0)}\right)}{x_i^{(1)}-x_i^{(0)}} \quad i=1\ a\ 2 \ \rightarrow \underline{\omega}=\begin{bmatrix} -17.75 \\ -6.1125 \end{bmatrix}$$

Implementación en el ejemplo

$$q_i = \frac{\omega_i}{\omega_i - 1} \quad i = 1 \ a \ 2 \quad \rightarrow \underline{\underline{Q}} = \begin{bmatrix} 0.9466667 & 0 \\ 0 & 0.8594025 \end{bmatrix}$$

$$\underline{x}^{(2)} = \underline{\underline{Q}}\,\underline{x}^{(1)} + \left(\underline{\underline{I}} - \underline{\underline{Q}}\right)\underline{F}\left(\underline{x}^{(1)}\right)$$

$$\underline{x}^{(2)} = \begin{bmatrix} 0.9466667 & 0 \\ 0 & 0.8594025 \end{bmatrix}\begin{bmatrix} 1.25 \\ -4 \end{bmatrix} + \begin{bmatrix} 0.0533333 & 0 \\ 0 & 0.1405975 \end{bmatrix}\begin{bmatrix} -3.1875 \\ 26.5625 \end{bmatrix}$$

$$\underline{x}^{(2)} = \begin{bmatrix} 1.0133333 \\ 0.2970123 \end{bmatrix} \rightarrow error = \left\| \underline{x}^{(2)} - \underline{x}^{(1)} \right\| = 4.3035248$$

$$\underline{x}^{(43)} = \begin{bmatrix} 1.3720655 \\ 0.2395020 \end{bmatrix} \rightarrow error = 1.343 \times 10^{-11}$$

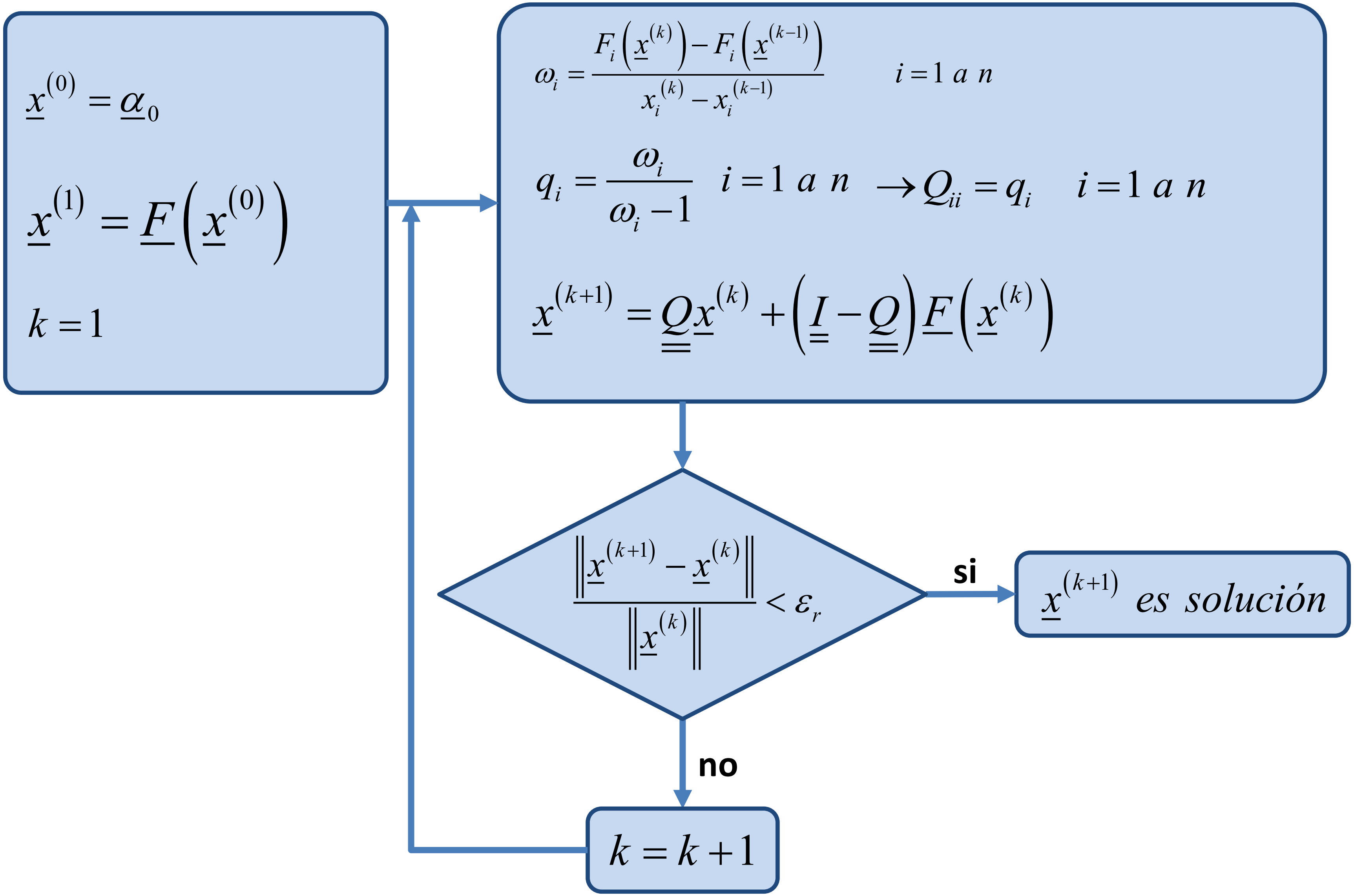
Implementación en el ejemplo

$$F\left(\underline{x}^{(43)}\right) = F\left(\begin{bmatrix} 1.3720655 \\ 0.2395020 \end{bmatrix}\right) = \begin{bmatrix} 1.3720658 \\ 0.2395020 \end{bmatrix}$$

$$f\left(\underline{x}^{(43)}\right) = f\left(\begin{bmatrix} 1.3720655 \\ 0.2395020 \end{bmatrix}\right) = \begin{bmatrix} 2.672907\text{e-}7 \\ -1.921863\text{e-}11 \end{bmatrix}$$

$$\underline{x}^{*} = \begin{bmatrix} 1.3720644 \\ 0.2395041 \end{bmatrix}$$

Método de Wegstein



Método de Wegstein (scilab)

```
function [out,k] =wegstein(fun, x0, tol)
   n=length(x0);
   x(:,1)=x0;
   x(:,2)=fun(x(:,1));
   FF(:,1)=fun(x(:,1));
   FF(:,2)=fun(x(:,2));
   for k=2:100
      w=(FF(:,k)-FF(:,k-1))./(x(:,k)-x(:,k-1));    q=w./(w-1);  Q=diag(q);
      x(:,k+1) = Q*x(:,k)+(eye(n,n)-Q)*FF(:,k)
      if norm(x(:,k+1)-x(:,k))/norm(x(:,k)) < tol then
         out = x(:,k+1);
         break
      end
      FF(:,k+1) = fun(x(:,k+1));
   end
   if k == 100
      out=[];
      disp('no converge');
   end
endfunction
```

Método de Newton

La formula recursiva de Newton corresponde a:

$$\underline{x}^{(k+1)} = \underline{x}^{(k)} - \underline{\underline{J}}^{-1}\left(\underline{x}^{(k)}\right)\underline{f}\left(\underline{x}^{(k)}\right)$$

Inversa de la matriz J evaluada en el punto

$$k = 1, 2, \ldots, k_{\max}$$

$$\frac{\left\|\underline{x}^{(k+1)} - \underline{x}^{(k)}\right\|}{\left\|\underline{x}^{(k)}\right\|} < \varepsilon_r$$

$$\left\|\underline{f}\left(\underline{x}^{(k+1)}\right)\right\| < \varepsilon$$

$$\underline{\underline{J}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

Matriz Jacobiana de la función f

$$\underline{x} \in R^n$$

$$\underline{f} : R^n \rightarrow R^n$$

$$\underline{\underline{J}} : R^n \rightarrow R^{n \times n}$$

Implementación en el ejemplo

$$\underline{f}\left(\underline{x}\right)=\begin{bmatrix} x_2+x_1{}^2-x_1-0.75 \\ x_2+5x_2x_1-x_1{}^2 \end{bmatrix} \qquad \underline{\underline{J}}\left(\underline{x}\right)=\begin{bmatrix} 2x_1-1 & 1 \\ 5x_2-2x_1 & 1+5x_1 \end{bmatrix}$$

$$\underline{x}^{(0)}=\begin{bmatrix}1\\1\end{bmatrix}$$

```
function Jx=Jsist(x)
   Jx = [2*x(1)-1          1
         5*x(2)-2*x(1)    1 + 5*x(1)];
endfunction
```

$$\underline{x}^{(1)}=\begin{bmatrix}1\\1\end{bmatrix}-\begin{bmatrix}2 & -1/3\\ -1 & 1/3\end{bmatrix}\begin{bmatrix}0.25\\5\end{bmatrix}=\begin{bmatrix}2.1666667\\-0.4166667\end{bmatrix}$$

$$\underline{x}^{(2)}=\begin{bmatrix}2.1666667\\-0.4166667\end{bmatrix}-\begin{bmatrix}0.2580254 & -0.0218050\\ 0.1399152 & 0.0726832\end{bmatrix}\begin{bmatrix}1.3611111\\-9.6250000\end{bmatrix}=\begin{bmatrix}1.6055926\\0.0924692\end{bmatrix}$$

$$\underline{e}=\underline{x}^{(2)}-\underline{x}^{(1)}=\begin{bmatrix}1.6055926\\0.0924692\end{bmatrix}-\begin{bmatrix}2.1666667\\-0.4166667\end{bmatrix}=\begin{bmatrix}-0.5610741\\0.5091359\end{bmatrix}\rightarrow\left\|\underline{e}\right\|=0.7576434$$

Implementación en el ejemplo

$$\underline{x}^{(3)} = \begin{bmatrix} 1.4037040 \\ 0.224078 \end{bmatrix} \rightarrow \|\underline{e}\| = 0.2409977$$

$$\underline{x}^{(4)} = \begin{bmatrix} 1.3727742 \\ 0.2392220 \end{bmatrix} \rightarrow \|\underline{e}\| = 0.0344382$$

$$\underline{x}^{(5)} = \begin{bmatrix} 1.3720658 \\ 0.2395018 \end{bmatrix} \rightarrow \|\underline{e}\| = 7.6168800 \times 10^{-4}$$

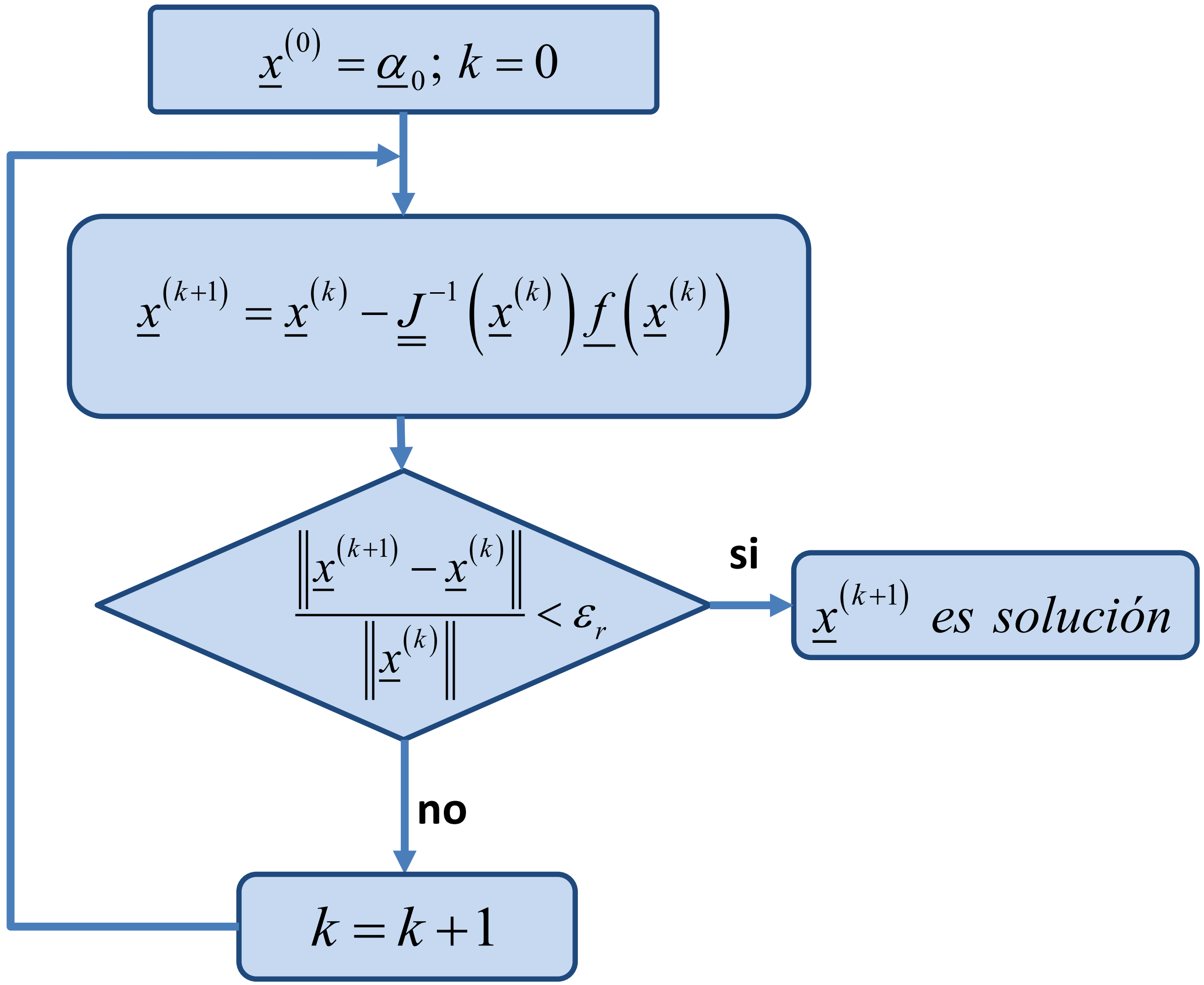
$$\underline{x}^{*} = \begin{bmatrix} 1.3720654 \\ 0.2395019 \end{bmatrix}$$

Podemos afirmar que es solución del sistema.

El error es del orden de 10^{-7}

$$\underline{f}\left(\underline{x}^{*}\right) = \begin{bmatrix} 0.1267875 \times 10^{-12} \\ -0.3406164 \times 10^{-12} \end{bmatrix}$$

Método de Newton



Método de Newton (scilab)

```
function [out,k] =newton(fun, Jx, x0, tol)
  x=x0;
  for k=1:100
    x(:,k+1)=x(:,k) - inv(Jx(x(:,k)))*fun(x(:,k));
    if norm(x(:,k+1)-x(:,k))/norm(x(:,k)) < tol then
      out = x(:,k+1);
      break
    end
  end
  if k == 100
    out=[];
    disp('no converge');
  end
endfunction
```

Método de Newton (II)

La formula recursiva de Newton corresponde a:

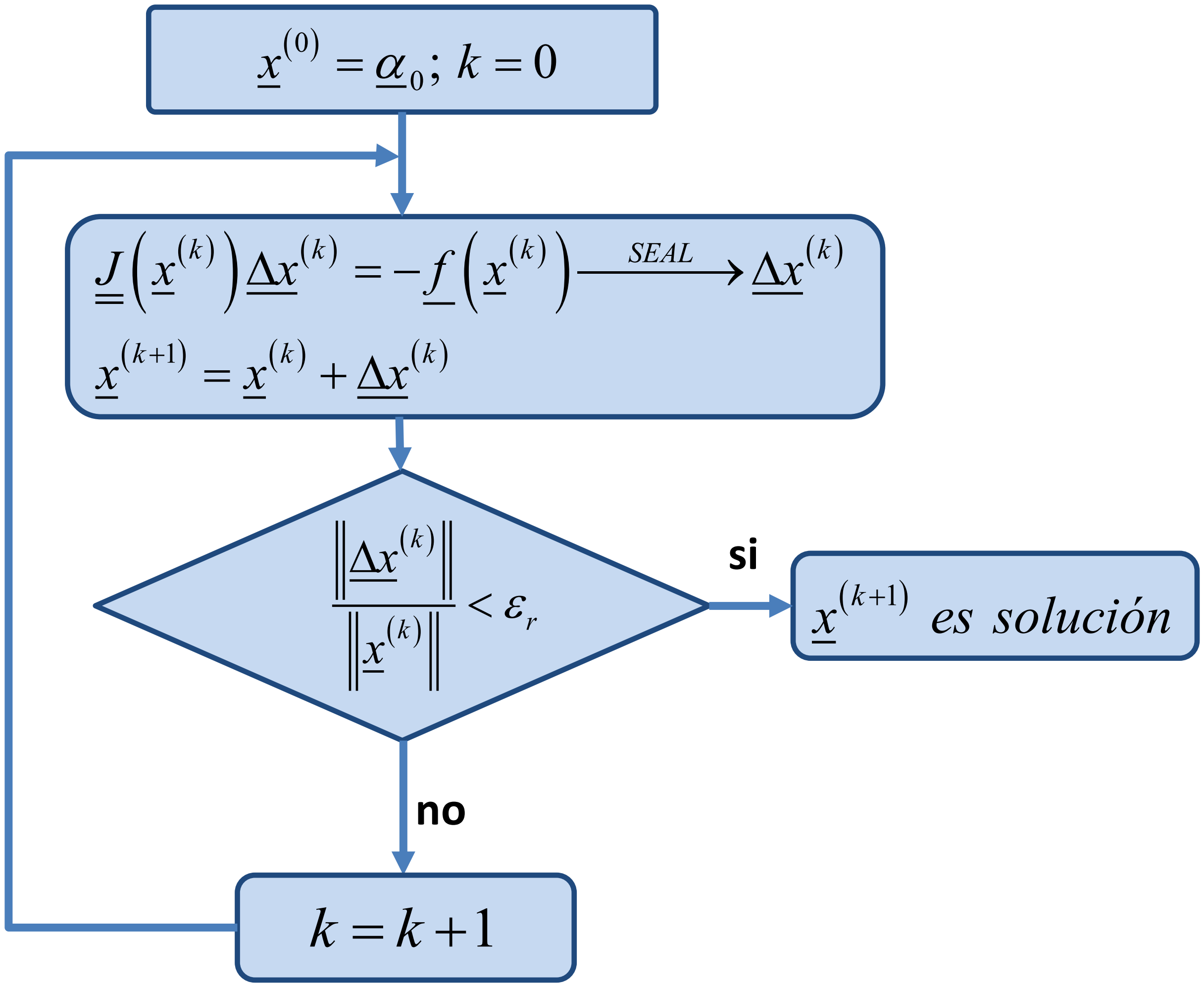
$$\underline{x}^{(k+1)} = \underline{x}^{(k)} - \underline{\underline{J}}^{-1}\left(\underline{x}^{(k)}\right)\underline{f}\left(\underline{x}^{(k)}\right)$$

Es la solución a un SEAL

$$\underline{\underline{J}}\left(\underline{x}^{(k)}\right)\underline{\Delta x}^{(k)} = -\underline{f}\left(\underline{x}^{(k)}\right)$$

$$\underline{x}^{(k+1)} = \underline{x}^{(k)} + \underline{\Delta x}^{(k)}$$

Método de Newton (II)



Método de Newton (II - scilab)

```
function [out,k] =newtonII(fun, Jx, x0, tol)
    x=x0;
    for k=1:100
        dx = Jx(x(:,k))\(-fun(x(:,k)));
        x(:,k+1)=x(:,k) + dx;
        if norm(dx)/norm(x(:,k)) < tol then
            out = x(:,k+1);
            break
        end
    end
    if k == 100
        out=[];
        disp('no converge');
    end
endfunction
```

Método de Newton con factor de relajación

Variación de la solución en la iteración actual (Δx):

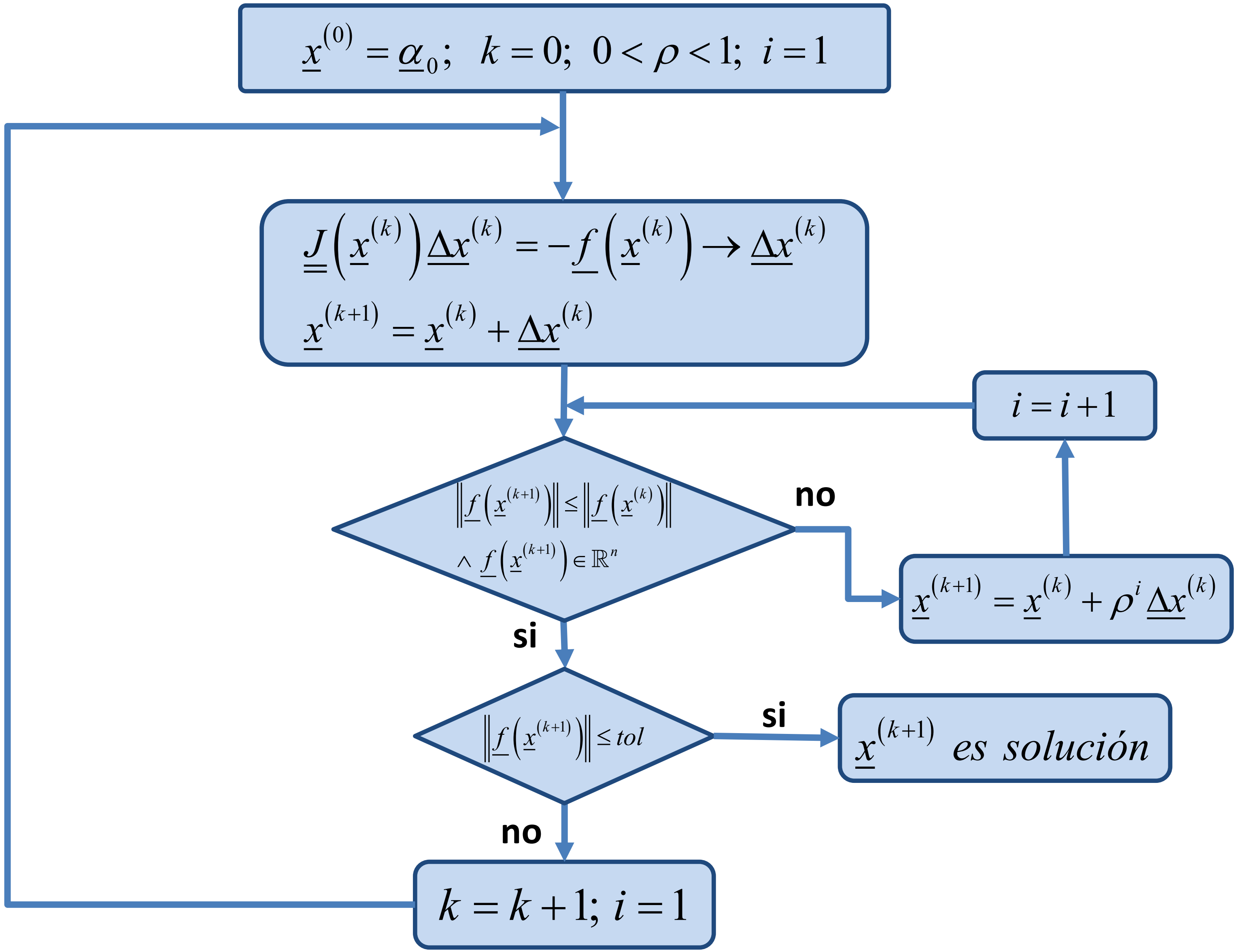
$$\underline{\underline{J}}\left(\underline{x}^{(k)}\right)\underline{\Delta x}^{(k)} = -\underline{f}\left(\underline{x}^{(k)}\right)$$

$$\underline{x}^{(k+1)} = \underline{x}^{(k)} + \rho\underline{\Delta x}^{(k)}$$

Factor de relajación ρ

- Se utiliza para reducir el tamaño del salto de Newton.
- Se lo reduce hasta garantizar una disminución del error. $\left\|\underline{f}\left(\underline{x}^{(k+1)}\right)\right\| \leq \left\|\underline{f}\left(\underline{x}^{(k)}\right)\right\|$
- También se reduce para evitar inconsistencias matemáticas en el nuevo valor $\underline{f}\left(\underline{x}^{(k+1)}\right) \in \mathbb{R}^n$

Método de Newton (II)



Método de Newton (II - Scilab)

```
function [solution, iter]=newtonrelax(fun, Jx, x0, rho0, tol)
  x = x0;  itemax = 100;   errora = 2*tol;   iter = 1;  rho=rho0;
  while errora > tol
    f = fun(x);  error1 = max(abs(f));
    J = Jx(x);
    dx = J\(-f);
    xnew = x + dx;
    f = fun(xnew);   error2 = max(abs(f));
    i=1;   imax=10;
    while (error2 >= error1 || ~isreal(f))
      xnew = x + (rho^i)*dx;
      f = fun(xnew);   error2 = max(abs(f));
      i = i+1;
      if i > imax
        disp(' No convergence: Change relaxation factor or initial guesses')
        solution = xnew;
      end
    end
    x = xnew;
    f = fun(x);  errora = max(abs(f))
    iter = iter + 1;
    if iter > itemax
      disp(' No convergence: Change initial guesses or maximum number of iterations ')
      solution = x;
    end
  end
  solution = x;
endfunction
```

Ejemplo 4 en SciLab

Método de Newton modificado

Requerimos que:

1) El algoritmo progrese hacia la solución en cada paso:

$$\left\| \underline{f}\left(\underline{x}^{(k+1)}\right) \right\| < \left\| \underline{f}\left(\underline{x}^{(k)}\right) \right\|$$

2) Los pasos no sean demasiado grandes:

$$\left\| \underline{x}^{(k+1)} - \underline{x}^{(k)} \right\| < \delta$$

donde δ es elegido por el algoritmo.

Método de Newton Modificado

Supongamos que tenemos:

$$\underline{x}^{(k+1)} = \underline{x}^{(k)} + \underline{p}$$

Corrección

A partir del Método de Newton esperamos que:

$$\underline{p} = -\underline{\underline{J}}^{-1}\left(\underline{x}^{(k)}\right)\underline{f}\left(\underline{x}^{(k)}\right)$$

pero puede que no satisfaga:

$$\left\|\underline{p}\right\| < \delta$$

Además, si ***J*** es singular, $\underline{\boldsymbol{p}}$ no existe.

Método de Newton Modificado

En lugar de evaluar la corrección por el Método de Newton, resolvemos un problema de optimización restringida:

$$\min_{\underline{p}} \; z = \left\| \underline{f}\left(\underline{x}^{(k)}\right) + \underline{\underline{J}}\left(\underline{x}^{(k)}\right).\underline{p} \right\|$$

sujeto a la restricción:

$$\left\| \underline{p} \right\| < \delta$$

Esto funciona aún para ***J*** singular!!!!

$$\boxed{\underline{f}\left(\underline{x}^{(k)}\right) + \underline{\underline{J}}\left(\underline{x}^{(k)}\right).\underline{p}} \longrightarrow \underline{f}\left(\underline{x}^{(k+1)}\right)$$

- Aproximación de primer orden del valor de la función del sistema en el punto k+1.
- Buscamos que el valor de su norma sea mínimo dentro de nuestra región de confianza.

Método de Newton Modificado (IV)

Si dispusiésemos de un ***solver*** que nos permita resolver el problema de optimización, entonces procederíamos de la siguiente manera:

1) Si, $\left\| \underline{f}\left(\underline{x}^{(k)}\right) \right\| < \varepsilon$ entonces parar.
2) Calcular ***p*** vía optimización restringida.
3) Si:
$$\left\| \underline{f}\left(\underline{x}^{(k)} + \underline{p}\right) \right\| < \left\| \underline{f}\left(\underline{x}^{(k)}\right) \right\|$$
aceptar ***p*** e ir a la Etapa (1). Aquí podríamos pensar en aumentar **δ**.
4) Si no, reducir **δ** y volver a la Etapa (2)

Este algoritmo permite determinar la solución de manera confiable. El problema es que pueda quedar atrapado en un mínimo local de la función.

La clave: Arrancar cerca de la solución!!!

Implementación en el ejemplo

$$\underline{f}\left(\underline{x}\right)=\begin{bmatrix} x_2+x_1^{\,2}-x_1-0.75 \\ x_2+5x_2x_1-x_1^{\,2} \end{bmatrix} \quad \underline{\underline{J}}\left(\underline{x}\right)=\begin{bmatrix} 2x_1-1 & 1 \\ 5x_2-2x_1 & 1+5x_1 \end{bmatrix}$$

$$\underline{x}^{(0)}=\begin{bmatrix}1\\1\end{bmatrix} \qquad \delta = 1$$

$$\min_{\underline{p}}\ z=\left\| \underline{\underline{J}}\left(\underline{x}^{(0)}\right).\underline{p}+\underline{f}\left(\underline{x}^{(0)}\right)\right\|$$

$s.t.$

$$\left\|\underline{p}\right\|<\delta$$

$$\min_{\underline{p}}\ z=\left\|\begin{bmatrix}1 & 1\\3 & 6\end{bmatrix}\begin{bmatrix}p_1\\p_2\end{bmatrix}+\begin{bmatrix}0.25\\5\end{bmatrix}\right\|$$

$s.t.$

$$\left\|\underline{p}\right\|<1$$

Implementación en el ejemplo

$$\min_{\underline{p}} \ z = \left\| \begin{bmatrix} 1 & 1 \\ 3 & 6 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} + \begin{bmatrix} 0.25 \\ 5 \end{bmatrix} \right\|$$

$s.a.$

$$\left\| \underline{p} \right\| < 1$$

$$\min_{\underline{p}} \ z = \sqrt{\left| p_1 + p_2 + 0.25 \right|^2 + \left| 3p_1 + 6p_2 + 5 \right|^2}$$

$s.a.$

$$\sqrt{\left| p_1 \right|^2 + \left| p_2 \right|^2} < 1$$

Implementación en el ejemplo

$$\underset{\underline{p}}{mín}\ z = \sqrt{\left|p_1 + p_2 + 0.25\right|^2 + \left|3p_1 + 6p_2 + 5\right|^2}$$

$s.a.$

$$\sqrt{\left|p_1\right|^2 + \left|p_2\right|^2} < 1$$

$$\underline{p} = \begin{bmatrix} 0.287116495751276 \\ -0.957895671703289 \end{bmatrix}$$

$$\underline{x}^{(1)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0.287116495751276 \\ -0.957895671703289 \end{bmatrix} = \begin{bmatrix} 1.287116495751276 \\ 0.042104328296711 \end{bmatrix}$$

Implementación en el ejemplo

$$\left\| \underline{f}\left(\underline{x}^{(0)} + \underline{p}\right) \right\| < \left\| \underline{f}\left(\underline{x}^{(0)}\right) \right\|$$

$$\left\| \begin{bmatrix} -0.338343293819521 \\ -1.343598667872214 \end{bmatrix} \right\| < \left\| \begin{bmatrix} 0.25 \\ 5 \end{bmatrix} \right\|$$

$$1.385544501191005 < 5.006246098625197$$