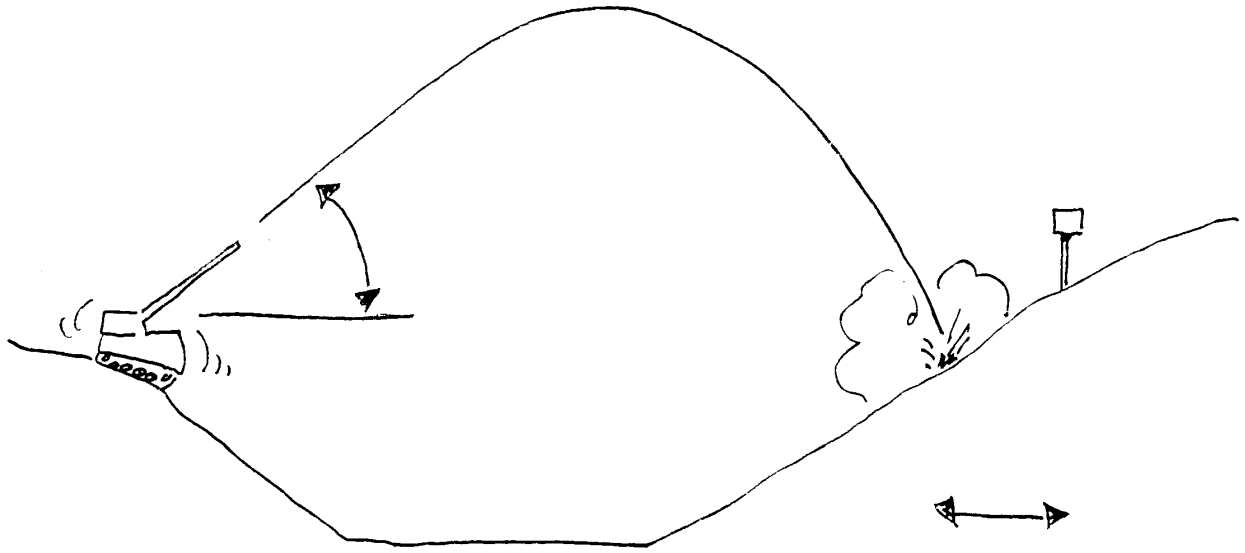
A quoi servent les méthodes numériques ?



Systèmes d'équations non-linéaires

$$\underset{\sim}{f}(\underset{\sim}{x}) = \underset{\sim}{0}$$

VECTEUR VECTEUR

$$\begin{cases} f_1(x_1, x_2, \ldots, x_n) &= 0 \\ f_2(x_1, x_2, \ldots, x_n) &= 0 \\ \qquad \ldots & \\ f_n(x_1, x_2, \ldots, x_n) &= 0 \end{cases}$$

Notation compacte :
les vecteurs sont en gras.

$$\mathbf{f}(\mathbf{x}) = 0$$

Que peut-on faire avec les systèmes ?

Méthodes numériques itératives

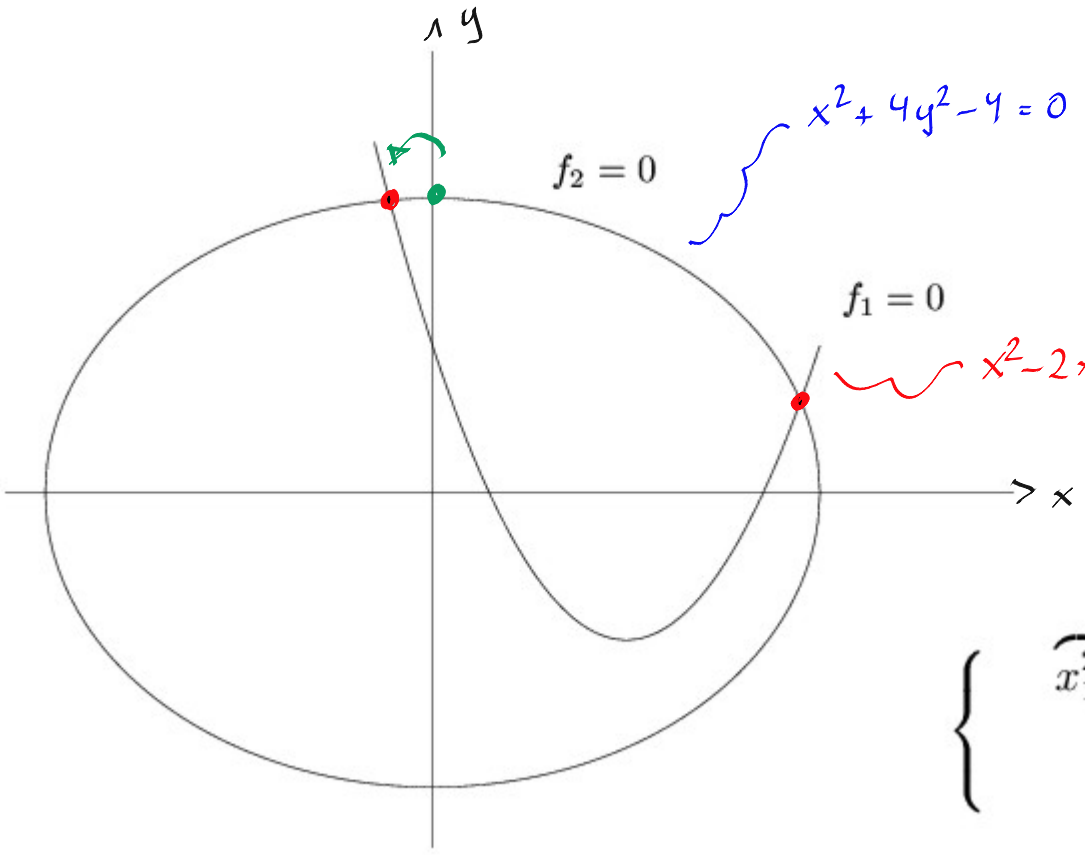
- **Méthode de bissection**
- **Méthodes du point fixe**
- **Méthode de Newton-Raphson**

Robuste, converge toujours si on a un intervalle de départ !

Mais pas généralisable aux systèmes !

Généralisables de manière immédiate aux systèmes..
Ne convergent que sous conditions…
Nécessitent un candidat initial proche de la solution…

Trouver la solution d'un système non-linéaire est très très difficile !



$$\vec{x}_{i+1} = g_i(\vec{x}_i)$$

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} \qquad \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$\begin{cases} \overbrace{x_1^2 - 2x_1 - x_2 + 0.5}^{f_1(x_1, x_2)} = 0 \\ \underbrace{x_1^2 + 4x_2^2 - 4}_{f_2(x_1, x_2)} = 0 \end{cases}$$

Construire une itération

$$\begin{cases} \overbrace{x_1^2 - 2x_1 - x_2 + 0.5}^{f_1(x_1,x_2)} = 0 \\ \underbrace{x_1^2 + 4x_2^2 - 4}_{f_2(x_1,x_2)} = 0 \end{cases}$$

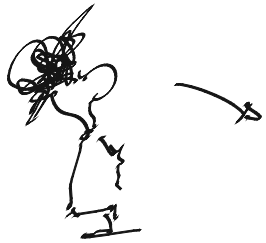
$x_{i+1} = \underbrace{(x_i^2 - y_i + 1/2)/2}_{g_1(x_i, y_i)}$

$x^2 - 2x - y + 1/2 = 0$

$y_{i+1} = \underbrace{(x_i^2 + 4y_i^2 - 4 + 8y)/8}_{g_2(x_i, y_i)}$

$8y + x^2 + 4y^2 - 4 - 8y = 0$

$11y + x^2 + 4y^2 - 4 - 11y = 0$



$y = \underbrace{(x^2 + 4y^2 - 4 + 11y)/11}_{\tilde{g}_2(x,y)}$

Méthode du point fixe

$$\begin{cases} x_{1,i+1} = \dfrac{x_{1,i}^2 - x_{2,i} + 0.5}{2} \\ x_{2,i+1} = \dfrac{-x_{1,i}^2 - 4x_{2,i}^2 + 8x_{2,i} + 4}{8} \end{cases}$$

Itération de Paul

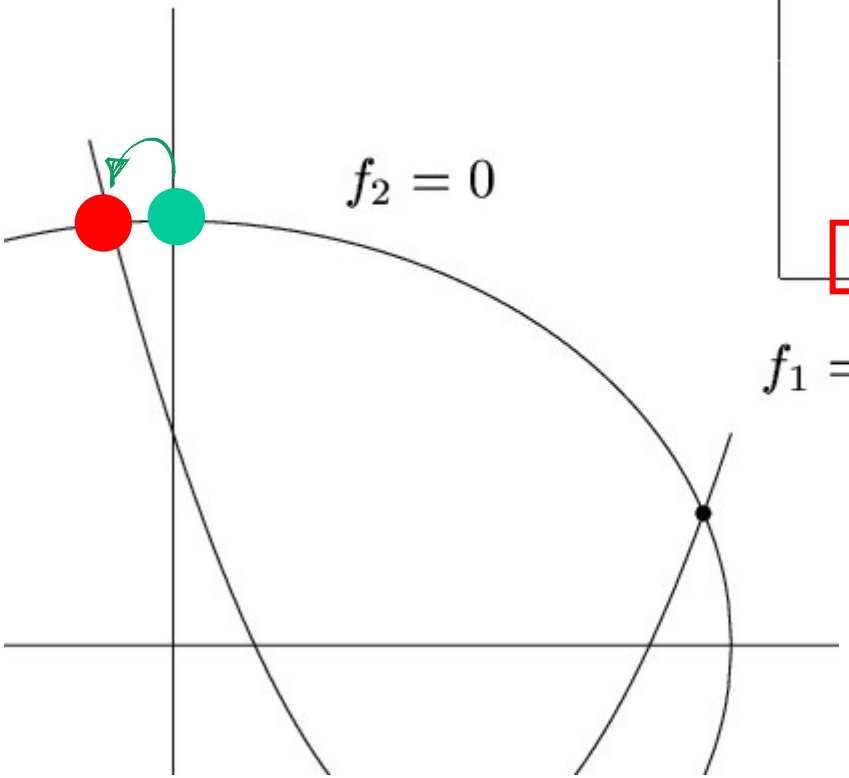
$$\begin{cases} \overbrace{x_1^2 - 2x_1 - x_2 + 0.5}^{f_1(x_1,x_2)} = 0 \\ \underbrace{x_1^2 + 4x_2^2 - 4}_{f_2(x_1,x_2)} = 0 \end{cases}$$

$$\begin{cases} x_{1,i+1} = \dfrac{-x_{1,i}^2 + 4x_{1,i} + x_{2,i} - 0.5}{2} \\ x_{2,i+1} = \dfrac{-x_{1,i}^2 - 4x_{2,i}^2 + 11x_{2,i} + 4}{11} \end{cases}$$

Itération de Pierre

Parfois, cela marche…

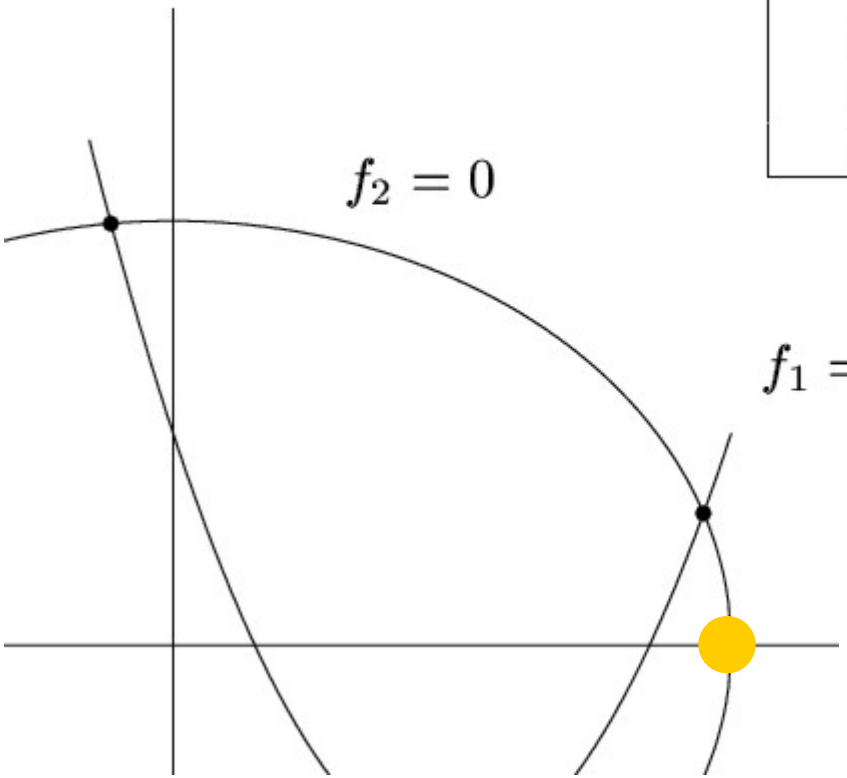
i	$x_{1,i}$	$x_{2,i}$
0	0.0000000	1.0000000
1	- 0.2500000	1.0000000
2	- 0.2187500	0.9921875
3	- 0.2221680	0.9939880
4	- 0.2223147	0.9938121
5	- 0.2221941	0.9938029
6	- 0.2222163	0.9938095
7	- 0.2222147	0.9938083
8	- 0.2222145	0.9938084
9	- 0.2222146	0.9938084



$$\begin{cases} x_{1,i+1} = \dfrac{x_{1,i}^2 - x_{2,i} + 0.5}{2} \\ x_{2,i+1} = \dfrac{-x_{1,i}^2 - 4x_{2,i}^2 + 8x_{2,i} + 4}{8} \end{cases}$$

Parfois, cela diverge..

i	$x_{1,i}$	$x_{2,i}$
0	2.0000000	0.0000000
1	2.2500000	0.0000000
2	2.7812500	- 0.1328125
3	4.1840820	- 0.6085510
4	9.3075467	- 2.4820360
5	44.8062311	- 15.8910907
6	1 011.9947186	- 392.6042650
7	512263.2073904	-205477.8225378



$$\begin{cases} x_{1,i+1} = \dfrac{x_{1,i}^2 - x_{2,i} + 0.5}{2} \\ x_{2,i+1} = \dfrac{-x_{1,i}^2 - 4x_{2,i}^2 + 8x_{2,i} + 4}{8} \end{cases}$$

Méthode du point fixe

On fournit $\mathbf{x}_0$

Tant que $\| \Delta \mathbf{x} \| > \epsilon$, on calcule $\mathbf{x}_{i+1}$ à partir de $\mathbf{x}_i$ avec

$$\mathbf{x}_{i+1} = \mathbf{g}(\mathbf{x}_i)$$

Si on converge, la solution $\mathbf{x}$ est le dernier $\mathbf{x}_{i+1}$ calculé

Notation compacte : les vecteurs sont en gras.

Condition de Lipschitz

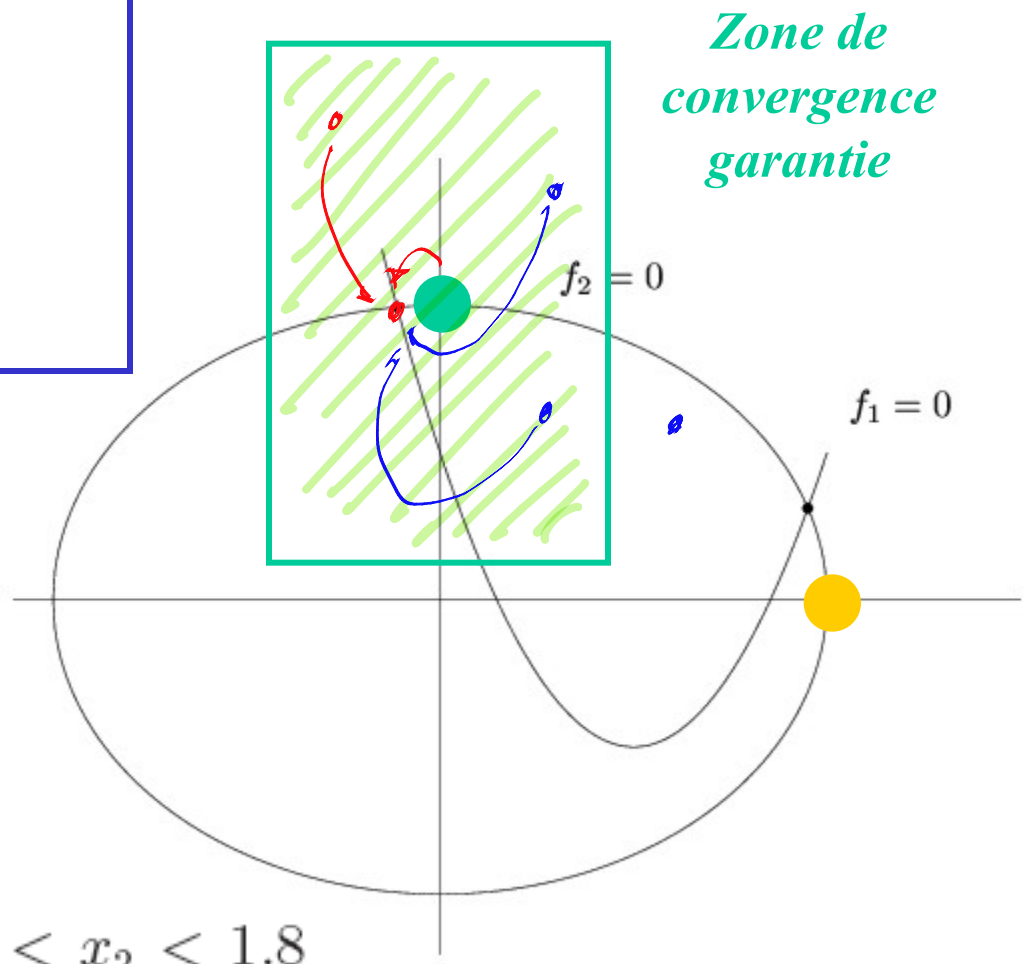
$$\sum_{i=1}^{n} \left| \frac{\partial g_j}{\partial x_i} \right| < 1 \qquad j = 1 \ldots n$$

La méthode du point fixe convergera vers la racine si la condition de Lipschitz est satisfaite !

Et la condition de Lipschitz…

$$|x_1| + |-0.5| < 1$$

$$\left|\frac{-x_1}{4}\right| + |-x_2 + 1| < 1$$

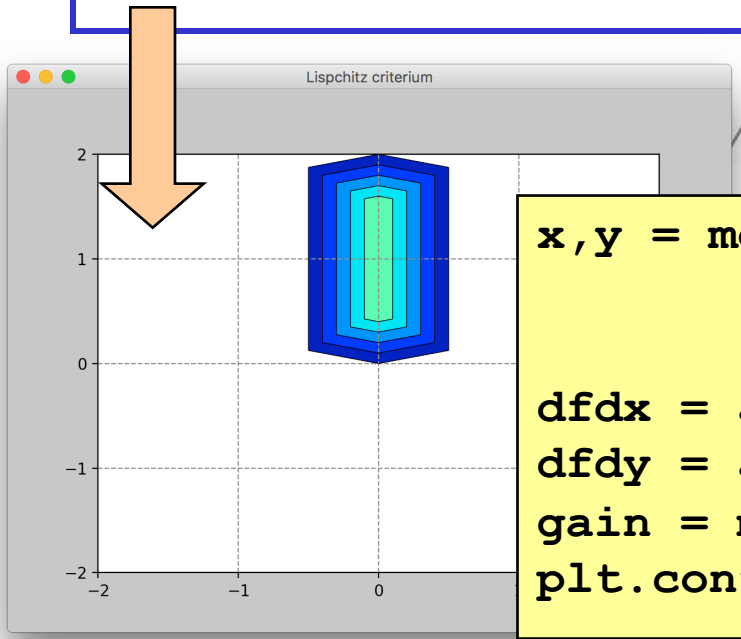


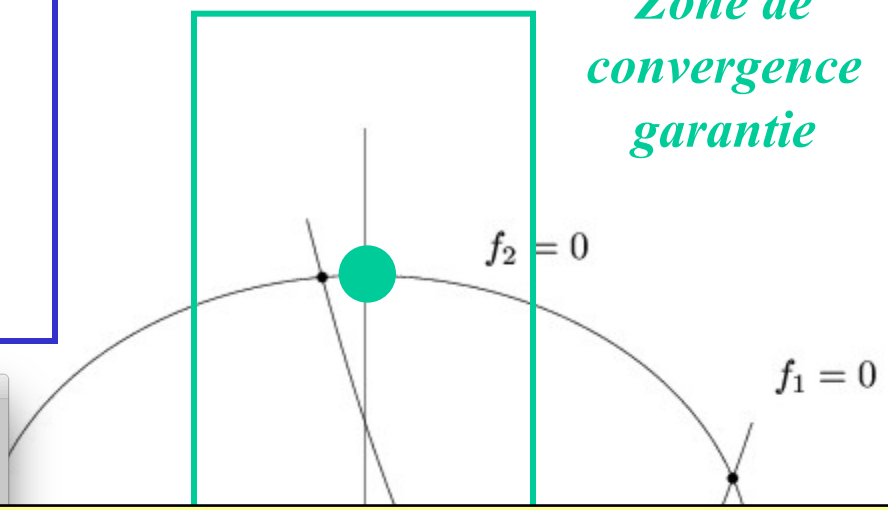
$$-0.5 < x_1 < 0.5 \text{ et } 0.2 < x_2 < 1.8$$

Et la condition de Lipschitz…

$$|x_1| + |-0.5| < 1$$

$$\left|\frac{-x_1}{4}\right| + |-x_2 + 1| < 1$$





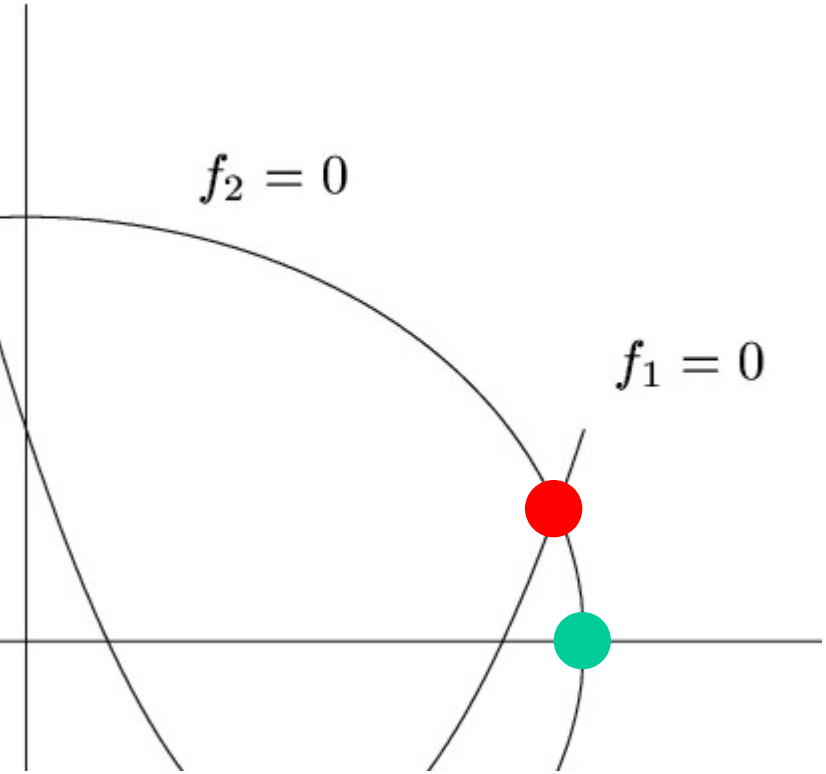
Zone de convergence garantie

```
x,y = meshgrid(linspace(-2,2,1000),
               linspace(-2,2,1000))

dfdx = abs(x)    + 0.5
dfdy = abs(x/4)  + abs(-y+1)
gain = maximum(dfdx,dfdy)
plt.contourf(x,y,gain,arange(0,1.1,0.1)
```

$$\begin{cases} x_{1,i+1} = \dfrac{-x_{1,i}^2 + 4x_{1,i} + x_{2,i} - 0.5}{2} \\ x_{2,i+1} = \dfrac{-x_{1,i}^2 - 4x_{2,i}^2 + 11x_{2,i} + 4}{11} \end{cases}$$

Une autre itération converge...



i	$x_{1,i}$	$x_{2,i}$
0	2.0000000	0.0000000
1	1.7500000	0.0000000
2	1.7187500	0.0852273
3	1.7530629	0.1776676
4	1.8083448	0.2504410
8	1.9035947	0.3160782
12	1.9009241	0.3112267
16	1.9006516	0.3111994
20	1.9006771	0.3112196
24	1.9006768	0.3112185

Est-il possible d'améliorer la vitesse de convergence ?

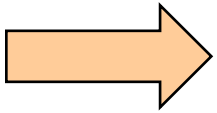
$$\begin{cases} x_{1,i+1} = \dfrac{-x_{1,i}^2 + 4x_{1,i} + x_{2,i} - 0.5}{2} \\ x_{2,i+1} = \dfrac{-x_{1,i}^2 - 4x_{2,i}^2 + 11x_{2,i} + 4}{11} \end{cases}$$

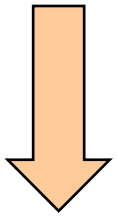
$$\begin{cases} x_{1,i+1} = \dfrac{-x_{1,i}^2 + 4x_{1,i} + x_{2,i} - 0.5}{2} \\ x_{2,i+1} = \dfrac{-x_{1,i+1}^2 - 4x_{2,i}^2 + 11x_{2,i} + 4}{11} \end{cases}$$

Algorithme de Seidel
On utilise la dernière valeur disponible pour chaque inconnue
Parfois, cela converge plus vite,
Parfois, cela converge moins vite, parfois cela diverge…

Peut-on appliquer la méthode du point fixe à un système linéaire ?

$$\underbrace{Ax - b}_{f(x)} = 0$$





Méthodes itératives (pt fixe)

- Parfois plus rapides
- Mais, ne convergent pas toujours
- Moins gourmandes en mémoire
- Utiles pour les très grands systèmes

Méthodes directes

- Elimination gaussienne (technique d'échelonnement du cours d'algèbre)
- Résultat toujours obtenu en un nombre fini d'opérations
- Nombre d'opérations requises = n^3

Un système linéaire

$$\underbrace{\mathbf{A}\mathbf{x} - \mathbf{b}}_{\mathbf{f}(\mathbf{x})} = 0$$

$$\underline{\underline{A}} = \underline{\underline{L}} + \underline{\underline{D}} + \underline{\underline{U}}$$

$$(\underline{\underline{A}} - \underline{\underline{D}})\,\underline{x} + \underline{\underline{D}}\,\underline{x} - \underline{b} = 0$$

JACOBI

$$\underline{x} = \underbrace{(\underline{\underline{D}})^{-1}\left[(\underline{\underline{D}} - \underline{\underline{A}})\,\underline{x} + \underline{b}\right]}_{g(\underline{x})}$$

GAUSS SEIDEL

$$(\underline{\underline{A}} - (\underline{\underline{D}} + \underline{\underline{L}}))\,\underline{x} + (\underline{\underline{D}} + \underline{\underline{L}})\,\underline{x} - \underline{b} = 0$$

$$\underline{x} = \underbrace{(\underline{\underline{D}} + \underline{\underline{L}})^{-1}\left[(\underline{\underline{D}} + \underline{\underline{L}} - \underline{\underline{A}})\,\underline{x} + \underline{b}\right]}_{g(\underline{x})}$$

Méthodes
de Jacobi et de Gauss-Seidel

$$\underbrace{\mathbf{A}\mathbf{x} - \mathbf{b}}_{\mathbf{f}(\mathbf{x})} = 0$$

$$\mathbf{x}_{i+1} = \underbrace{\mathbf{D}^{-1}(\mathbf{D} - \mathbf{A})\mathbf{x}_i + \mathbf{D}^{-1}\mathbf{b}}_{\mathbf{g}_{Jacobi}(\mathbf{x}_i)}$$

Approximation de A^{-1}

$$\mathbf{x}_{i+1} = \underbrace{(\mathbf{D}+\mathbf{L})^{-1}\overbrace{(\mathbf{D}+\mathbf{L}-\mathbf{A})}^{-\mathbf{U}}\,\mathbf{x}_i + (\mathbf{D}+\mathbf{L})^{-1}\,\mathbf{b}}_{\mathbf{g}_{Gauss-Seidel}(\mathbf{x}_i)}$$

Exemple concret

$$\begin{bmatrix} 5 & 3 \\ 4 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 6 \\ 8 \end{bmatrix}$$

On remplace la deuxième équation par « equation(2) – 2 equation(1) » pour obtenir la convergence !

```
def g(x):
  y = copy(x)
  y[0] =  (-3*x[1]+6)/5
  y[1] =  -(6*x[0]-4)/8
  return y

def jacobi(x,tol,nmax):
  n = 0; delta = tol+1;
  x = array(x,dtype=float)
  while (norm(delta) > tol and n < nmax):
    n = n+1; xold = x.copy()
    x = g(x)
    delta = x - xold
  return x

print(jacobi([0,0],10e-6,50))
```

$5x + 3y = 6$

$\hookrightarrow x = (-3y+6)/5$

$4x - 2y = 8$

$-10x - 6y = -12$

$-6x - 8y = -4$

$\hookrightarrow y = -(6x-4)/8$

Et Gauss-Seidel ?

$$\begin{bmatrix} 5 & 3 \\ 4 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 6 \\ 8 \end{bmatrix}$$

L'implémentation de Gauss-Seidel est plus simple que celle de Jacobi !

```
def g(x):
  y = copy(x)
  y[0] =  (-3*x[1]+6)/5
  y[1] =  -(6*
  return y

def jacobi(x,t
  n = 0; delta
  x = array(x,
  while (norm(
    n = n+1; x
    x = g(x)
    delta = x
  return x
```

```
def g(x):

  x[0] =  (-3*x[1]+6)/5
  x[1] =  -(6*x[0]-4)/8
  return x

def gaussseidel(x,tol,nmax):
  n = 0; delta = tol+1;
  x = array(x,dtype=float)
  while (norm(delta) > tol and n < nmax):
    n = n+1; xold = x.copy()
    x = g(x)
    delta = x - xold
return x
```

Jacobi

```
>>>print(jacobi([0,0],10e-6,50))
 Estimated error  1.3000000e+00 at iteration 1
 Estimated error  9.4868330e-01 at iteration 2
 Estimated error  5.8500000e-01 at iteration 3
 Estimated error  4.2690748e-01 at iteration 4
 Estimated error  2.6325000e-01 at iteration 5


 Estimated error  2.9436348e-05 at iteration 28
 Estimated error  1.8151752e-05 at iteration 29
 Estimated error  1.3246357e-05 at iteration 30
 Estimated error  8.1682883e-06 at iteration 31
[ 1.63636089 -0.72726502]
```

Gauss-Seidel

```
>>>print(gaussseidel([0,0],10e-6,50))
 Estimated error  1.2649111e+00 at iteration 1
 Estimated error  3.0000000e-01 at iteration 2
 Estimated error  1.3500000e-01 at iteration 3


 Estimated error  1.0215189e-04 at iteration 12
 Estimated error  4.5968349e-05 at iteration 13
 Estimated error  2.0685757e-05 at iteration 14
 Estimated error  9.3085907e-06 at iteration 15
[ 1.63635754 -0.72726816]
```

Convergence ?

$$\mathbf{x}_{i+1} = \mathbf{M}\mathbf{x}_i + \mathbf{c}$$

$\underline{x}_{i+1} - \underline{x} = \underline{\underline{M}}\,\underline{x}_i + \underline{c} - \underline{\underline{M}}\,\underline{x} - \underline{c}$

$\underline{e}_{i+1} = \underline{\underline{M}}\,\underline{e}_i$

$= \underline{\underline{M}}^i \underline{e}_0$

$\lim_{i \to \infty} \underline{\underline{M}}^i = 0$

$\forall k \quad |\lambda_k| < 1$

RAYON SPECTRAL !

Convergence d'une méthode itérative ?

$$\underbrace{(\mathbf{x}_{i+1} - \mathbf{x})}_{\mathbf{e}_{i+1}} = \mathbf{M}\mathbf{x}_i + \mathbf{c} - \mathbf{M}\mathbf{x} - \mathbf{c}$$

$$= \mathbf{M}\,(\mathbf{x}_i - \mathbf{x})$$

En procédant de la même manière pour chaque étape,

$$= \mathbf{M}^{i+1}\,\underbrace{(\mathbf{x}_0 - \mathbf{x})}_{\mathbf{e}_0}$$

Il faut que…

$$\lim_{i \to \infty} \mathbf{M}^i\,\mathbf{e}^{(0)} = 0$$

Ecrivons l'erreur comme une combili de vecteurs propres…

$$\mathbf{e}_0 = \sum_{j=1}^{n} \alpha_j \mathbf{v}_j$$

↓ Puisque $\mathbf{M}\ \mathbf{v}_i = \lambda_i \mathbf{v}_i$,

$$\mathbf{e}_i = \sum_{j=1}^{n} \alpha_j \lambda_j^i \mathbf{v}_j$$

Pour obtenir…

$$\lim_{i \to \infty} \mathbf{M}^i \mathbf{e}^{(0)} = 0$$

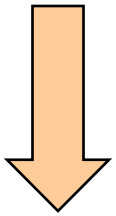
… on doit exiger que le rayon spectral de la matrice M soit inférieur à l'unité

$$|\lambda_i| < 1 \qquad \forall\ i$$

Condition pratique pour Jacobi

$$\sum_{j=1,\neq i}^{n} \left| \frac{a_{ij}}{a_{ii}} \right| < 1 \qquad i = 1, \ldots, n$$

La dominance diagonale permet d'avoir la convergence…



Soit $\mathbf{M}$ *une matrice diagonalisable d'ordre* n*, on a alors :*

$$|\lambda_i| \leq \sum_{j=1}^{n} |m_{ij}| \qquad i = 1, \ldots, n$$

$$|\lambda_i| < 1 \qquad \forall\ i$$

Méthode de Newton-Raphson

On fournit $\mathbf{x}_0$

Tant que $\Delta\mathbf{x} > \epsilon$, on calcule $\mathbf{x}_{i+1}$ à partir de $\mathbf{x}_i$ avec

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_i)\overbrace{(\mathbf{x}_{i+1} - \mathbf{x}_i)}^{\Delta\mathbf{x}} = -\mathbf{f}(\mathbf{x}_i)$$

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta\mathbf{x}$$

Si on converge, la solution $\mathbf{x}$ est le dernier $\mathbf{x}_{i+1}$ calculé

Notation compacte :
les vecteurs sont en gras.

Matrice jacobienne du système

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_i) = \left.\frac{\partial f_j}{\partial x_k}\right|_{(x_{1,i}, x_{2,i}, \ldots, x_{n,i})}$$

Tableau à deux indices
On calcule la dérivée partielle par rapport à la k-ième inconnue de la j-ième composante de f

Tableau à deux indices
Il s'agit de la seconde composante du vecteur Inconnue lors de la i-ème itération

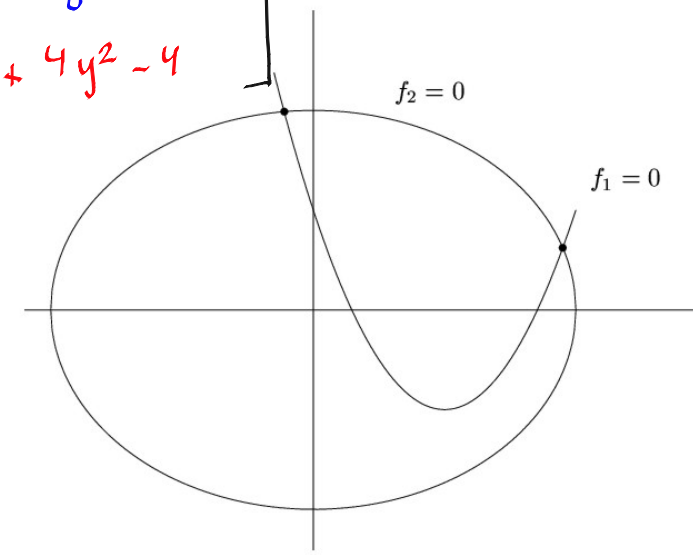
Exemple

$$\begin{cases} \overbrace{x_1^2 - 2x_1 - x_2 + 0.5}^{f_1(x_1,x_2)} &= 0 \\ \underbrace{x_1^2 + 4x_2^2 - 4}_{f_2(x_1,x_2)} &= 0 \end{cases}$$

$$\begin{bmatrix} x^2 - 2x - y + \tfrac{1}{2} \\ x^2 + 4y^2 - 4 \end{bmatrix} = \underbrace{\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}}_{\underset{\sim}{f}(x,y)}$$

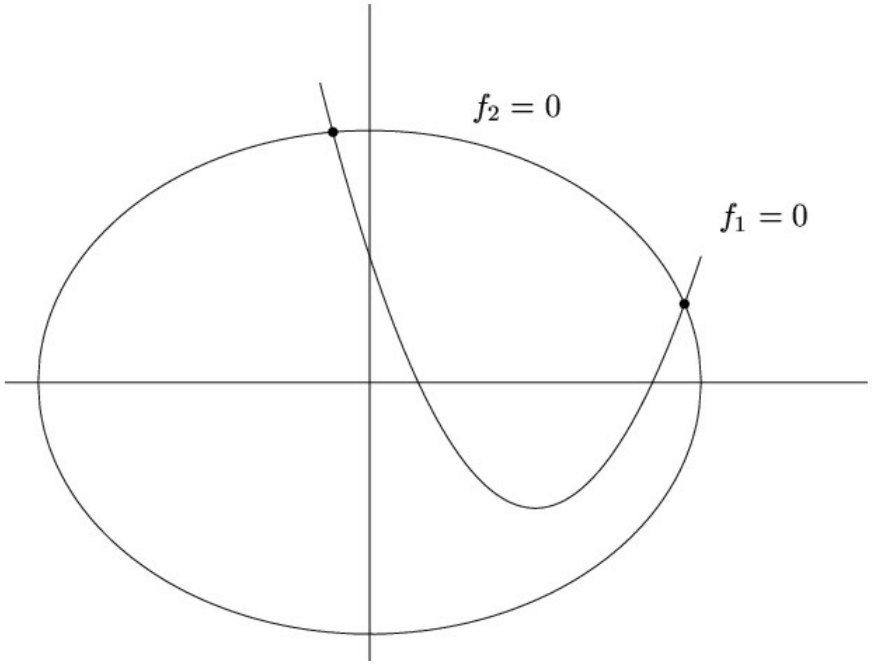
$$\frac{\partial \underset{\sim}{f}}{\partial \underset{\sim}{x}} = \begin{bmatrix} 2x-2 & -1 \\ 2x & 8y \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} 2x-2 & -1 \\ 2x & 8y \end{bmatrix}}_{\frac{\partial \underset{\sim}{f}}{\partial \underset{\sim}{x}}} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = - \overbrace{\begin{bmatrix} x^2 - 2x - y + \tfrac{1}{2} \\ x^2 + 4y^2 - 4 \end{bmatrix}}^{\underset{\sim}{f}}$$



Exemple

$$\begin{cases} \overbrace{x_1^2 - 2x_1 - x_2 + 0.5}^{f_1(x_1,x_2)} &= 0 \\ \underbrace{x_1^2 + 4x_2^2 - 4}_{f_2(x_1,x_2)} &= 0 \end{cases}$$



$$\left.\frac{\partial f_j}{\partial x_k}\right|_{(x_1,x_2)} = \begin{bmatrix} \left.\frac{\partial f_1}{\partial x_1}\right|_{(x_1,x_2)} & \left.\frac{\partial f_1}{\partial x_2}\right|_{(x_1,x_2)} \\ \left.\frac{\partial f_2}{\partial x_1}\right|_{(x_1,x_2)} & \left.\frac{\partial f_2}{\partial x_2}\right|_{(x_1,x_2)} \end{bmatrix}$$

$$= \begin{bmatrix} 2x_1 - 2 & -1 \\ 2x_1 & 8x_2 \end{bmatrix}$$

Comment appliquer la méthode de Newton-Raphson à ce système d'équations non-linéaires ?

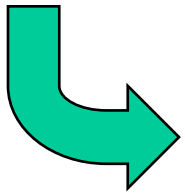
On fournit $\mathbf{x}_0$

Tant que $\Delta\mathbf{x} > \epsilon$, on calcule $\mathbf{x}_{i+1}$ à partir de $\mathbf{x}_i$ avec

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_i)\overbrace{(\mathbf{x}_{i+1} - \mathbf{x}_i)}^{\Delta\mathbf{x}} = -\mathbf{f}(\mathbf{x}_i)$$

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta\mathbf{x}$$

Si on converge, la solution $\mathbf{x}$ est le dernier $\mathbf{x}_{i+1}$ calculé



$$\begin{bmatrix} \left.\frac{\partial f_1}{\partial x_1}\right|_{(x_{1,i},x_{2,i})} & \left.\frac{\partial f_1}{\partial x_2}\right|_{(x_{1,i},x_{2,i})} \\ \left.\frac{\partial f_2}{\partial x_1}\right|_{(x_{1,i},x_{2,i})} & \left.\frac{\partial f_2}{\partial x_2}\right|_{(x_{1,i},x_{2,i})} \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = \begin{bmatrix} -f_1(x_{1,i}, x_{2,i}) \\ -f_2(x_{1,i}, x_{2,i}) \end{bmatrix}$$

Que fournit la méthode de Newton-Raphson pour ce système d'équations non-linéaires ?

$$\begin{cases} \overbrace{x_1^2 - 2x_1 - x_2 + 0.5}^{f_1(x_1,x_2)} &= 0 \\ \underbrace{x_1^2 + 4x_2^2 - 4}_{f_2(x_1,x_2)} &= 0 \end{cases}$$

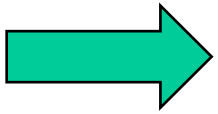
On fournit $\mathbf{x}_0$

Tant que $\Delta\mathbf{x} > \epsilon$, on calcule $\mathbf{x}_{i+1}$ à partir de $\mathbf{x}_i$ avec

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_i)\overbrace{(\mathbf{x}_{i+1} - \mathbf{x}_i)}^{\Delta\mathbf{x}} = -\mathbf{f}(\mathbf{x}_i)$$

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta\mathbf{x}$$

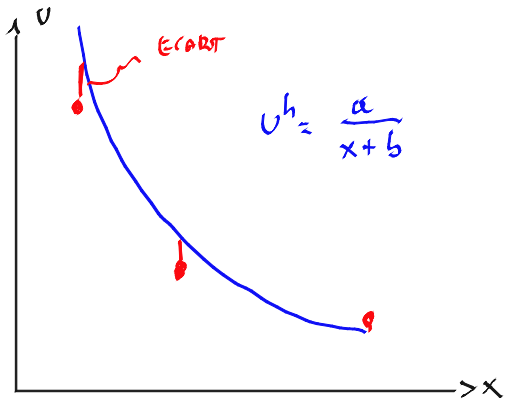
Si on converge, la solution $\mathbf{x}$ est le dernier $\mathbf{x}_{i+1}$ calculé



i	$x_{1,i}$	$x_{2,i}$
0	2.000000	0.250000
1	1.906250	0.312500
2	1.900691	0.311213
4	1.900677	0.311219

Dans le domaine de convergence asymptotique, on double le nombre de chiffres significatifs à chaque itération

Optimisation non-linéaire



n mesures (X_i, U_i)

$$u^h(x) = \frac{a}{x+b}$$

$$f(a,b) = \sum_{i=1}^{n} \left(U_i - \frac{a}{(X_i + b)} \right)^2$$

Fonction à minimiser

On ajuste les deux paramètres *a* et *b* afin de minimiser le carré des écarts

$$f(a,b) = \sum_{i=1}^{n} \left(U_i - \frac{a}{(X_i + b)} \right)^2$$

Fonction à minimiser

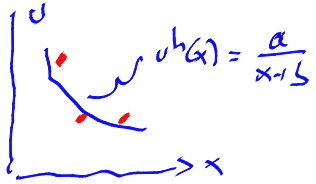
$f(\underline{x}) = 0$

$$\frac{\partial f}{\partial \underline{x}} \Delta \underline{x} = -f$$

$$\underline{x}^{n+1} = \underline{x}^n + \Delta \underline{x}$$

NEWTON RAPHSON

$$\begin{cases} \frac{\partial f}{\partial a}(a,b) = 0 \\ \frac{\partial f}{\partial b}(a,b) = 0 \end{cases}$$



$$f = \begin{bmatrix} \partial f/\partial a \\ \partial f/\partial b \end{bmatrix}$$

$$\underline{x} = \begin{bmatrix} a \\ b \end{bmatrix}$$

$$0 = \frac{\partial f}{\partial a} = 2 \sum \left(U_i - \frac{a}{X_i + b} \right) \left(\frac{-1}{(X_i + b)} \right)$$

$$0 = \frac{\partial f}{\partial b} = 2 \sum \left(\qquad \right) \left(\frac{a}{(X_i + b)^2} \right)$$

$$0 = \frac{\partial f}{\partial a} = 2 \sum \left(U_i - \frac{a}{X_i + b} \right) \left(\frac{-1}{(X_i + b)} \right)$$

$$0 = \frac{\partial f}{\partial b} = 2 \sum \left(\qquad \right) \left(\frac{a}{(X_i + b)^2} \right)$$

$$f(a,b) = \sum_{i=1}^{n} \left(U_i - \frac{a}{(X_i + b)} \right)^2$$

Fonction à minimiser

$$\begin{bmatrix} \frac{\partial^2 f}{\partial a^2} & \frac{\partial^2 f}{\partial a \partial b} \\ \frac{\partial^2 f}{\partial a \partial b} & \frac{\partial^2 f}{\partial b^2} \end{bmatrix} \begin{bmatrix} \Delta a \\ \Delta b \end{bmatrix} = - \begin{bmatrix} \frac{\partial f}{\partial a} \\ \frac{\partial f}{\partial b} \end{bmatrix}$$

$\sum \frac{2}{(\quad)^2}$

$\sum \frac{2U_i}{(\quad)^2} - \sum \frac{4a}{(\quad)^3}$

$\sum \frac{-2U_i}{(\quad)} + \sum \frac{2a}{(\quad)^2}$

$\sum \frac{2U_i}{(\quad)^2} - \sum \frac{4a}{(\quad)^3}$

$\sum \frac{-4aU_i}{(\quad)^3} + \sum \frac{6a^2}{(\quad)^4}$

$\sum \frac{2aU_i}{(\quad)^2} - \sum \frac{2a^2}{(\quad)^3}$

Extréma d'une fonction à deux variables…

$$0 = \frac{\partial f}{\partial a}$$

$$0 = \frac{\partial f}{\partial b}$$

Conditions nécessaires pour obtenir un minimum !

Deux équations non-linéaires

$$0 = \sum_{i=1}^{n} 2\left(U_i - \frac{a}{(X_i+b)}\right)\frac{-1}{(X_i+b)}$$

$$0 = \sum_{i=1}^{n} 2\left(U_i - \frac{a}{(X_i+b)}\right)\frac{a}{(X_i+b)^2}$$

Utilisons Newton-Raphson !

On calcule (a_{k+1}, b_{k+1}) à partir de (a_k, b_k) avec

$$\begin{bmatrix} \frac{\partial^2 f}{\partial a^2}(a_k, b_k) & \frac{\partial^2 f}{\partial a \partial b}(a_k, b_k) \\ \frac{\partial^2 f}{\partial b \partial a}(a_k, b_k) & \frac{\partial^2 f}{\partial b^2}(a_k, b_k) \end{bmatrix} \begin{bmatrix} \Delta a \\ \Delta b \end{bmatrix} = - \begin{bmatrix} \frac{\partial f}{\partial a}(a_k, b_k) \\ \frac{\partial f}{\partial b}(a_k, b_k) \end{bmatrix}$$

$$\begin{aligned} a_{k+1} &= a_k + \Delta a \\ b_{k+1} &= b_k + \Delta b \end{aligned}$$

$$\frac{\partial^2 f}{\partial a^2} = 2 \sum_{i=1}^{n} \frac{1}{(X_i + b)^2}$$

$$\frac{\partial^2 f}{\partial b^2} = -4a \sum_{i=1}^{n} \frac{U_i}{(X_i + b)^3} + 6a^2 \sum_{i=1}^{n} \frac{1}{(X_i + b)^4}$$

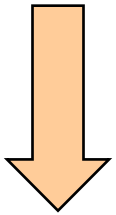
$$\frac{\partial^2 f}{\partial a \partial b} = \frac{\partial^2 f}{\partial b \partial a} = 2 \sum_{i=1}^{n} \frac{U_i}{(X_i + b)^2} - 4a \sum_{i=1}^{n} \frac{1}{(X_i + b)^3}$$

$$\frac{\partial f}{\partial a} = -2 \sum_{i=1}^{n} \frac{U_i}{(X_i + b)} + 2a \sum_{i=1}^{n} \frac{1}{(X_i + b)^2}$$

$$\frac{\partial f}{\partial b} = 2a \sum_{i=1}^{n} \frac{U_i}{(X_i + b)^2} - 2a^2 \sum_{i=1}^{n} \frac{1}{(X_i + b)^3}$$

Et espérer que cela converge…

Il y aura un problème…. si on obtient une valeur de *b* identique à l'opposé des données !

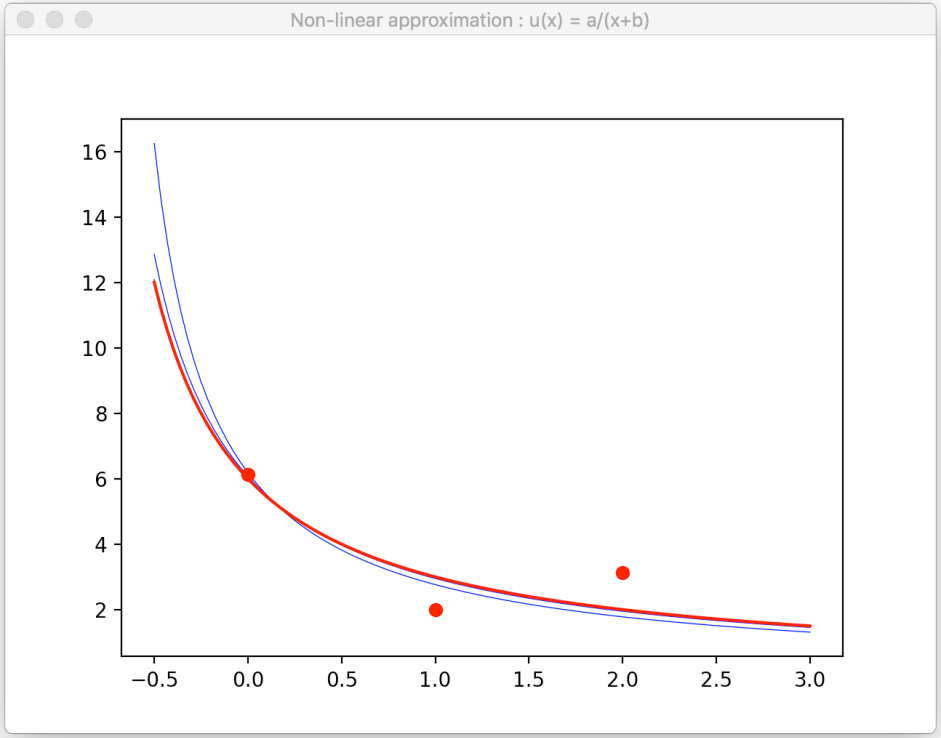


```
Warning: Divide by zero.
ans = Inf
```

$$\sum_{i=1}^{n} 2\left(U_i - \frac{a}{(X_i+b)}\right)\frac{-1}{(X_i+b)}$$

$$\sum_{i=1}^{n} 2\left(U_i - \frac{a}{(X_i+b)}\right)\frac{a}{(X_i+b)^2}$$

En pratique…




```
Iteration 1 :  5.0476842e-01 (a = 4.9952884e+00 b = 8.0757337e-01)
Iteration 2 :  7.6236375e-01 (a = 5.7448233e+00 b = 9.4684249e-01)
Iteration 3 :  2.3523119e-01 (a = 5.9751743e+00 b = 9.9450952e-01)
Iteration 4 :  2.5103378e-02 (a = 5.9996858e+00 b = 9.9992860e-01)
Iteration 5 :  3.2217774e-04 (a = 5.9999999e+00 b = 9.9999999e-01)
Iteration 6 :  5.8254259e-08 (a = 6.0000000e+00 b = 1.0000000e+00)
Iteration 7 :  5.4721560e-16 (a = 6.0000000e+00 b = 1.0000000e+00)
Observed rate of convergence : 2.44
Theoretical rate             : 2.00
```

Systèmes d'équations différentielles non-linéaires

$$\begin{cases} u_1'(x) &= f_1(x, u_1(x), u_2(x), \ldots, u_n(x)) \\ u_2'(x) &= f_2(x, u_1(x), u_2(x), \ldots, u_n(x)) \\ \quad\vdots & \\ u_n'(x) &= f_n(x, u_1(x), u_2(x), \ldots, u_n(x)) \end{cases}$$

$$\begin{cases} u_1(a) &= \overline{u}_1 \\ u_2(a) &= \overline{u}_2 \\ \quad\vdots & \\ u_n(a) &= \overline{u}_n \end{cases}$$

VECTEURS

VECTEUR :-)

Notation compacte : les vecteurs sont en gras.

Trouver $\mathbf{u}(x)$ tel que

$$\begin{cases} \mathbf{u}'(x) &= \mathbf{f}(x, \mathbf{u}(x)), \qquad x \in [a, b] \\ \mathbf{u}(a) &= \overline{\mathbf{u}} \end{cases}$$

C’est exactement la même histoire !

$$U_{ji} = u_j^h(X_i) \approx u_j(X_i)$$

$$\mathbf{U}_i = \mathbf{u}^h(X_i) \approx \mathbf{u}(X_i)$$

Notation compacte : les vecteurs sont en gras.

Méthode d'Euler explicite

$$U_{ji+1} = U_{ji} + h f_j(X_i, U_{1i}, \ldots, U_{ni})$$

Tableau à deux indices

On calcule la j-ème composante du vecteur des inconnues à la i-ème abscisse temporelle

Vecteur

On calcule la j-ème composante du vecteur des inconnues.

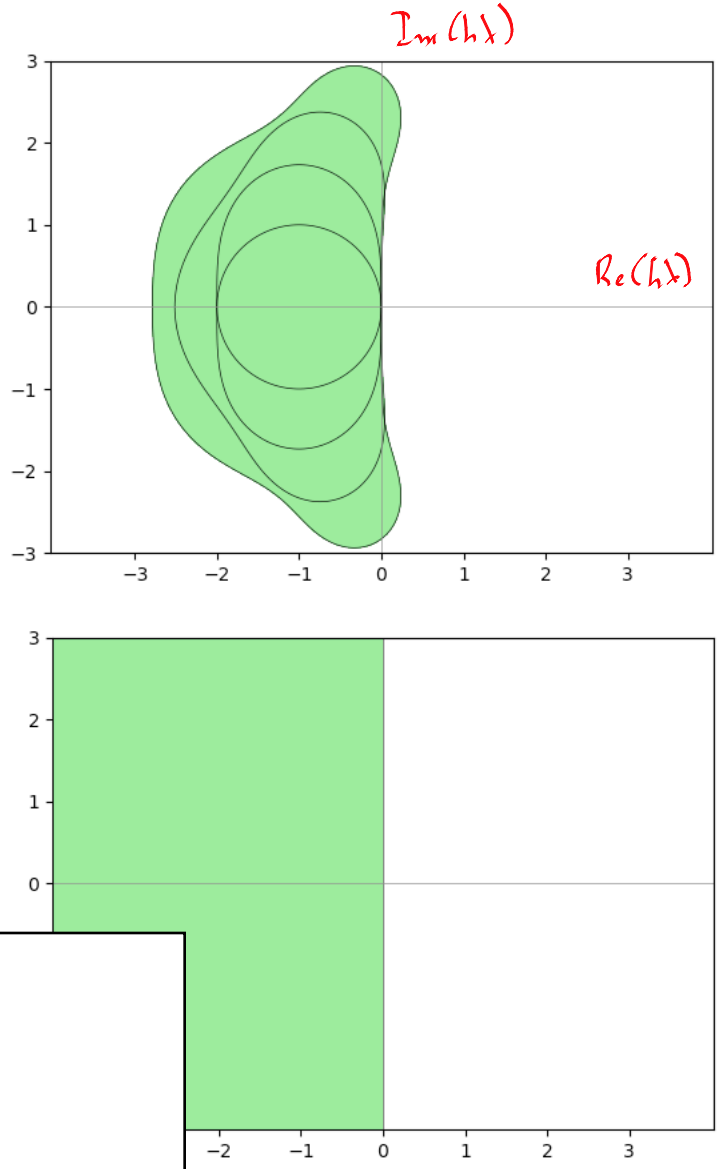
Stabilité des systèmes

$$\mathbf{u}' = \mathbf{A}\mathbf{u}$$

$$\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1}$$

$$\mathbf{v}(x) = \mathbf{P}^{-1}\mathbf{u}(x)$$

$$\mathbf{v}' = \mathbf{D}\mathbf{v}$$



Le problème différentiel initial est équivalent à n équations scalaires

$$v_i'(x) = \lambda_i v_i(x), \qquad i = 1, \ldots, n$$

Exemple

Trouver $(u(x), v(x))$ tels que

$$\begin{cases} u'(x) &= \overbrace{-u^2(x) + v(x)}^{f_1} \\ v'(x) &= \underbrace{-50\ v^2(x) + x}_{f_2}, \qquad x \in [0,4] \\ u(0) &= 1 \\ v(0) &= 1 \end{cases}$$

$$u' = f_1(x, u, v)$$
$$v' = f_2(x, u, v)$$

$$\begin{bmatrix} \frac{\partial f_1}{\partial u} & \frac{\partial f_1}{\partial v} \\ \frac{\partial f_2}{\partial u} & \frac{\partial f_2}{\partial v} \end{bmatrix} = \begin{bmatrix} -2u(x) & 1 \\ 0 & -100\ v(x) \end{bmatrix}$$

$$\begin{bmatrix} \frac{\partial f_1}{\partial v} & \frac{\partial f_1}{\partial r} \\ \frac{\partial f_2}{\partial v} & \frac{\partial f_2}{\partial r} \end{bmatrix} = \begin{bmatrix} -2v(x) & 1 \\ 0 & -100\ r(x) \end{bmatrix}$$

$$\det\left[\underline{\underline{A}} - \lambda \underline{\underline{I}}\right] = 0$$

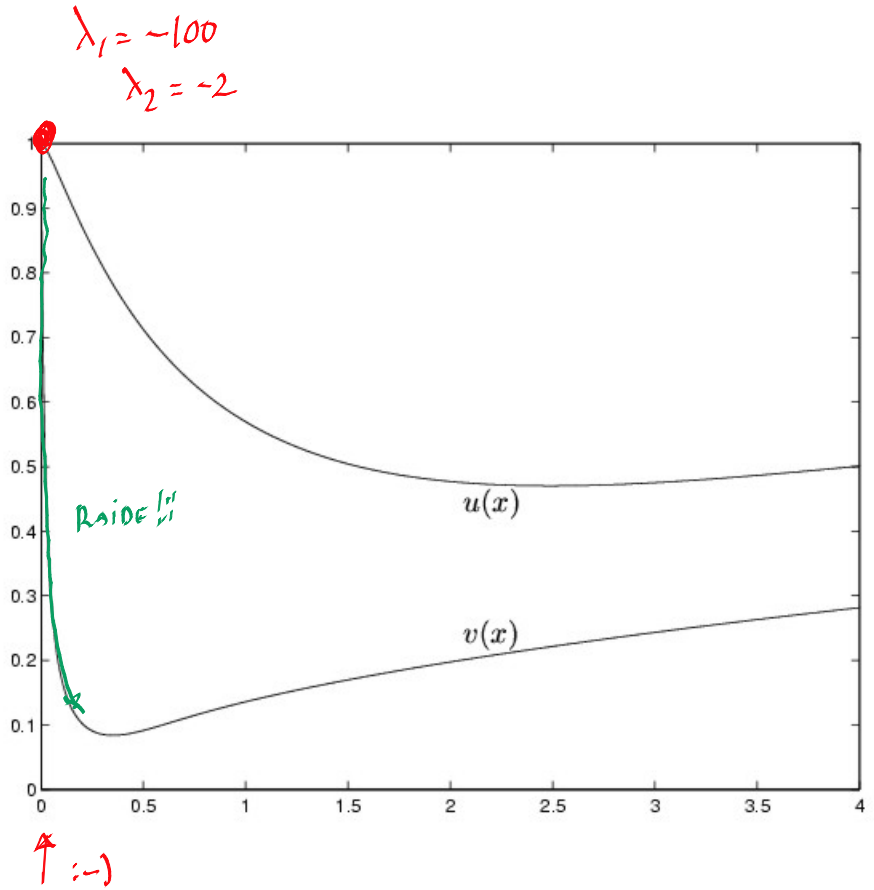
$$\det\begin{bmatrix} -2v-\lambda & 1 \\ 0 & -100r-\lambda \end{bmatrix} = 0$$

$$(-2v-\lambda)(-100r-\lambda) = 0$$

$$\lambda_1(x) = -100\ r(x)$$
$$\lambda_2(x) = -2v(x)$$

$$\begin{bmatrix} \left.\frac{\partial f_1}{\partial u}\right|_x & \left.\frac{\partial f_1}{\partial v}\right|_x \\ \left.\frac{\partial f_2}{\partial u}\right|_x & \left.\frac{\partial f_2}{\partial v}\right|_x \end{bmatrix} = \underline{\underline{A}} = \begin{bmatrix} -2u & 1 \\ 0 & -100v \end{bmatrix}$$

Il s'agit d'un problème raide



$$\det \begin{bmatrix} -2u(x) - \lambda & 1 \\ 0 & -100\ v(x) - \lambda \end{bmatrix} = 0$$

$$\downarrow$$

$$\begin{aligned} \lambda_1(x) &= -100\ v(x) \\ \lambda_2(x) &= -2\ u(x) \end{aligned}$$

Euler explicite

$$\begin{cases} U_{i+1} &= U_i + h\ (-U_i^2 + V_i) \\ V_{i+1} &= V_i + h\ (-50\ V_i^2 + X_i) \end{cases}$$

on utilise U_i V_i

$g_1(U_{i+1}, V_{i+1})$

$$U_{i+1} - \left[U_i + h\left(-U_{i+1}^2 + V_{i+1}\right)\right] = 0$$

$$V_{i+1} - \left[V_i + h\left(-50\,V_{i+1}^2 + X_{i+1}\right)\right] = 0$$

$g_2(U_{i+1}, V_{i+1})$

on utilise U_{i+1} V_{i+1}

$$\begin{cases} U_{i+1} &= U_i + h\ (-U_{i+1}^2 + V_{i+1}) \\ V_{i+1} &= V_i + h\ (-50\ V_{i+1}^2 + X_{i+1}) \end{cases}$$

Euler implicite

Newton-Raphson et Euler implicite

$g(x) = 0$!!

$$\frac{\partial g}{\partial x}\,\Delta x = -f(x)$$

$$\begin{bmatrix} U_{i+1} \\ V_{i+1} \end{bmatrix}$$

$$\begin{cases} \overbrace{U_{i+1} - U_i - h\,\left(-U_{i+1}^2 + V_{i+1}\right)}^{g_1(U_{i+1},\,V_{i+1})} = 0 \\ \underbrace{V_{i+1} - V_i - h\,\left(-50\,V_{i+1}^2 + X_{i+1}\right)}_{g_2(U_{i+1},\,V_{i+1})} = 0 \end{cases}$$

Notation compacte : les vecteurs sont en gras.

$$\mathbf{g}(\mathbf{x}) = 0$$

Comment appliquer la méthode de Newton-Raphson à ce système d'équations non-linéaires ?

On fournit $\mathbf{x}_0$

Tant que $\Delta\mathbf{x} > \epsilon$, on calcule $\mathbf{x}_{i+1}$ à partir de $\mathbf{x}_i$ avec

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_i)\overbrace{(\mathbf{x}_{i+1} - \mathbf{x}_i)}^{\Delta\mathbf{x}} = -\mathbf{f}(\mathbf{x}_i)$$

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta\mathbf{x}$$

Si on converge, la solution $\mathbf{x}$ est le dernier $\mathbf{x}_{i+1}$ calculé

JACOBIENNE DU SYSTEME D'EDO :-)

$$\begin{bmatrix} \left.\frac{\partial g_1}{\partial x_1}\right|_{(x_1,x_2)} & \left.\frac{\partial g_1}{\partial x_2}\right|_{(x_1,x_2)} \\ \left.\frac{\partial g_2}{\partial x_1}\right|_{(x_1,x_2)} & \left.\frac{\partial g_2}{\partial x_2}\right|_{(x_1,x_2)} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - h \begin{bmatrix} -2x_1 & 1 \\ 0 & -100x_2 \end{bmatrix}$$

Convergence de Newton-Raphson ?

$$\begin{bmatrix} \left.\frac{\partial g_1}{\partial x_1}\right|_{(x_1,x_2)} & \left.\frac{\partial g_1}{\partial x_2}\right|_{(x_1,x_2)} \\ \left.\frac{\partial g_2}{\partial x_1}\right|_{(x_1,x_2)} & \left.\frac{\partial g_2}{\partial x_2}\right|_{(x_1,x_2)} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - h \begin{bmatrix} -2x_1 & 1 \\ 0 & -100x_2 \end{bmatrix}$$

CECI EST LE CRITERE DE STABILITE D'EULER EXPLICITE !

Schémas imbriqués

Euler implicite

On fournit $\mathbf{U}_0$

Pour chaque i, on calcule $\mathbf{x} = \mathbf{U}_{i+1}$ à partir de $\mathbf{U}_i$ en résolvant le problème

$$\underbrace{\mathbf{U}_{i+1} - \mathbf{U}_i - hf(\mathbf{U}_{i+1}, X_{i+1})}_{\mathbf{g}(\mathbf{x})} = 0$$

par la méthode de Newton-Raphson en partant de $\mathbf{U}_i$

Newton-Raphson

On fournit $\mathbf{x}_0 = \mathbf{U}_i$

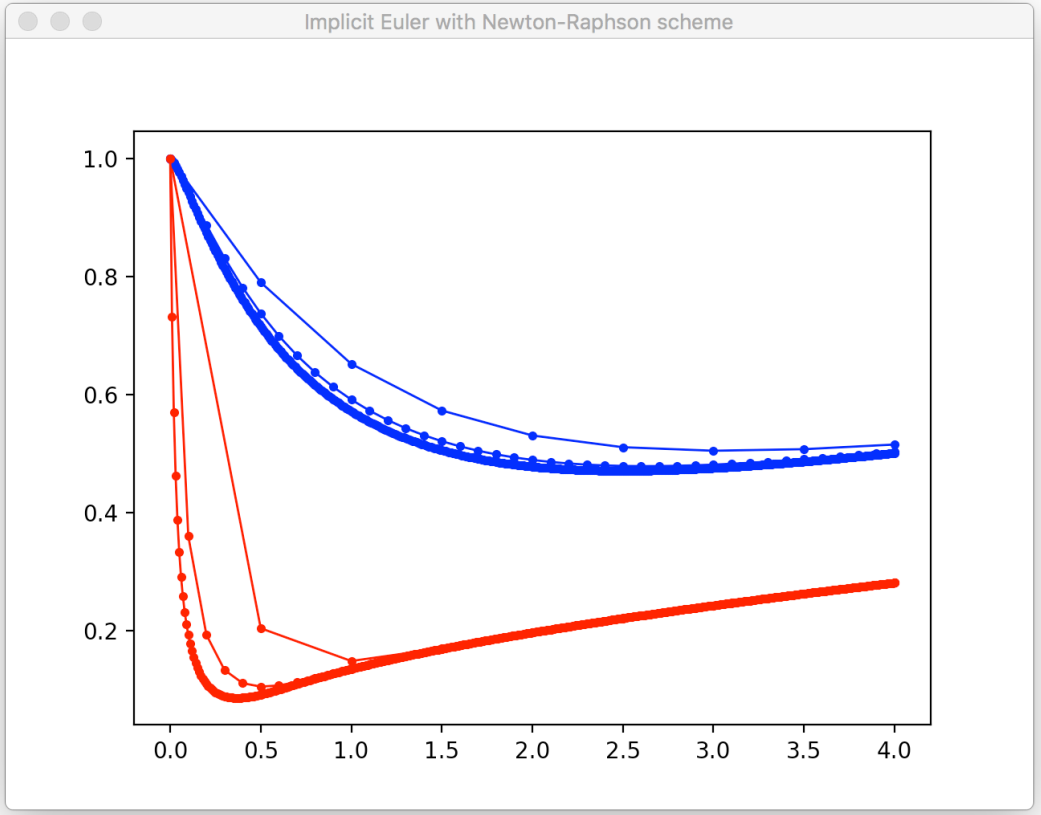
Tant que $\Delta\mathbf{x} > \epsilon$, on calcule $\mathbf{x}_{j+1}$ à partir de $\mathbf{x}_j$ avec

$$\frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\mathbf{x}_j) \overbrace{(\mathbf{x}_{j+1} - \mathbf{x}_j)}^{\Delta\mathbf{x}} = -\mathbf{g}(\mathbf{x}_j)$$

$$\mathbf{x}_{j+1} = \mathbf{x}_j + \Delta\mathbf{x}$$

Si on converge, la solution $\mathbf{U}_{i+1}$ est le dernier $\mathbf{x}_{j+1}$ calculé

Utilisation conjointe des méthodes d'Euler implicite et de Newton-Raphson

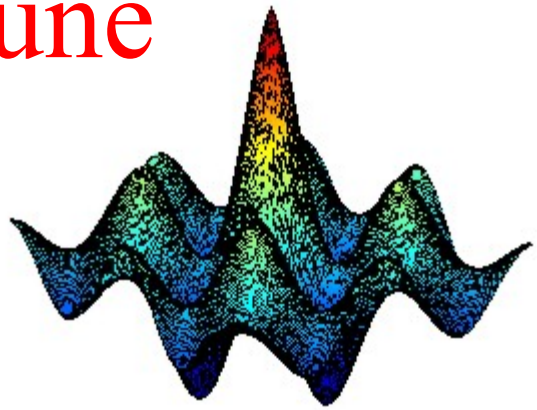


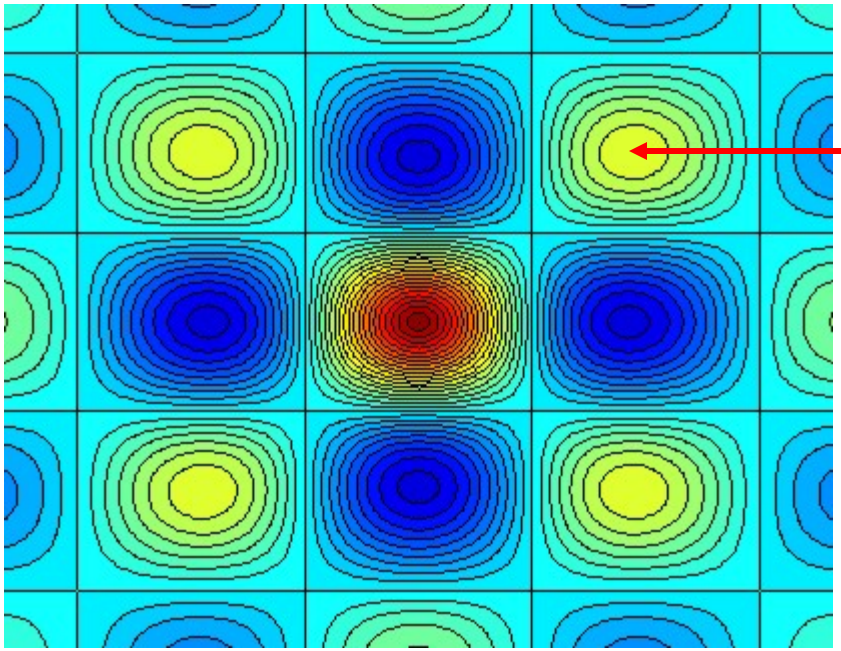
Minimiser une fonction non-linéaire est très très difficile !

Les méthodes numériques… existantes

Des propos dignes de l'un de ces fous énigmatiques qui hantent les forêts enchantées dans les pièces de Shakespeare

Trouver le maximum d'une fonction non-linéaire est une tâche très très difficile…





Un maximum local ?

Comment savoir la présence ou l'absence d'une fosse plus profonde sur base d'informations locales….

Impossible ?

Minimiser une fonction non-linéaire est très très difficile !

Le message est que rien n'est su de manière sûre. Il y a des choses dont nous savons que nous les savons. Il y a des inconnues connues, c'est-à-dire des choses dont nous savons maintenant que nous ne les savons pas.

Mais, il y a aussi des inconnues inconnues. Il y a des choses dont nous savons que nous ne les savons pas …. Et chaque année, nous découvrons un peu plus de ces inconnues inconnues.

… Il y a une autre façon de dire cela, c'est que l'inexistence de preuves n'est pas une preuve d'inexistence

Secrétaire à la Défense US
à l'OTAN à Bruxelles, juin 2002 !

(Lewis Lapham)

Les méthodes numériques… existantes

Des propos dignes de l'un de ces fous énigmatiques qui hantent les forêts enchantées dans les pièces de Shakespeare