Optimisation

Optimisation sans dérivées:

De Nelder-Mead aux méthodess globales

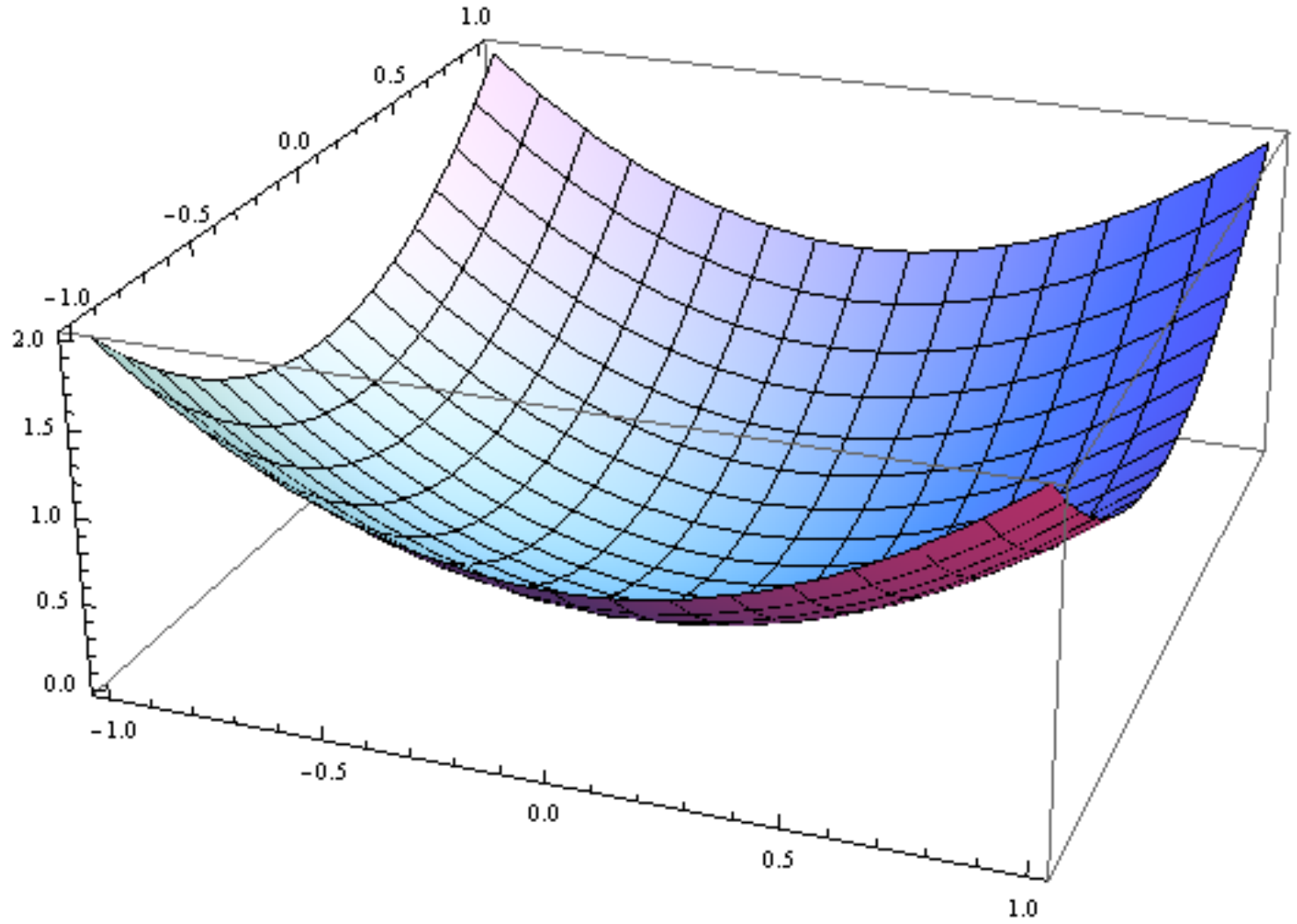
Définition

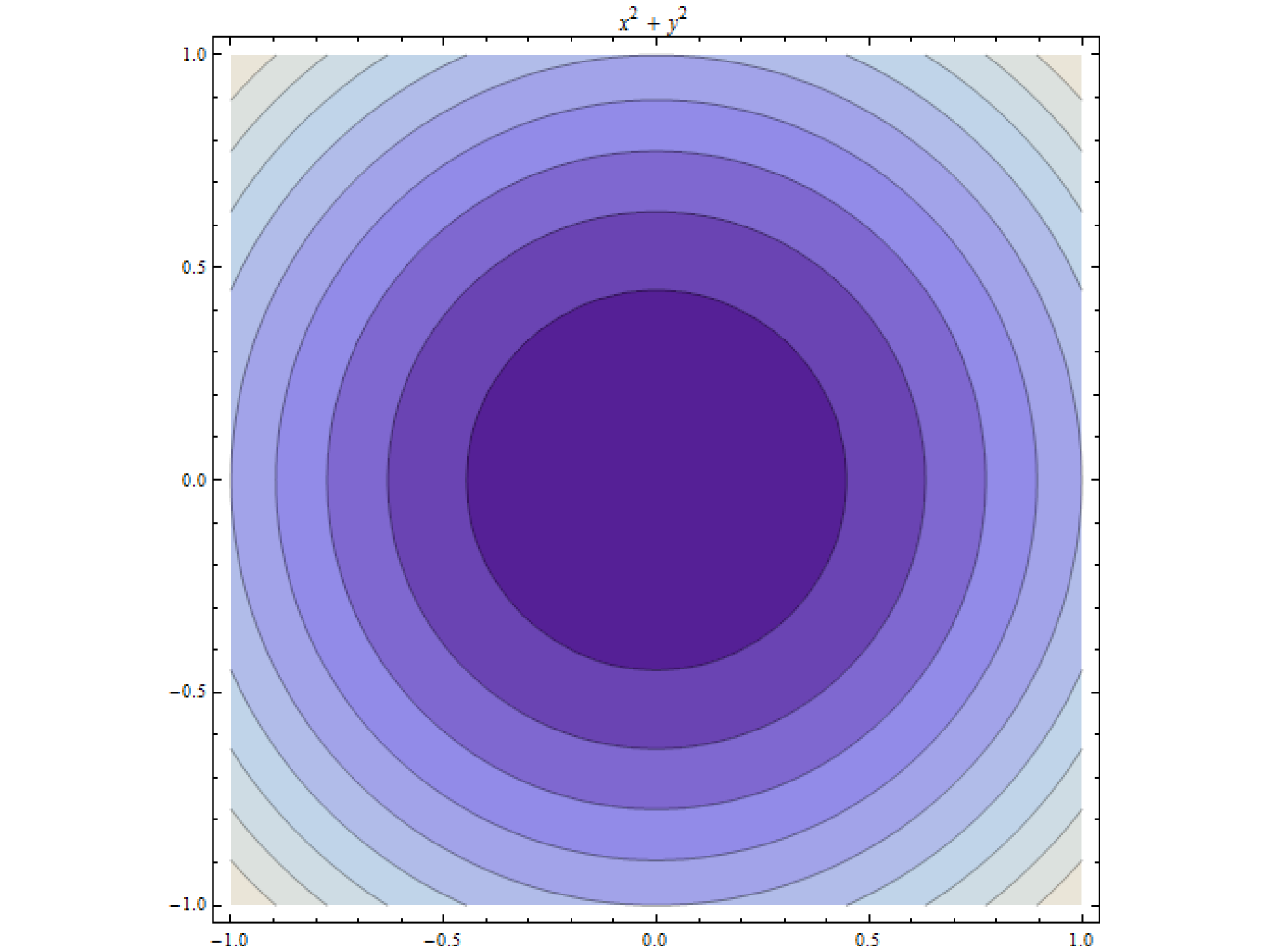
- L’optimisation d’un problème ou d’une fonction consiste à chercher la valeur des variables de la fonction qui minimise (ou qui maximise) sa valeur.
- L’optimisation est généralement un problème compliqué, et il existe de multiples techniques, dont l’utilisation doit être adaptée au type du problème.

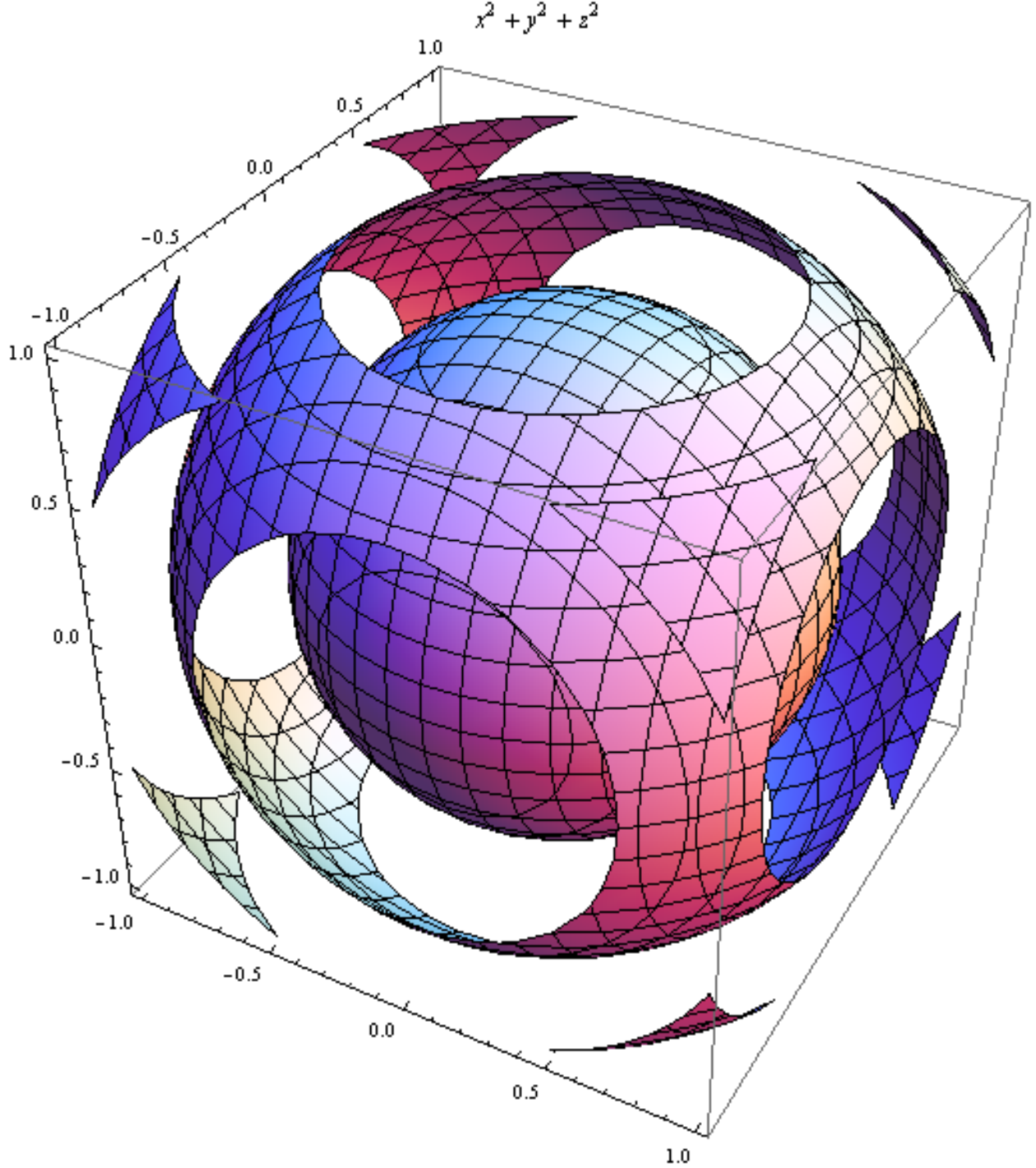
Caractérisation des techniques d'optimisation

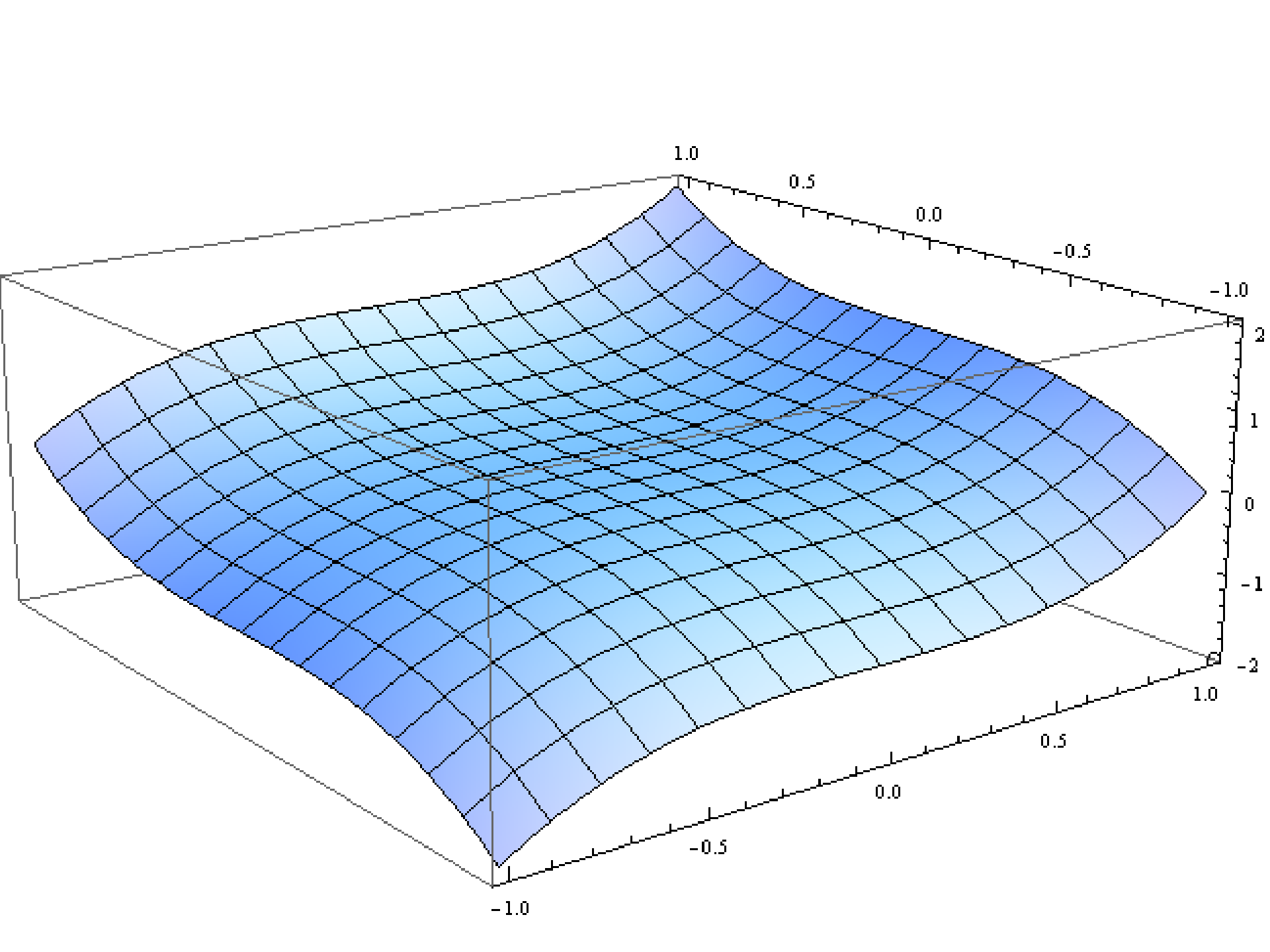
- Optimisation globale/locale
  - L'optimisation globale consiste à chercher le maximum de la fonction sur l'ensemble de définition
  - L'optimisation locale consiste à chercher le maximum de la fonction au voisinage d'un point.
- Méthode déterministe/stochastique
  - Une méthode stochastique parcourt l'espace de recherche de façon « aléatoire ». Deux exécutions successives peuvent donner deux résultats différents.
  - Une méthode déterministe parcourt toujours l'espace de recherche de la même façon.

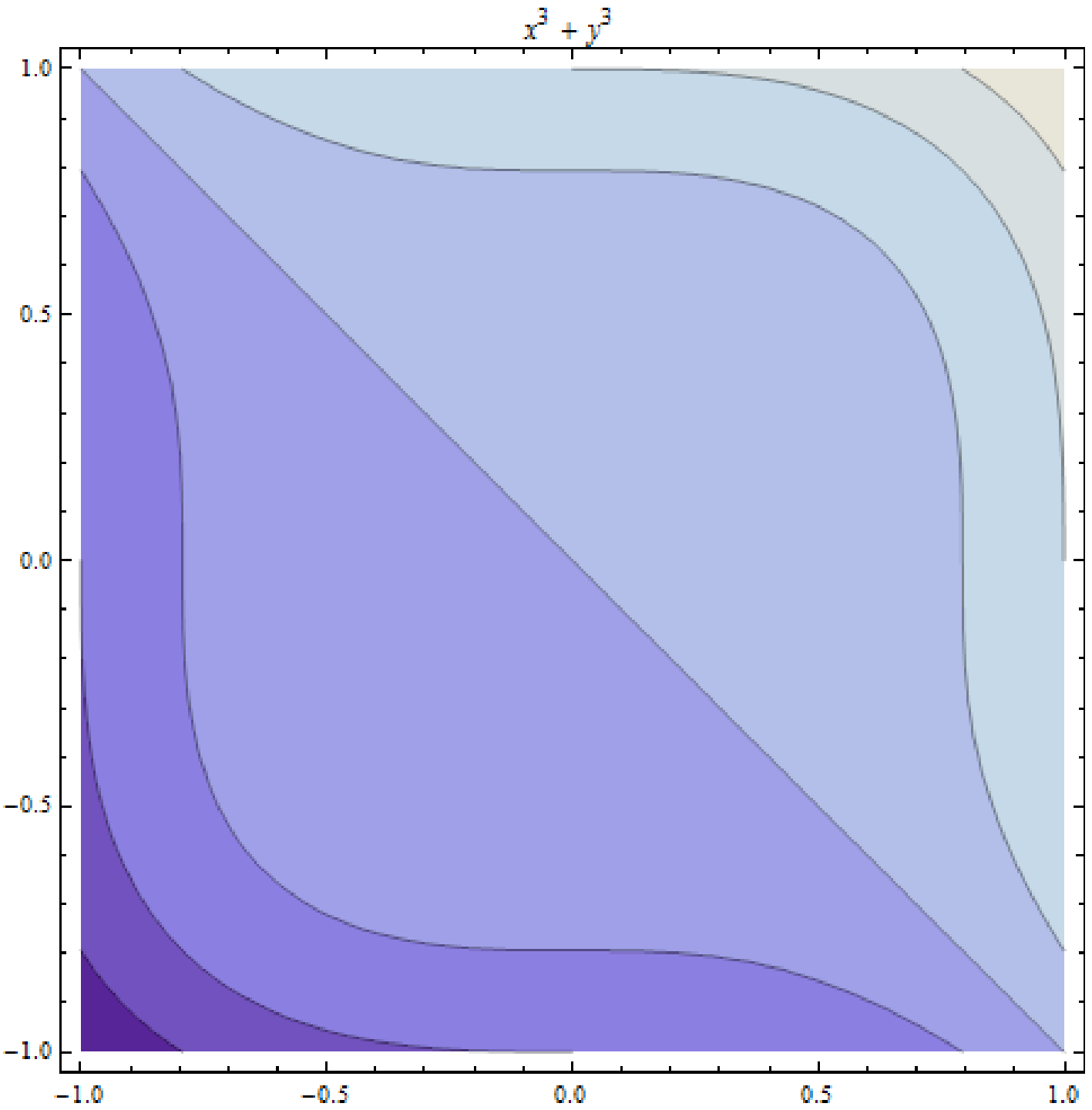
x^2+y^2

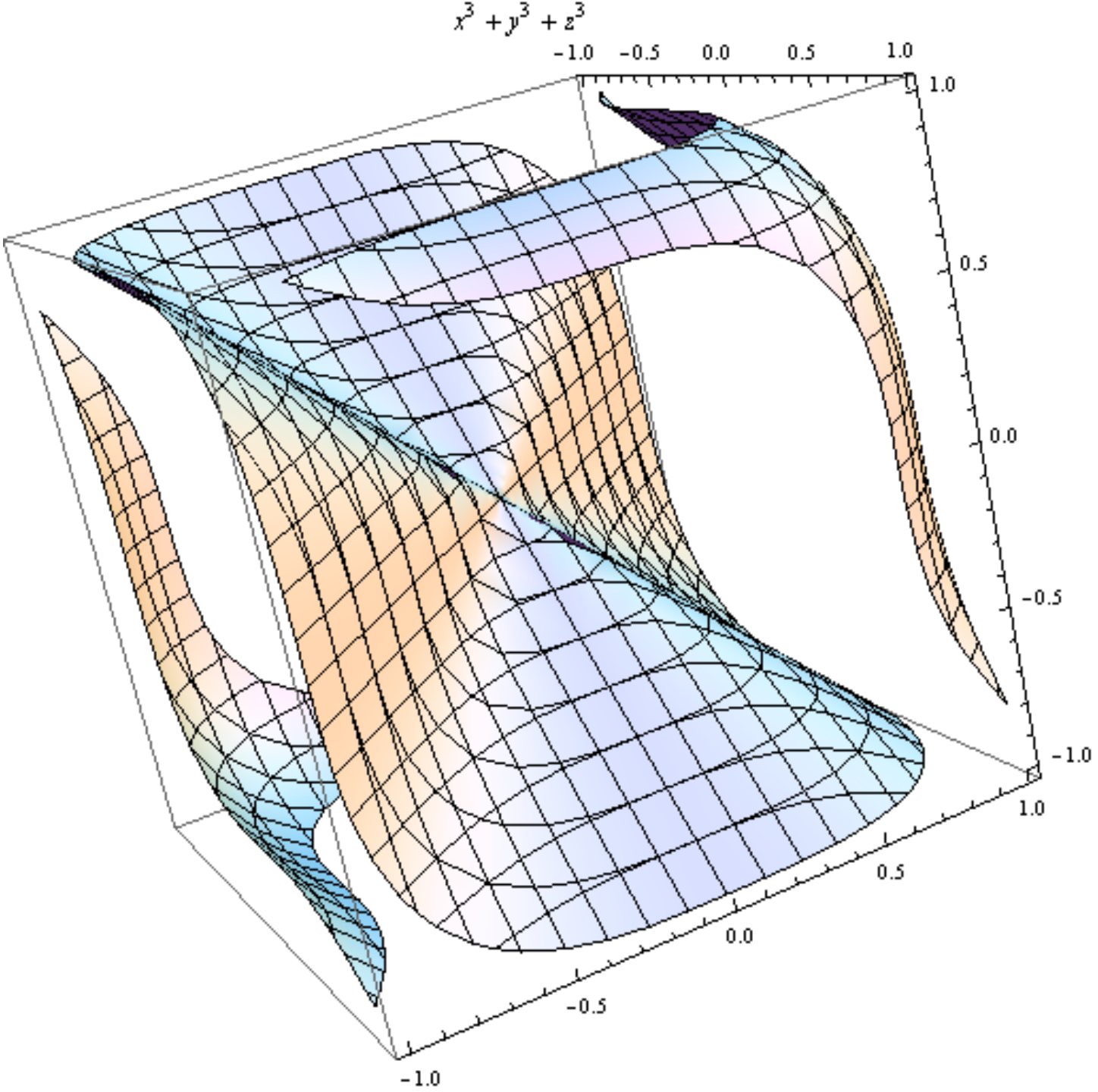


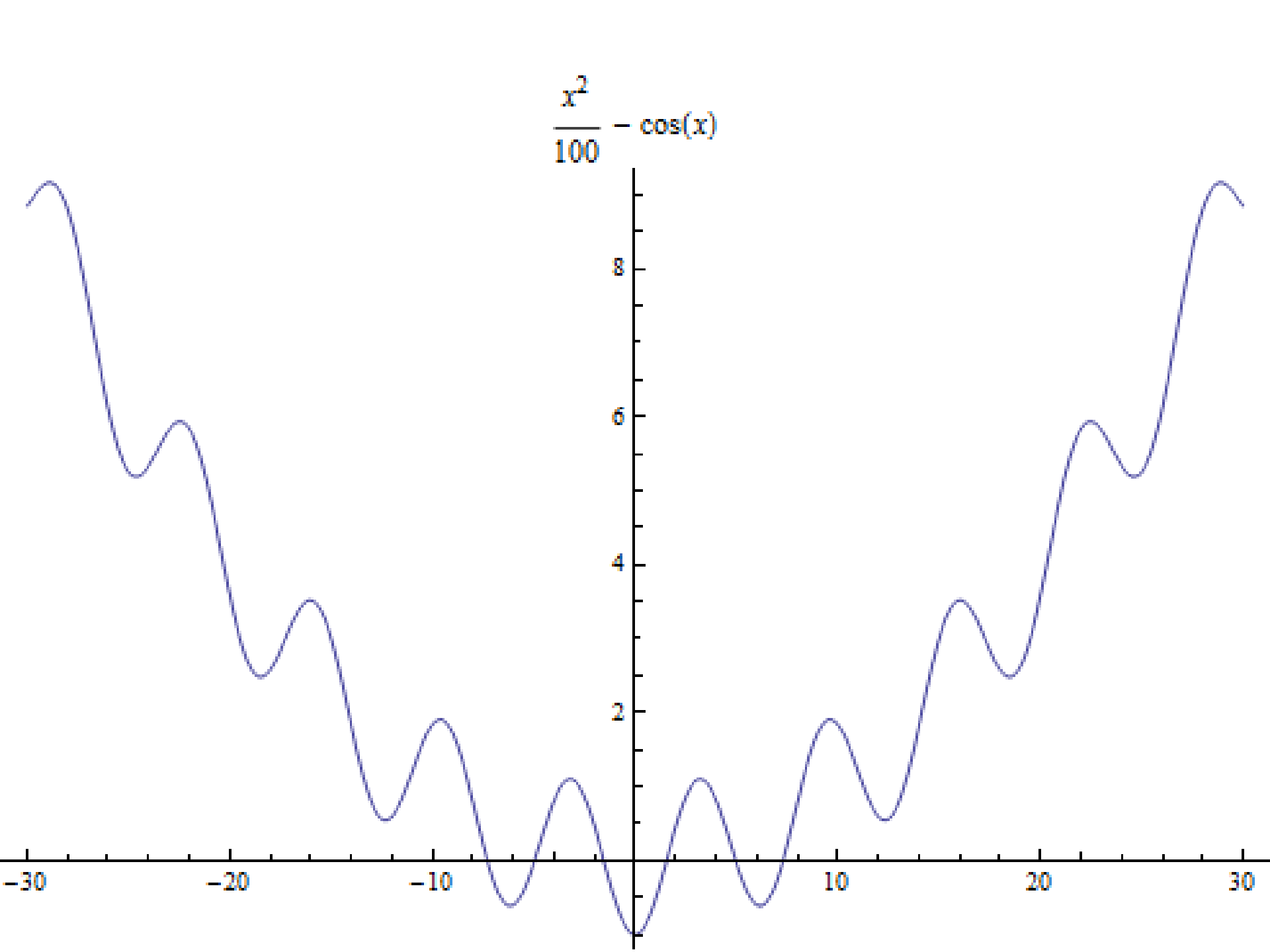

$x^2 + y^2$
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

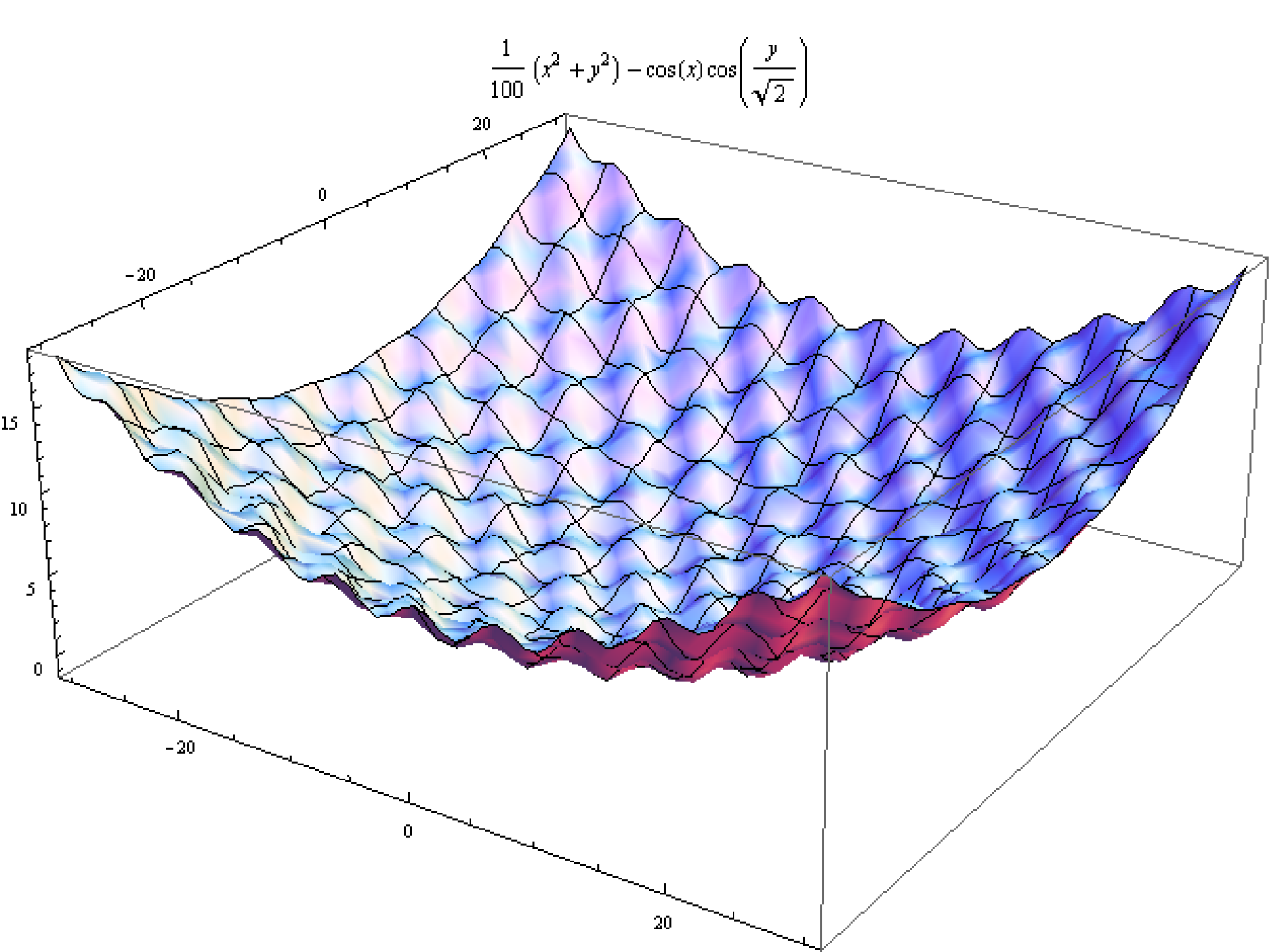

$x^2 + y^2 + z^2$
1.0
0.5
0.0
−0.5
−1.0
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0


1.0
0.5
0.0
−0.5
−1.0
2
1
0
−1
−2
1.0
0.5
0.0
−0.5
−1.0

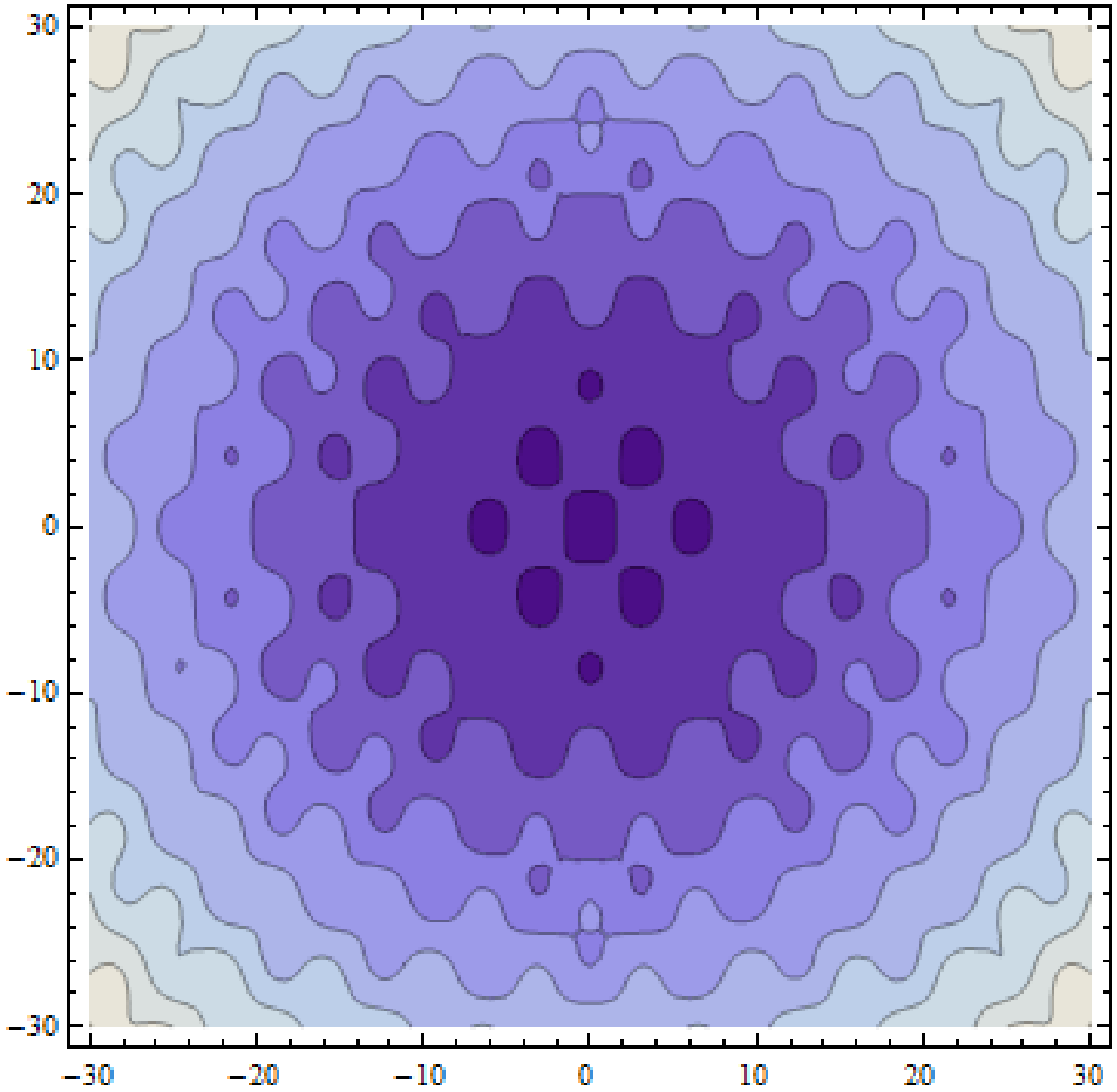

$x^3 + y^3$
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0


$x^3 + y^3 + z^3$
-1.0
-0.5
0.0
0.5
1.0
1.0
0.5
0.0
-0.5
-1.0
1.0
0.5
0.0
-0.5
-1.0

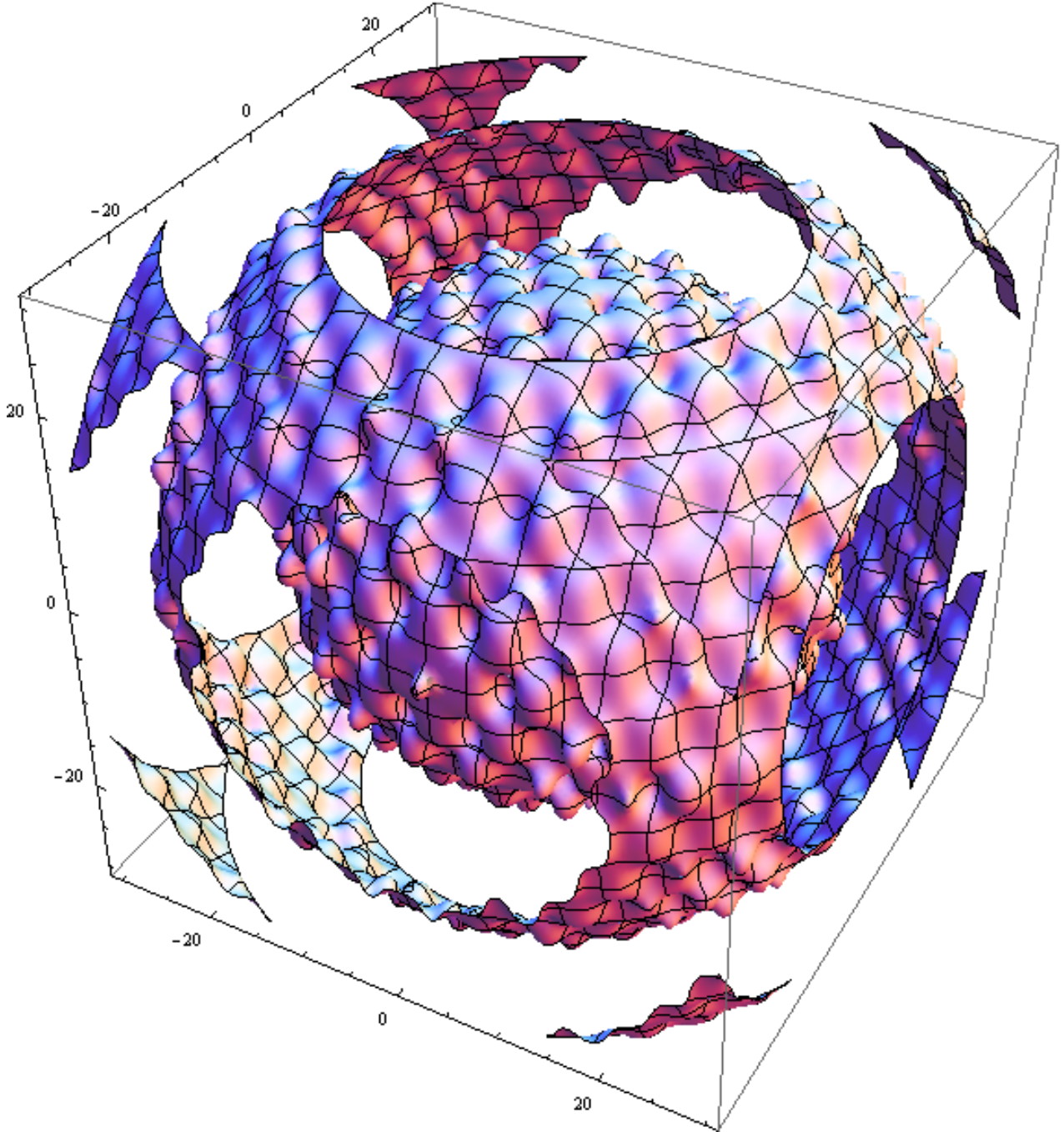

$\frac{x^2}{100} - \cos(x)$
8
6
4
2
−30
−20
−10
10
20
30


$\frac{1}{100}(x^2+y^2)-\cos(x)\cos\left(\frac{y}{\sqrt{2}}\right)$
20
0
-20
15
10
5
0
-20
0
20

$$\frac{1}{100}\left(x^2+y^2\right)-\cos(x)\cos\left(\frac{y}{\sqrt{2}}\right)$$



$$\frac{1}{100}\left(x^2+y^2+z^2\right)-\cos(x)\cos\left(\frac{y}{\sqrt{2}}\right)\cos\left(\frac{z}{\sqrt{3}}\right)$$



Méthode de la dérivée

- Lorsque l'on sait calculer explicitement les zéros de la dérivée (ou du gradient), on sait que f'(X)=0 est une condition nécessaire pour que X soit un point extrémal hors des frontières de l'espace de définition.
- Cette méthode ne fonctionne que pour des fonctions analytiques simples.

Méthodes déterministes locales: le gradient

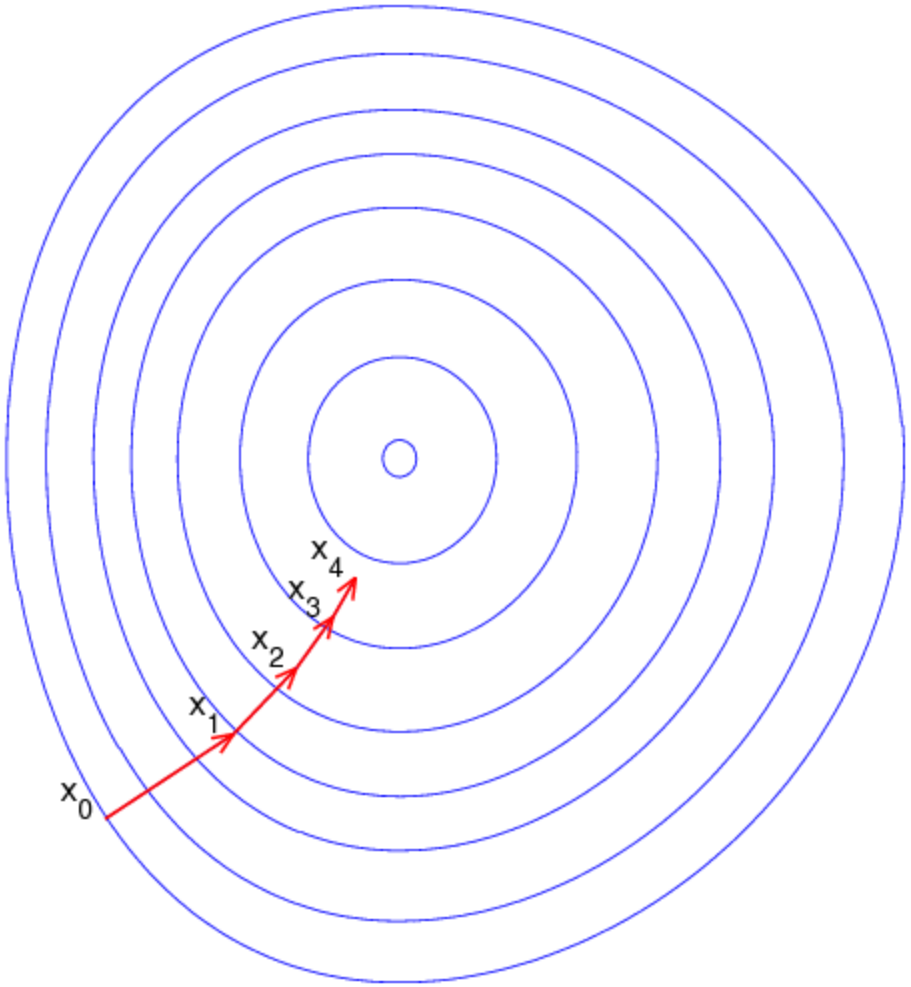
Soit f(X) fonction d'un vecteur X, à valeurs réelles, et son gradient f'(X). On calcule la suite:

$X_{n+1} = X_n - a\, f'(X_n),\ a>0$

Le choix de a est idéalement fait en minimisant pour a>0:

$G(a)=f(X_n - a\, f'(X_n))$

Ceci n'est généralement pas possible, et on utilise pour trouver a des méthodes approchées.



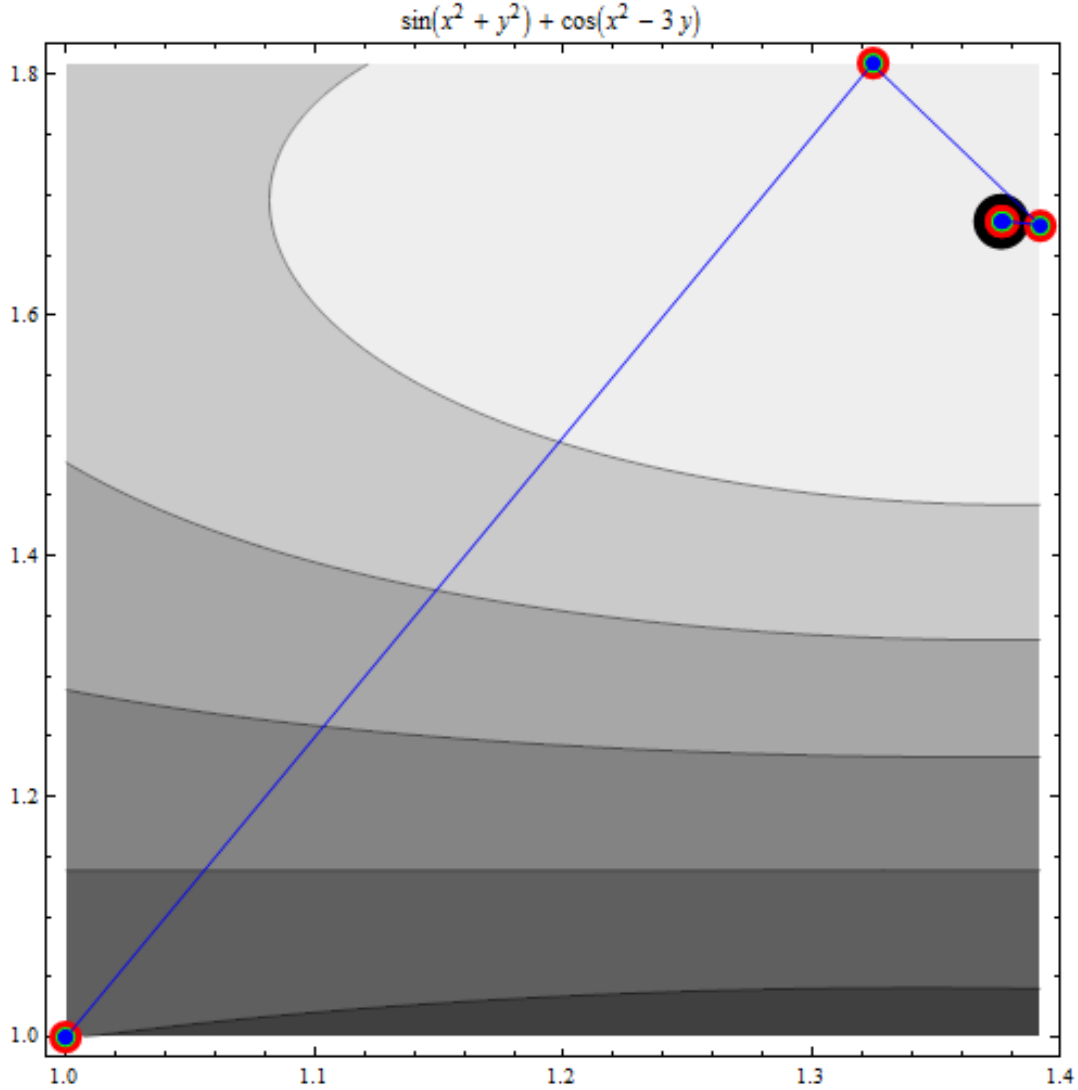
Méthodes déterministes locales d'ordre 2

- Pour accélérer la descente, on utilise les informations apportées par la dérivée seconde de la fonction (le Hessien pour les fonctions à plusieurs variables)
- Ces méthodes nécessitent de calculer la dérivée et le Hessien simultanément.

Méthodes déterministes locales d'ordre 2

- $f(y) = f(x) + f'(x)\,(y-x) + \frac{1}{2} f''(x)\,(y-x)^2 + d$
- En minimisant la forme quadratique en y:
  - $f'(x)+f''(x)(y-x)=0$ soit $y = x - f'(x)/f''(x)$
- Algorithme:
  - $x_{n+1} = x_n - f'(x_n) / f''(x_n)$
- Méthode connue en général sous le nom de méthode de Newton.
- Sa convergence est plus rapide que la méthode de gradient.

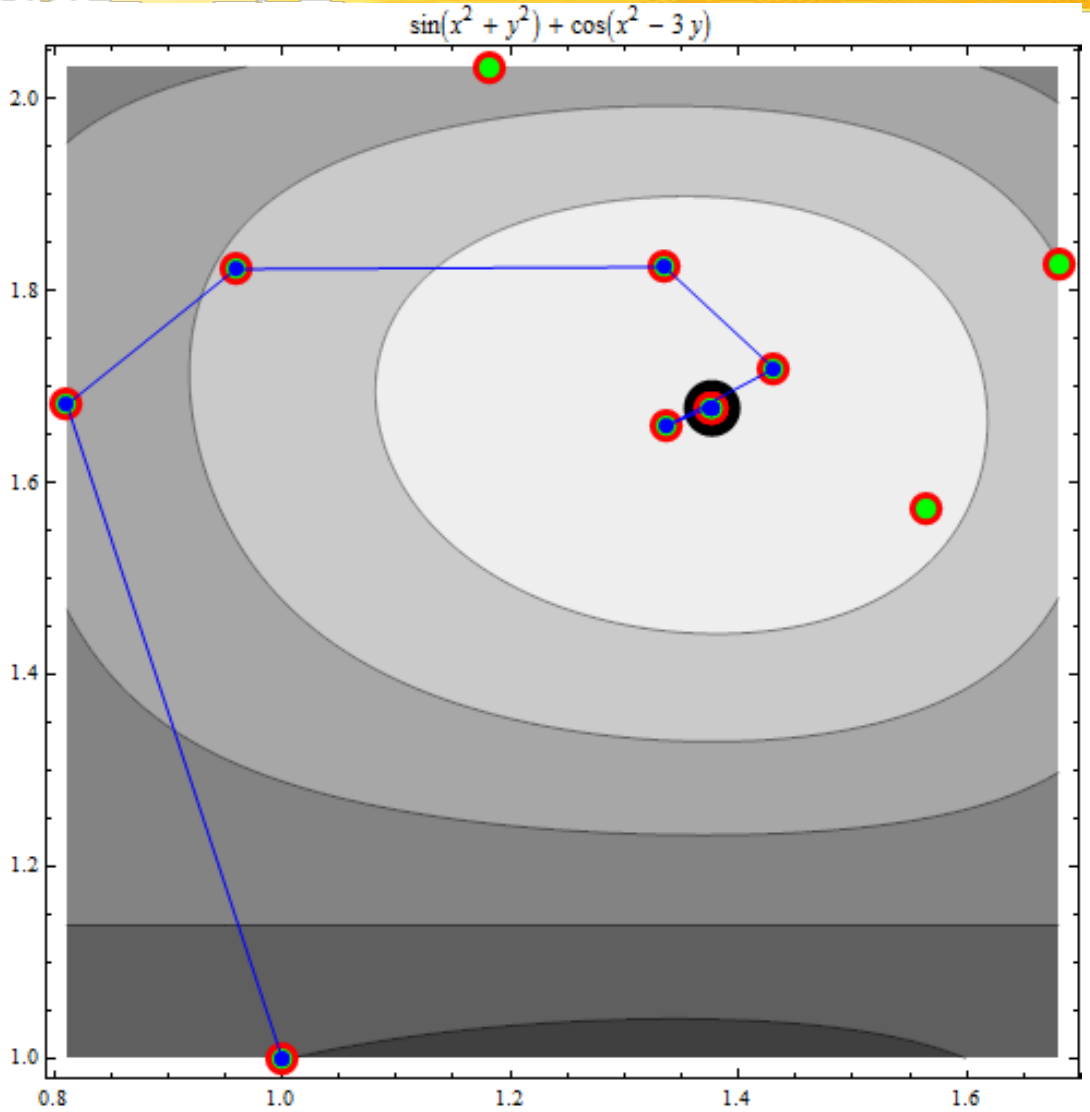
Méthodes déterministes locales d'ordre 2 (Newton)



Méthodes déterministes locales: BFGS

- BFGS est une méthode qui approxime la matrice du hessien par analyse des gradients successifs
- Elle ne nécessite que la connaissance du gradient.
- Elle est plus rapide que le gradient, et moins rapide que la méthode de Newton
- C'est une méthode très utilisée en pratique.

BFGS



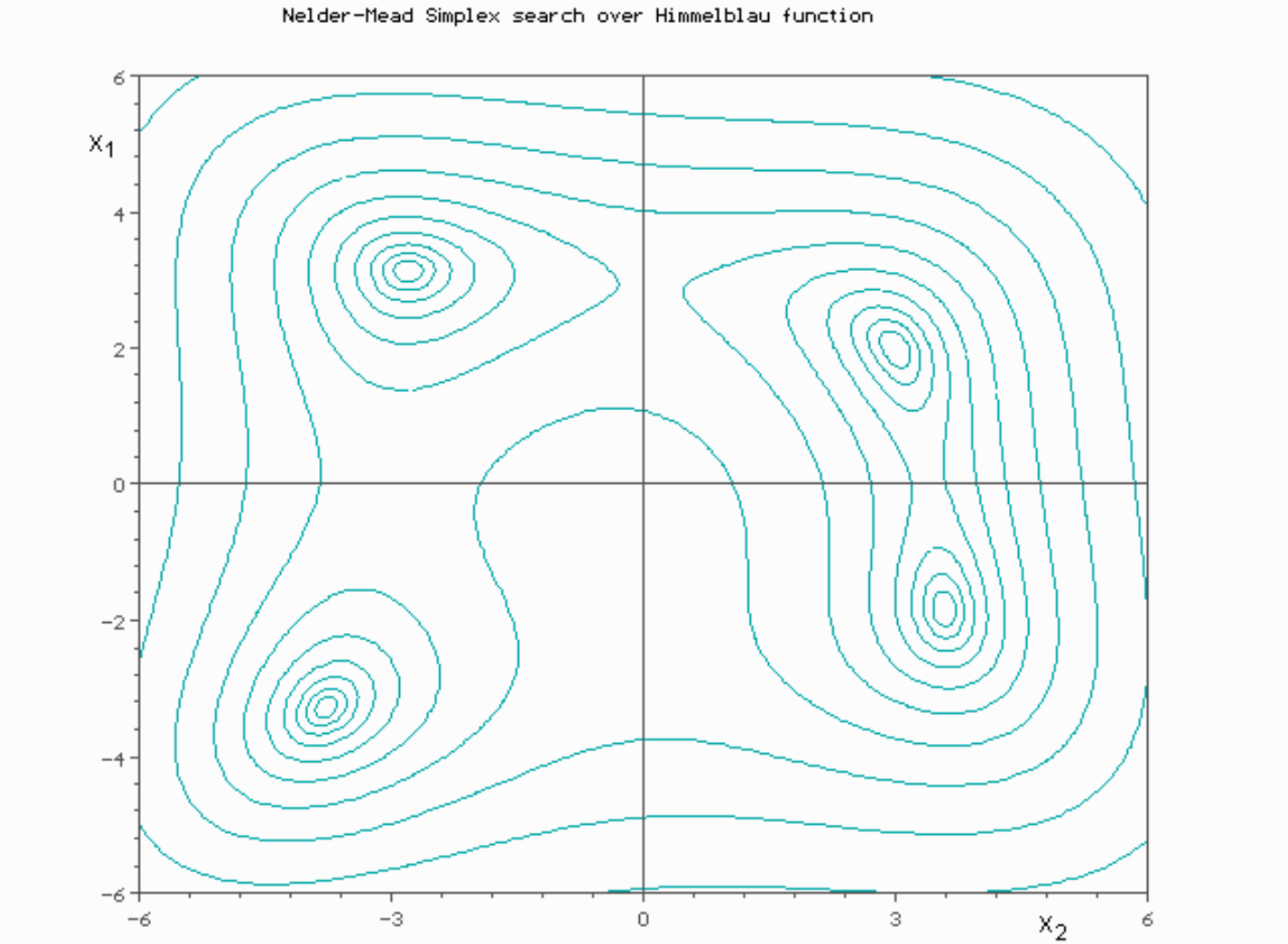
Méthodes locales: le simplexe de Nelder-Mead

- Méthode travaillant sur un polytope: pour une fonction de n variables, on choisit n+1 points et on les fait évoluer par extension et contraction.
- Méthode qui ne nécessite pas le calcul du gradient ou du hessien, et donc très pratique sur les fonctions complexes.
- Les preuves de convergence sont peu claires
- L'algorithme est simple et pratique.

Le simplexe de Nelder-Mead

- Choix de n+1 points $(x_1,..x_{n+1})$
- Tri du simplexe: $f(x_1)<f(x_2)...<f(x_{n+1})$
- Barycentre de n points: $x_0 = (x_1+...+x_n)/n$
- Réfléchi de x_{n+1}/x_0: $x_r=x_0+(x_0-x_{n+1})$
- Si $f(x_r)<f(x_1)$, $x_e=x_0+2(x_0-x_{n+1})$. Si $f(x_e)<f(x_r)$, $x_{n+1}<-x_e$, sinon $x_{n+1}<-x_r$. Retour au tri.
- Si $f(x_n)<f(x_r)$, $x_c=x_{n+1}+(x_0-x_{n+1})/2$.Si $f(x_c)<f(x_r)$ $x_{n+1}<-x_c$, retour au tri
- Sinon: x_i <- $x_0+(x_i-x_1)/2$. Retour au tri.

Nelder Mead


Nelder-Mead Simplex search over Himmelblau function
6
4
2
0
-2
-4
-6
x_1
-6
-3
0
3
6
x_2
(c) P.A. Simionescu 2006

Optimisation stochastique

- Méthodes d'optimisation qui ne requièrent pas de régularité sur les fonctions à optimiser
- Méthodes couteuses en temps de calcul qui ne garantissent pas de trouver l'optimum.
- Les résultats de convergence ne s'appliquent pas en pratique.

Le recuit simulé

- Initialisation: on part d'un point x_0 choisi au hasard dans l'espace de recherche.
  - On construit $x_{n+1}=x_n+B(0,s)$
  - On fait évoluer la température de recuit: $t_{n+1}=H(t_n)$
  - Si $f(x_{n+1})<f(x_n)$ alors on conserve x_{n+1}
  - Si $f(x_{n+1})>f(x_n)$ alors :
    - Si $|f(x_{n+1})-f(x_n)|<e^{-k\,t}$ alors on conserve x_{n+1}
    - Si $|f(x_{n+1})-f(x_n)|>e^{-k\,t}$ alors on conserve x_n

Paramètres importants

- Le schéma de recuit H détermine la façon dont l'algorithme converge.
  - Trop rapide=>L'algorithme converge vers un minimum local
  - Trop lent=>L'algorithme converge trop lentement.
- Le déplacement B(0,s) doit balayer suffisamment l'espace sans trop déplacer le point.

Efficacité

- Les algorithmes de recuit sont utiles sur des problèmes trop difficiles pour les techniques déterministes.
- On leur préfèrera des algorithmes de type génétique quand on peut construire des croisements qui ont un « sens ».

Algorithmes génétiques

- Techniques d'optimisation s'appuyant sur des techniques dérivées de la génétique et de l'évolution naturelle:
  - Reproduction
  - Croisement
  - Mutation
- Apparus aux Etats-Unis dans les années 60 à travers les travaux de John Holland
- Popularisés par David Goldberg.

Codage d'un élément et création de population

- Soit x, variable de la fonction f(x) à optimiser sur $[x_{min},x_{max}]$.
- On réécrit x : 2^n $(x-x_{min})/(x_{max}-x_{min})$
- On obtient alors un nombre compris dans l'intervalle $[0,2^n]$, soit une chaine de n bits:
  - Pour n=8: 01001110
  - Pour n=16: 0100010111010010
- On tire n éléments au hasard et les code comme ci-dessus.

Croisement

- On choisit deux parents :
  - 01100111
  - 10010111
- On tire au sort un site de croisement (3):
  - 011|00111
  - 100|10111
- On récupère les deux enfants:
  - 011|10111
  - 100|00111

Mutation

- On sélectionne un élément:
  - 01101110
- On sélectionne un site de mutation (5):
  - 01101110
- On inverse la valeur du bit:
  - 01100110

Reproduction

- Pour chaque élément x_i on calcule
  - $f(x_i)$ et $S=\Sigma(f(x_i))$
- Pour chaque x_i on calcule
  - $p(x_i)=f(x_i)/\Sigma(f(x_i))$
- On retire les n éléments de la population k+1 à partir des n éléments de la population k en prenant comme probabilité de tirage $p(x_i)$

Exemple de reproduction

- Soit f(x)=4x(1-x)
- x prend ses valeurs dans [0,1[

Séquence	Valeur	$U(x)$	% de chance de reproduction	% cumulés	Après reproduction
10111010	0.7265625	0.794678	0.794678 / 2.595947 = 0.31	0.31	11011110
11011110	0.8671875	0.460693	0.460693 / 2.595947 = 0.18	0.31+0.18=0.49	10111010
00011010	0.1015625	0.364990	0.364990 / 2.595947 = 0.14	0.49+0.14=0.63	01101100
01101100	0.4218750	0.975586	0.975586 / 2.595947 = 0.37	0.62+0.37=1.00	01101100
=		2.595947			

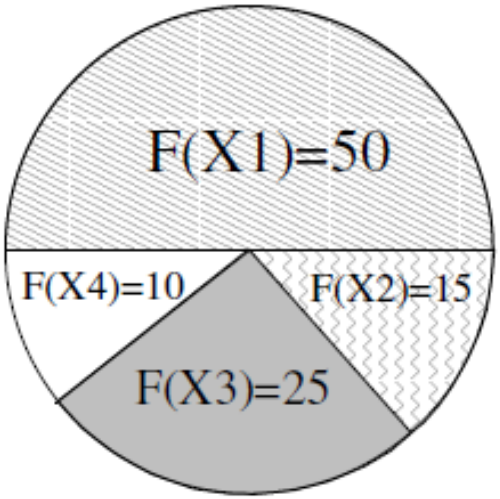
Fonctionnement d'un AG

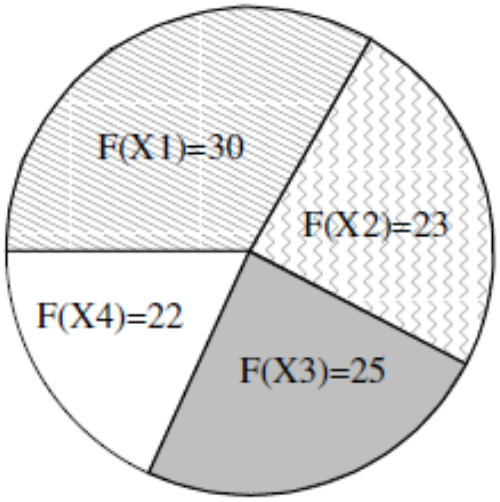
- Etape 1: reproduction
- Etape 2: croisement
- Etape 3: mutation
- Etape 4: test de fin, et retour à l'étape 1.

Le scaling

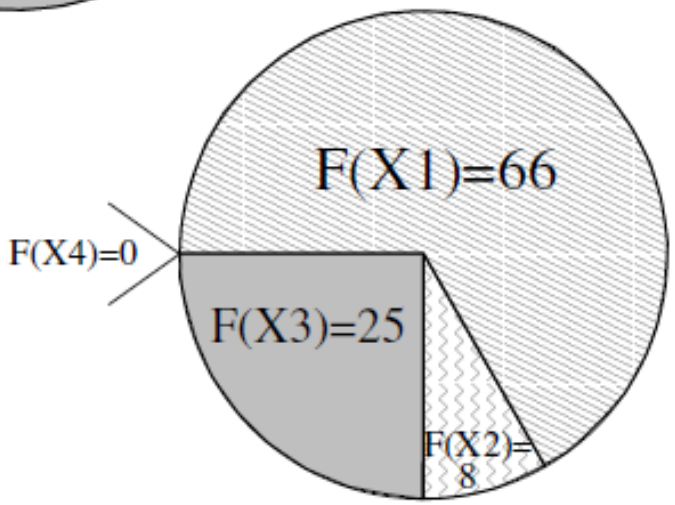
- Le fonctionnement de l'algorithme dépend fortement de la valeur de l'adaptation.
- Au lieu d'utiliser directement f(x) comme adaptation, on la « met à l'échelle » en appliquant une fonction croissante.
- Exemples:
  - 5 (f(x)-10)/3: augmente la pression
  - 0.2 f + 20 : diminue la pression

Exemple de scaling





0.2 F + 20



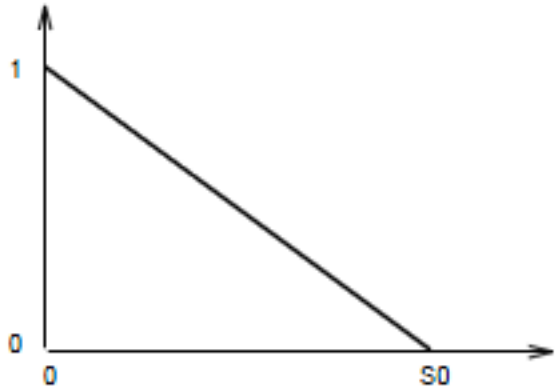
5/3 (F – 10)

Le sharing

- La pression de sélection peut entrainer une convergence locale trop rapide.
- Le sharing modifie l'adaptation en fonction du nombre d'éléments voisins de l'élément courant:
  - $f_s(x_i)=f(x_i)/\Sigma_j\ s(d(x_i,x_j))$
  - s est une fonction décroissante.
  - $d(x_i,x_j)$ mesure la distance entre i et j

Le sharing

- Le sharing demande la mise en place d'une fonction distance sur l'espace des variables.
- Forme générale de s:



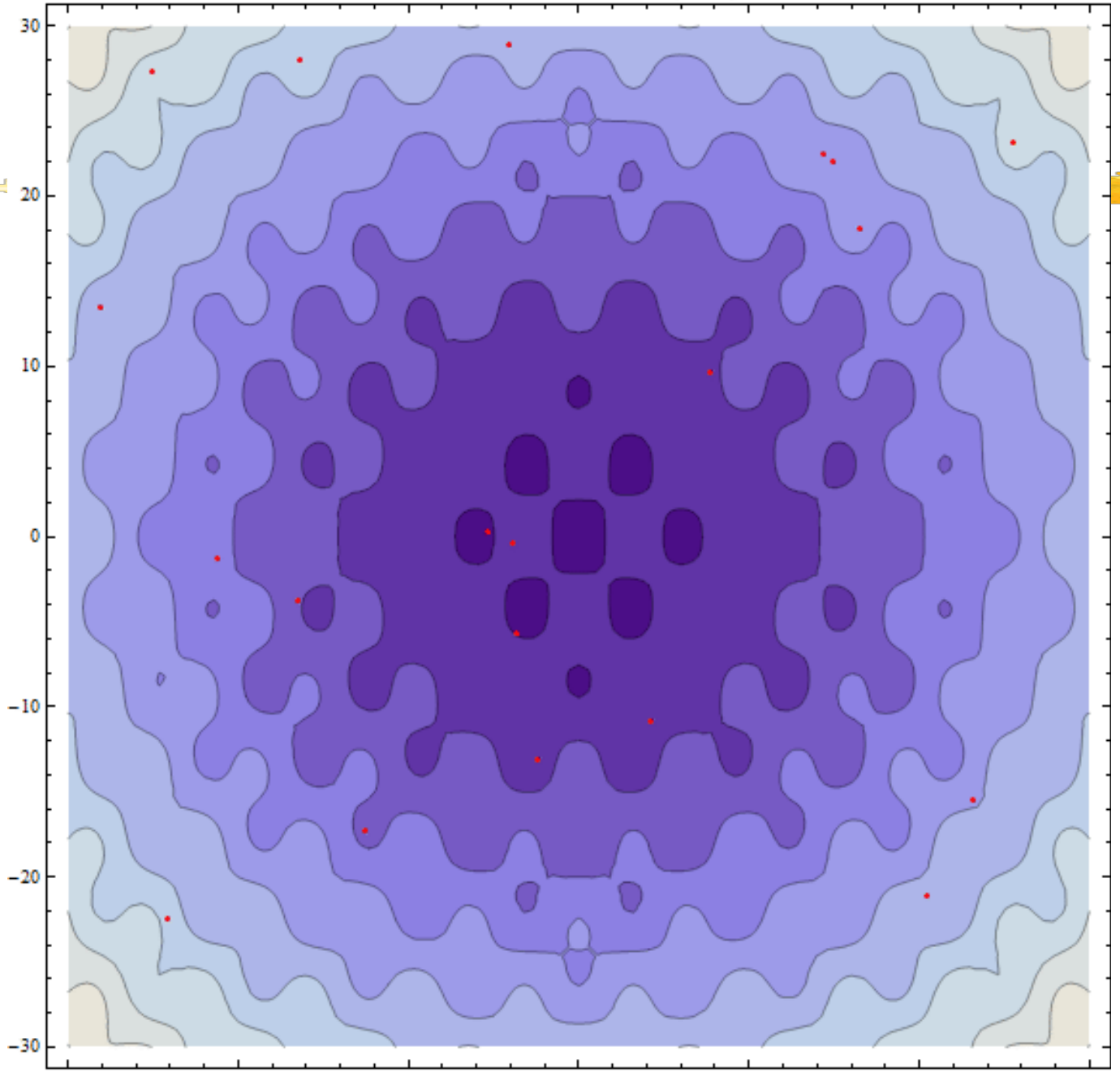
Problème du codage en chaine de bit

- Deux éléments très différents au niveau du génotype peuvent avoir des phénotypes identiques.
  - Sur un codage simple de [0,1] en 8 bits:
  - 10000000 et 01111111 représentent quasiment la même valeur (1/2) mais leur distance de Hamming est maximale.
  - On peut utiliser des codes de Grey, ou employer des représentations adaptées.

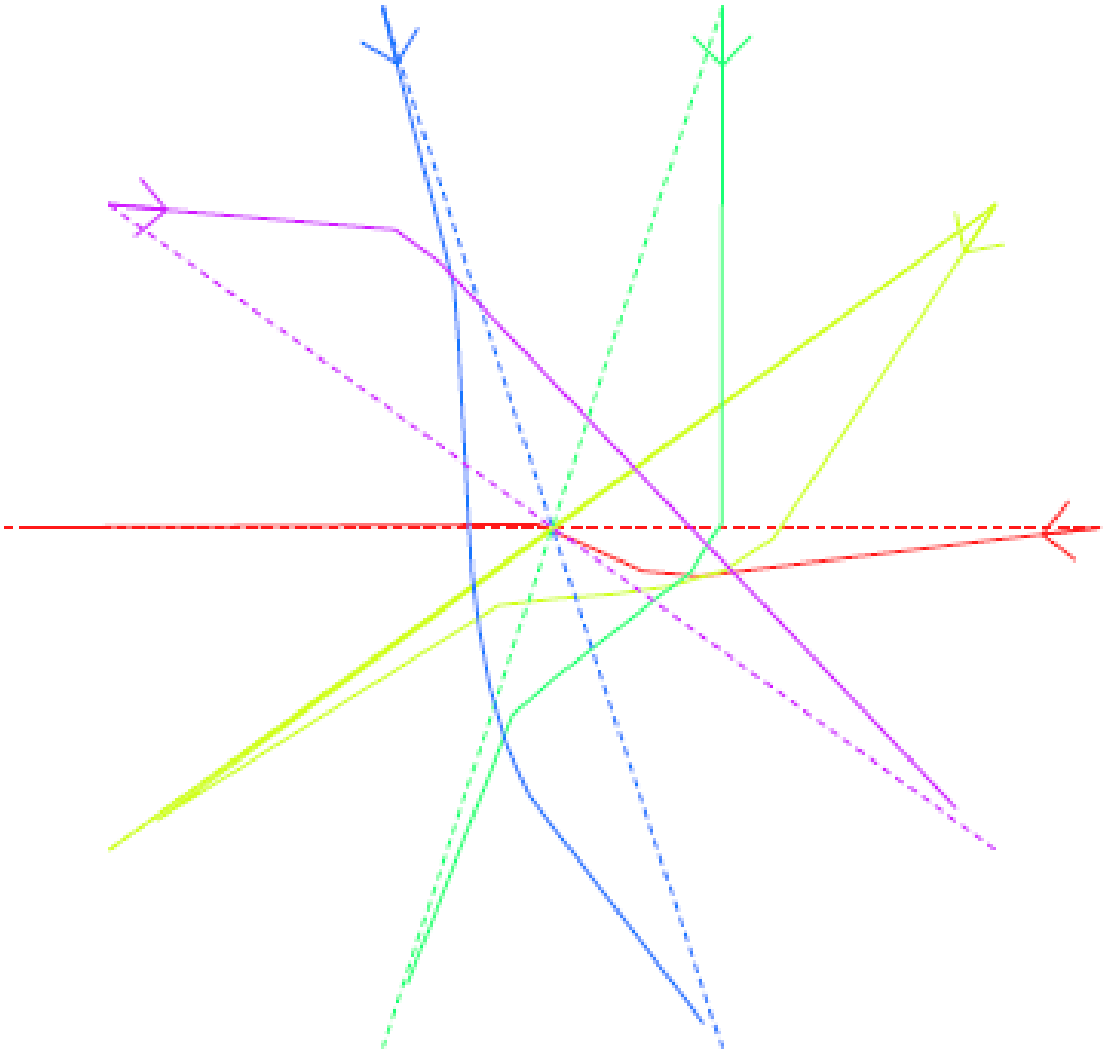
Représentation adaptée

- Pour les fonctions à variable réelle, on code directement la variable par sa valeur
- Croisement:
  - $y_1 = \alpha\, x_1 + (1-\alpha)\, x_2$
  - $y_2 = (1-\alpha)\, x_1 + \alpha\, x_2$
  - α pris dans [0.5,1.5]
- Mutation:
  - $y_1 = x_1 + B(0,\sigma)$

$$\frac{1}{100}\left(x^2+y^2\right)-\cos(x)\cos\left(\frac{y}{\sqrt{2}}\right)$$



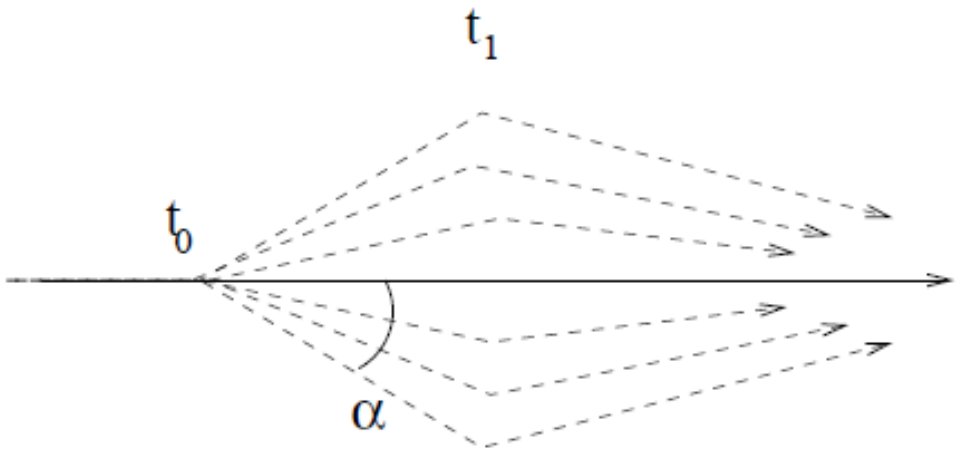
Résolution de conflits aériens



Modélisation

Modélisation

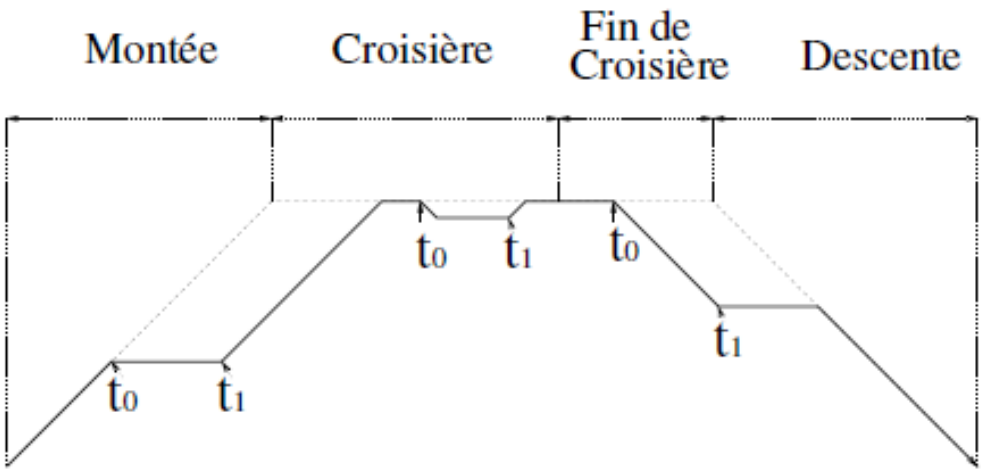
- Une seule manœuvre par avion
- Manoeuvres horizontales
- Manoeuvres verticales
- Modélisation de l'incertitude
- $3n$ variables



Modélisation

Modélisation

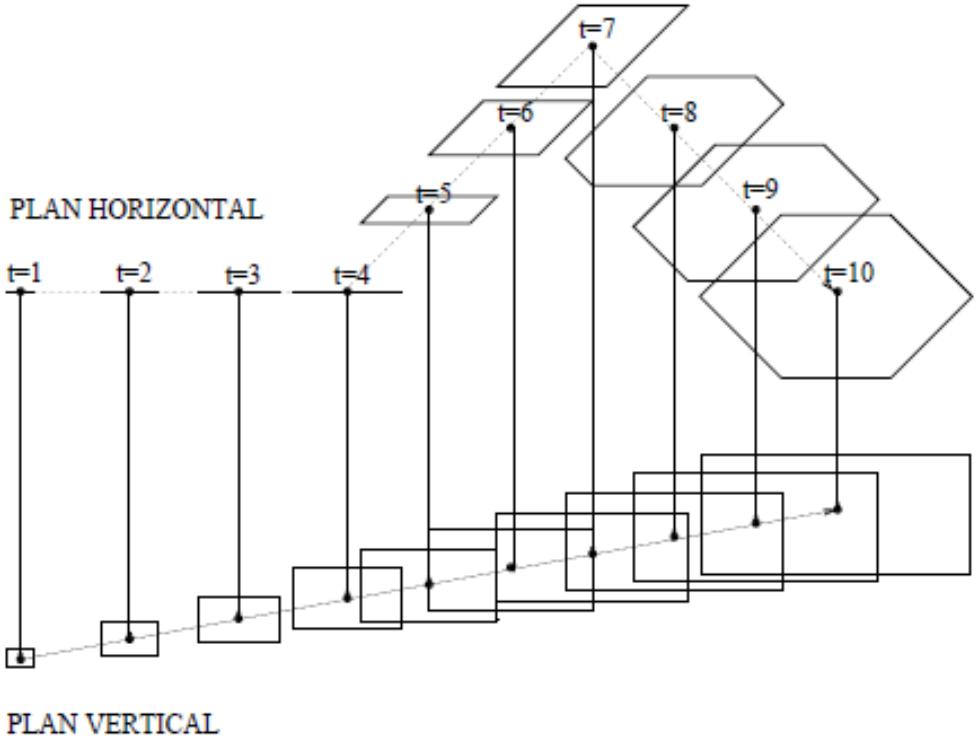
- Une seule manœuvre par avion
- Manoeuvres horizontales
- Manoeuvres verticales
- Modélisation de l'incertitude
- $3n$ variables



Modélisation

Modélisation

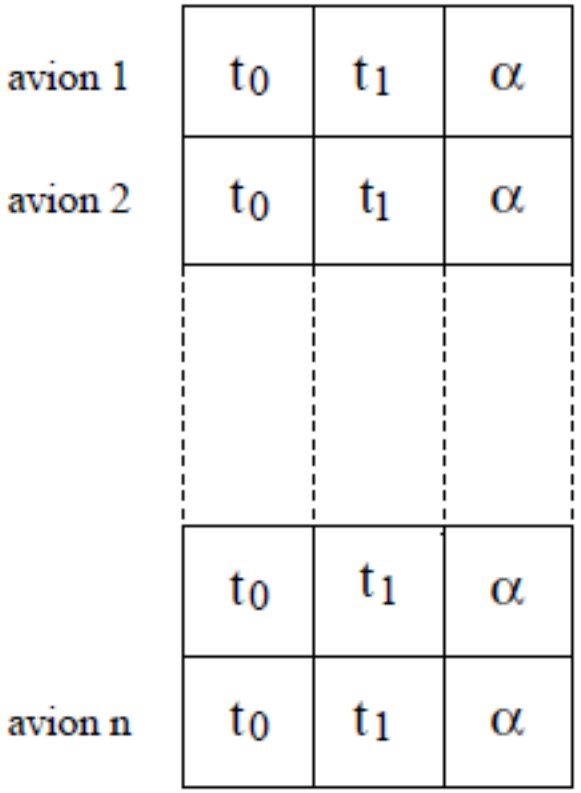
- Une seule manœuvre par avion
- Manoeuvres horizontales
- Manoeuvres verticales
- Modélisation de l'incertitude
- $3n$ variables



Modélisation

Modélisation

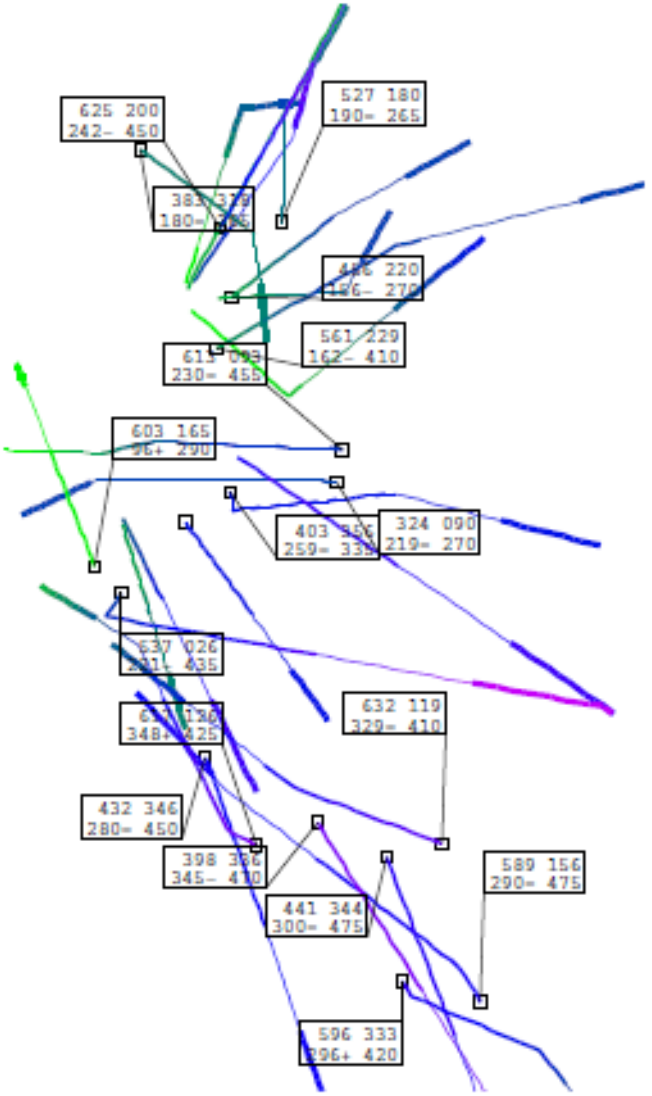
- Une seule manœuvre par avion
- Manoeuvres horizontales
- Manoeuvres verticales
- Modélisation de l'incertitude
- $3n$ variables

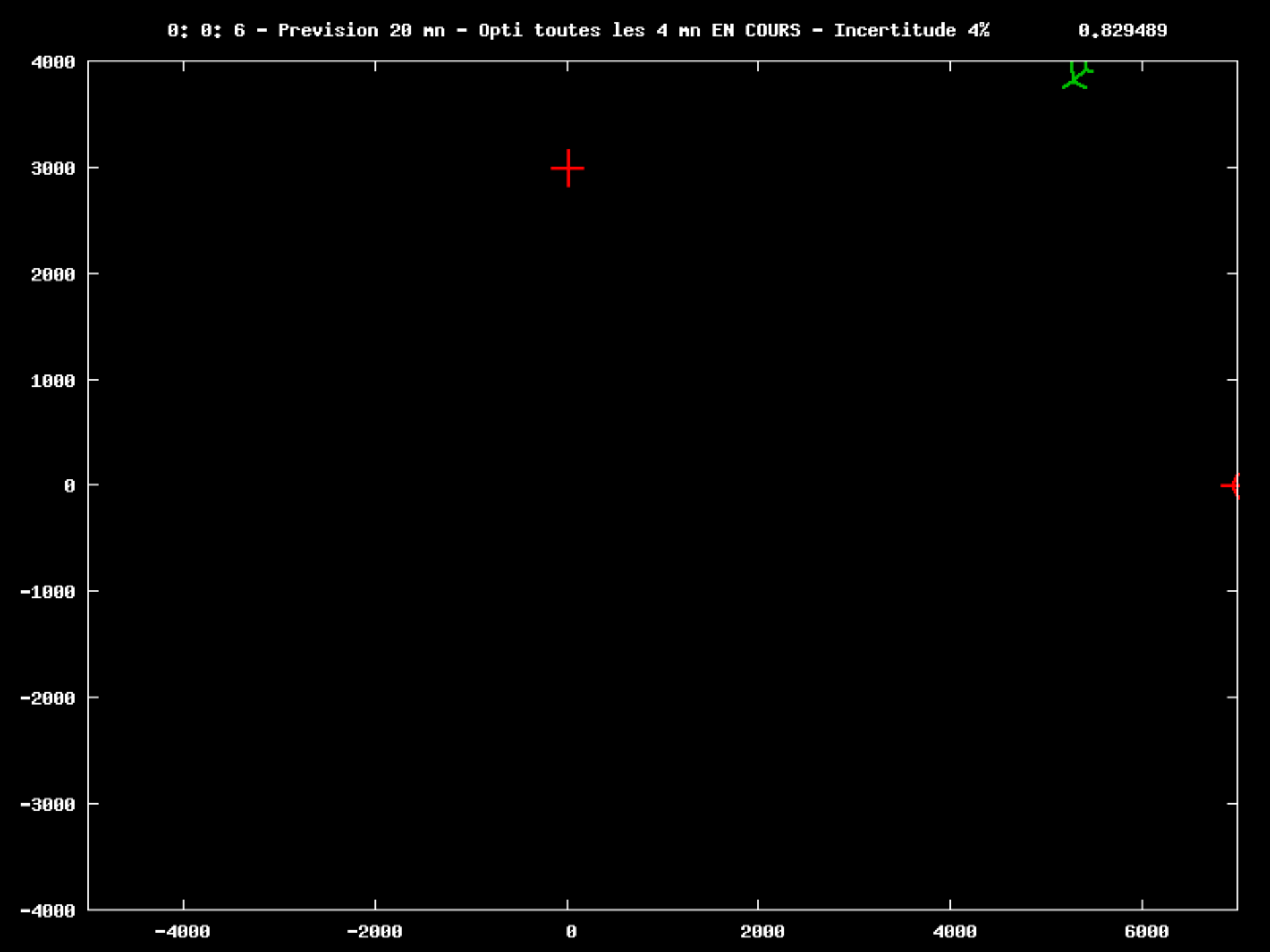


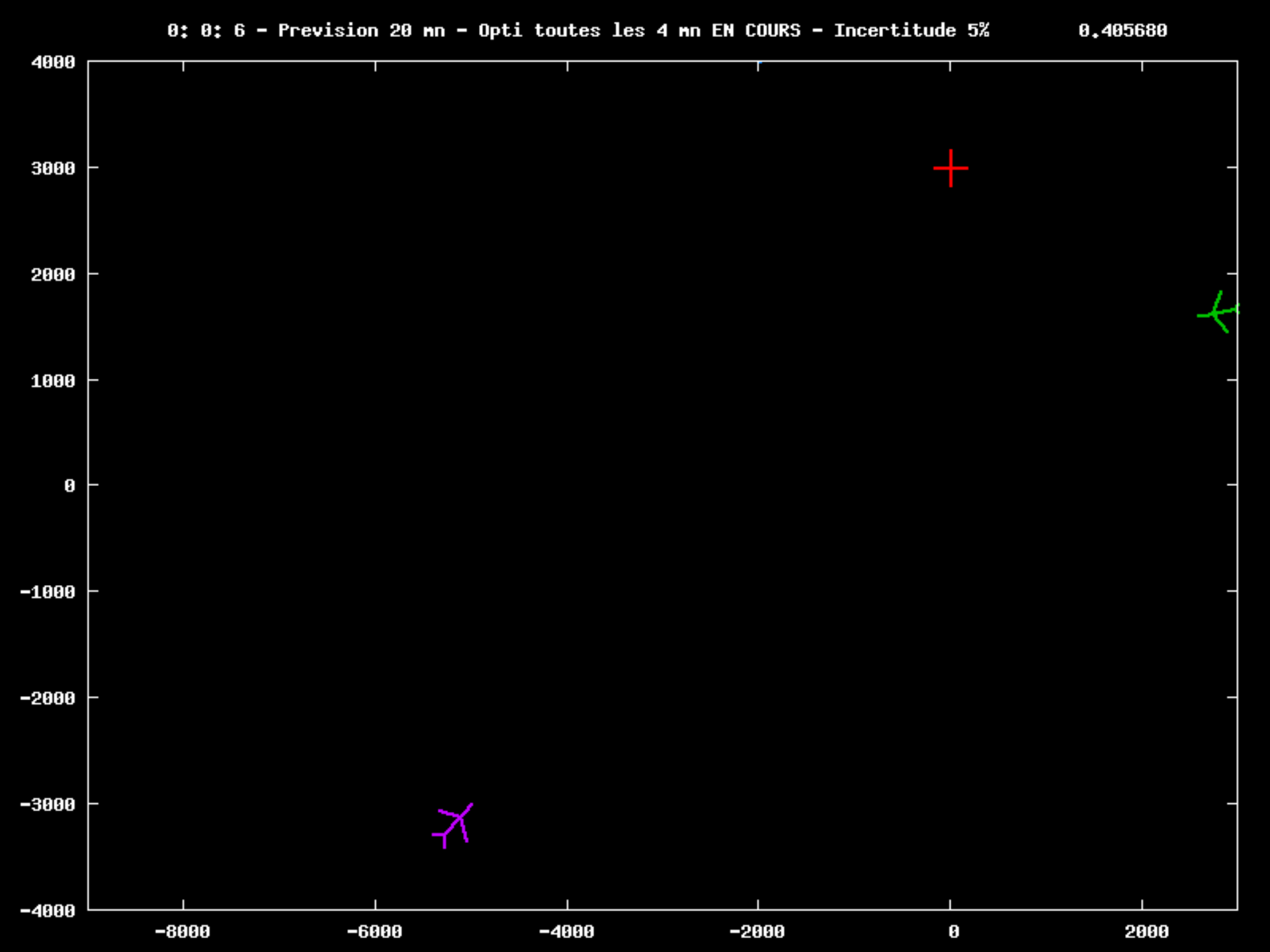
Résultats

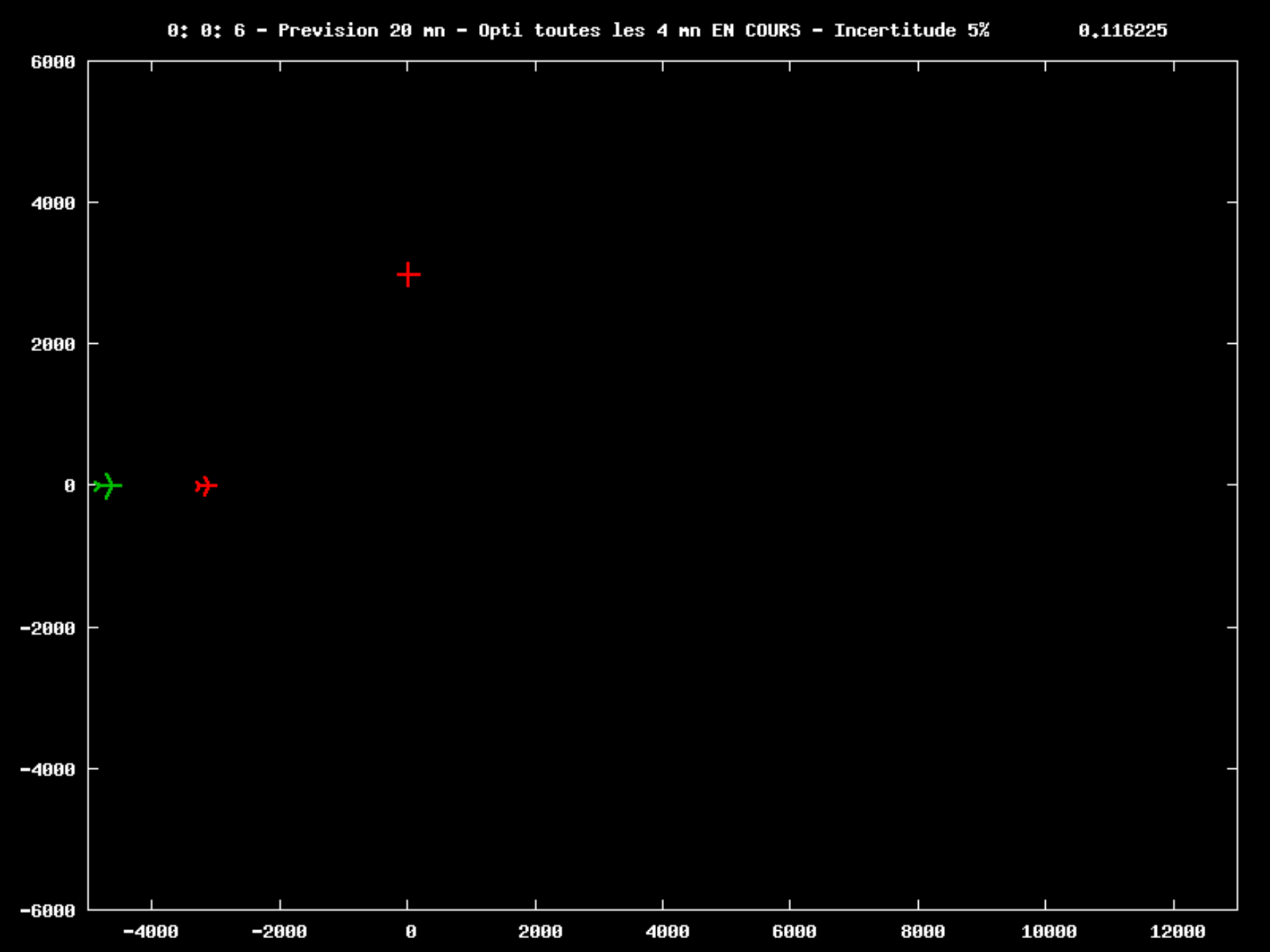
Résultats

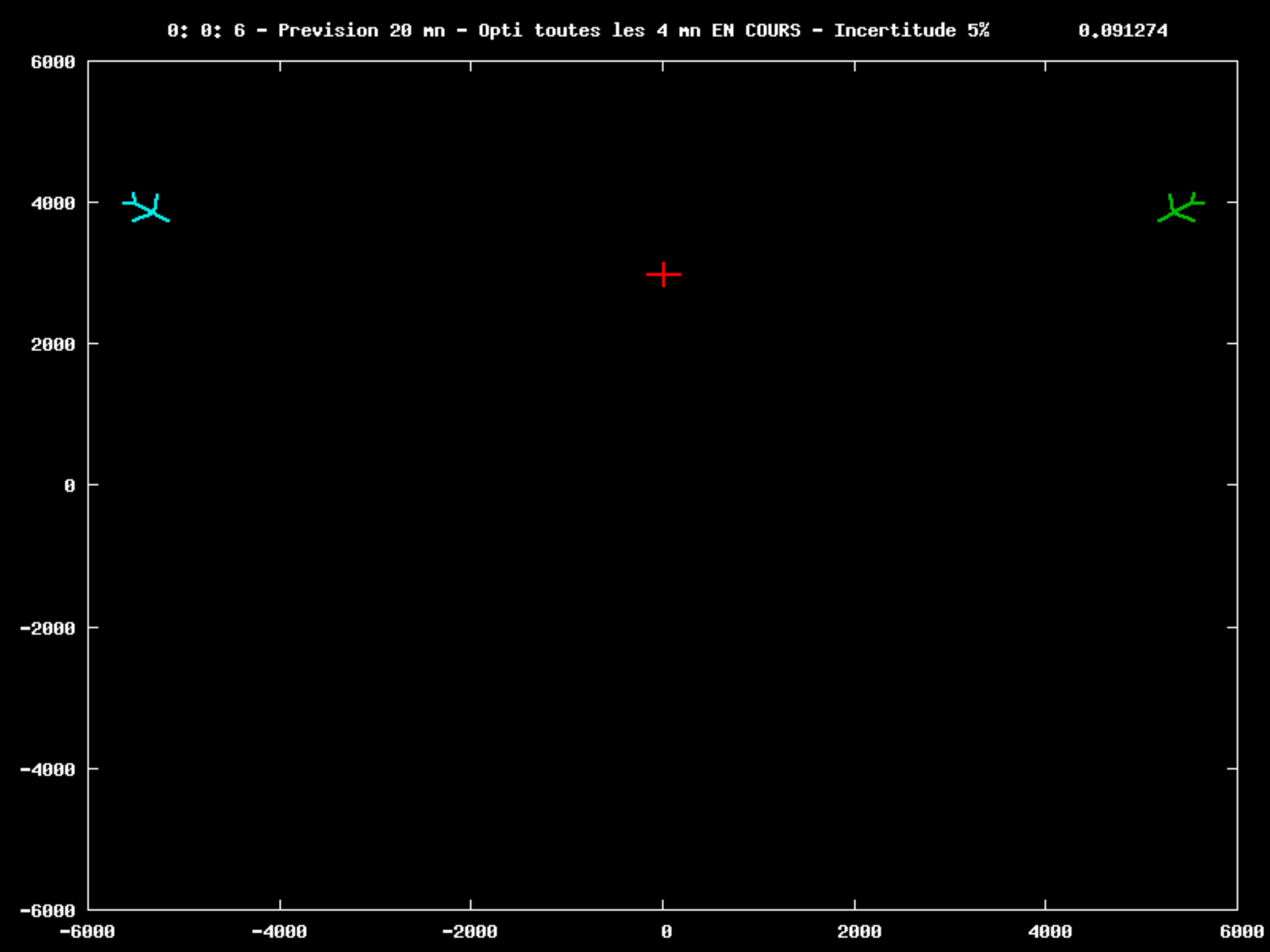
- Résout des gros conflits (30 avions)
- Intégration dans un outil de simulation (CATS/OPAS)
- Testé sur des journées de trafic réel
- Peu de restrictions sur la modélisation
- Pas de garantie d'optimalité

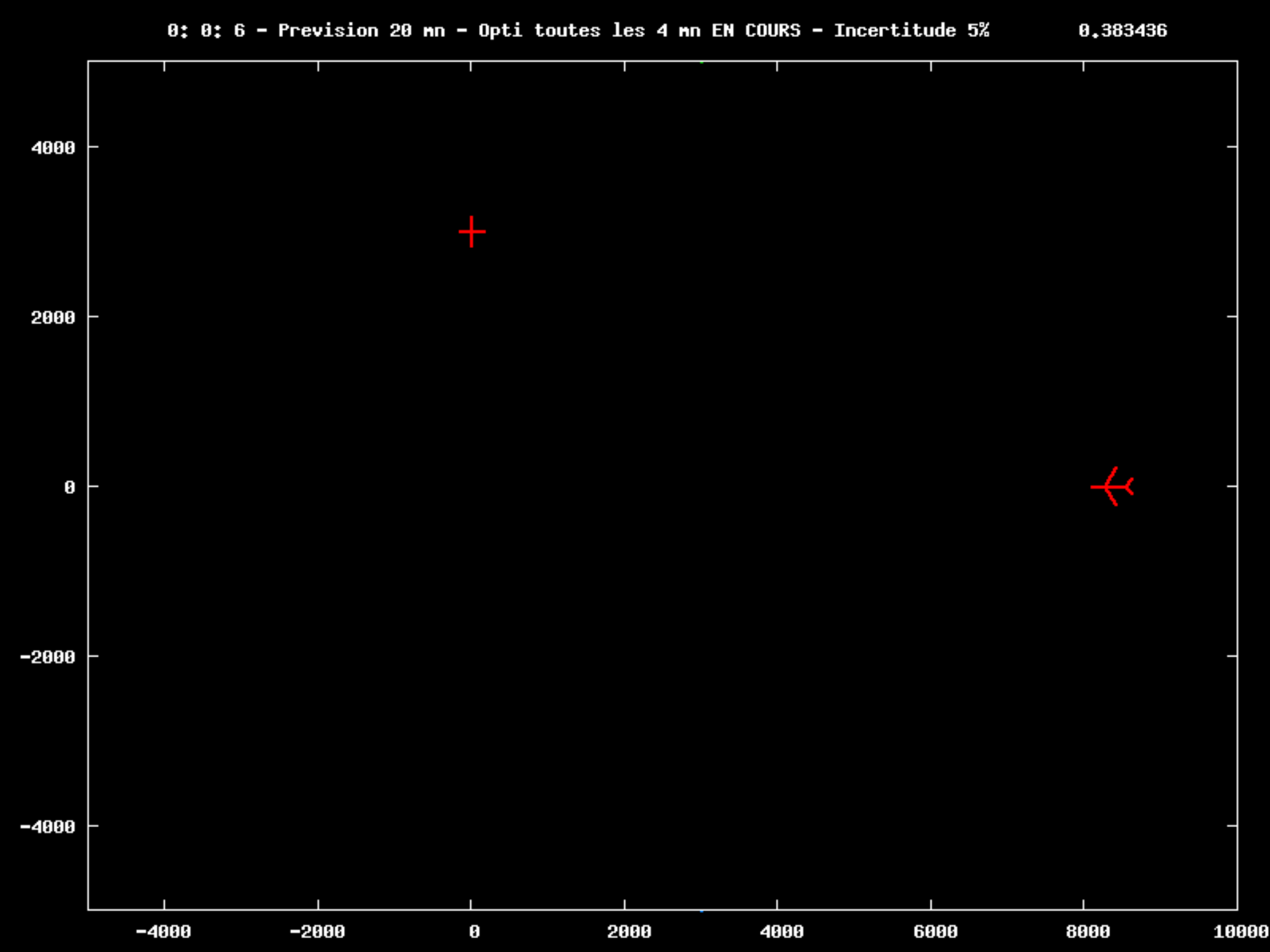


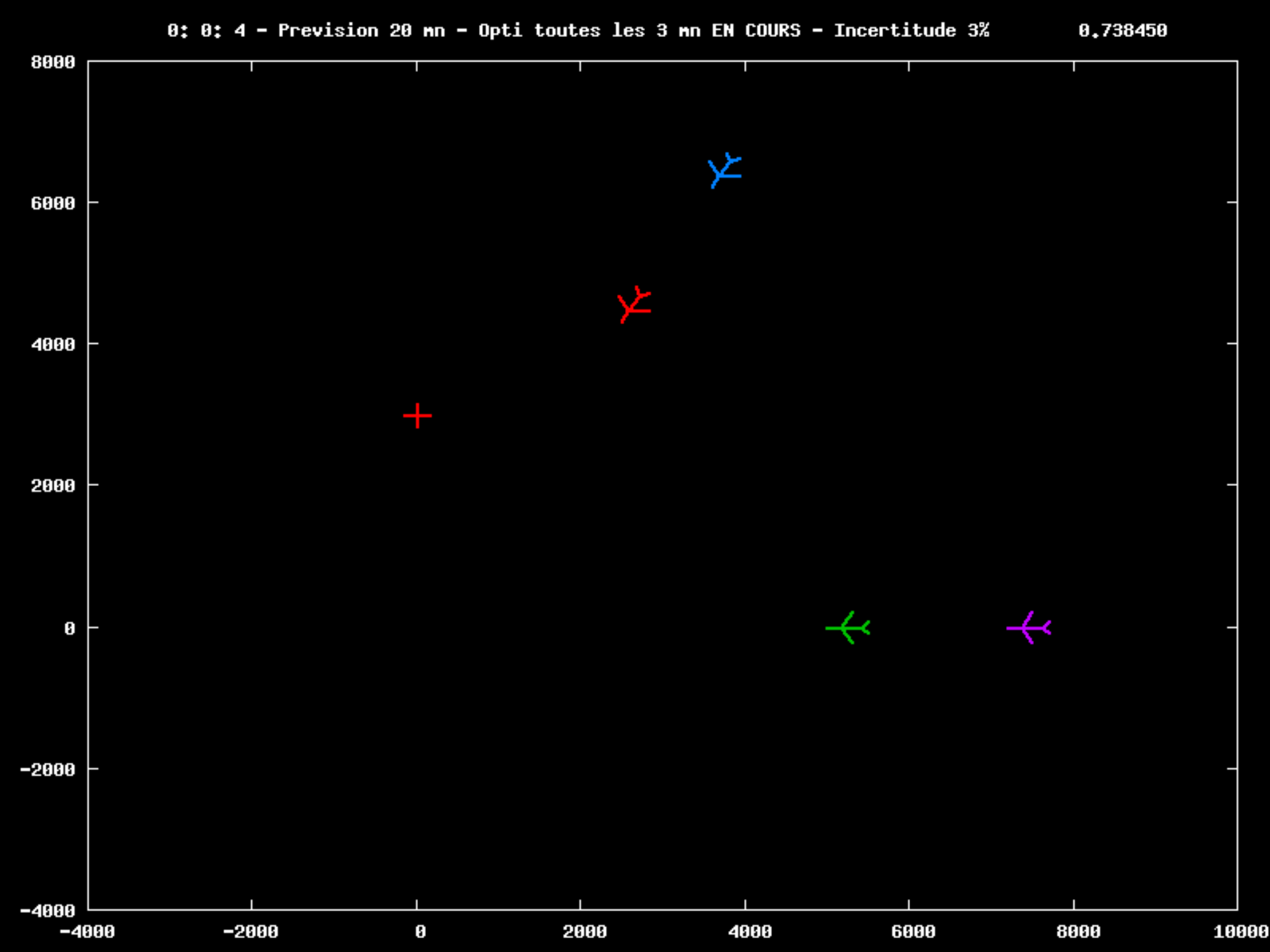

0: 0: 6 - Prevision 20 mn - Opti toutes les 4 mn EN COURS - Incertitude 4%
0.829489
4000
3000
2000
1000
0
-1000
-2000
-3000
-4000
-4000
-2000
0
2000
4000
6000


0: 0: 6 - Prevision 20 mn - Opti toutes les 4 mn EN COURS - Incertitude 5%
0.405680
4000
3000
2000
1000
0
-1000
-2000
-3000
-4000
-8000
-6000
-4000
-2000
0
2000

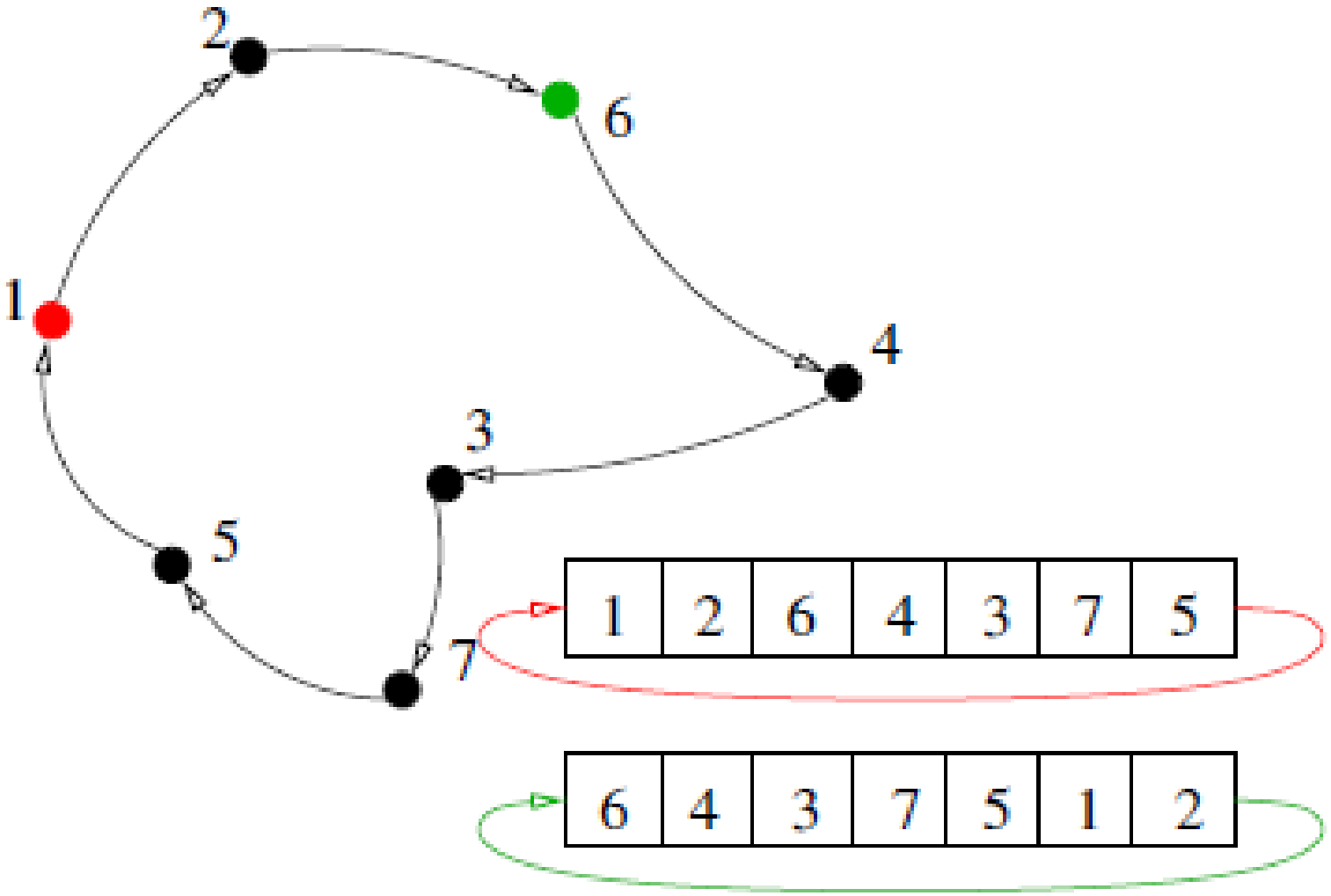

0: 0: 6 - Prevision 20 mn - Opti toutes les 4 mn EN COURS - Incertitude 5%
0.116225
6000
4000
2000
0
-2000
-4000
-6000
-4000
-2000
0
2000
4000
6000
8000
10000
12000


0: 0: 6 - Prevision 20 mn - Opti toutes les 4 mn EN COURS - Incertitude 5%
0.091274
6000
4000
2000
0
-2000
-4000
-6000
-6000
-4000
-2000
0
2000
4000
6000

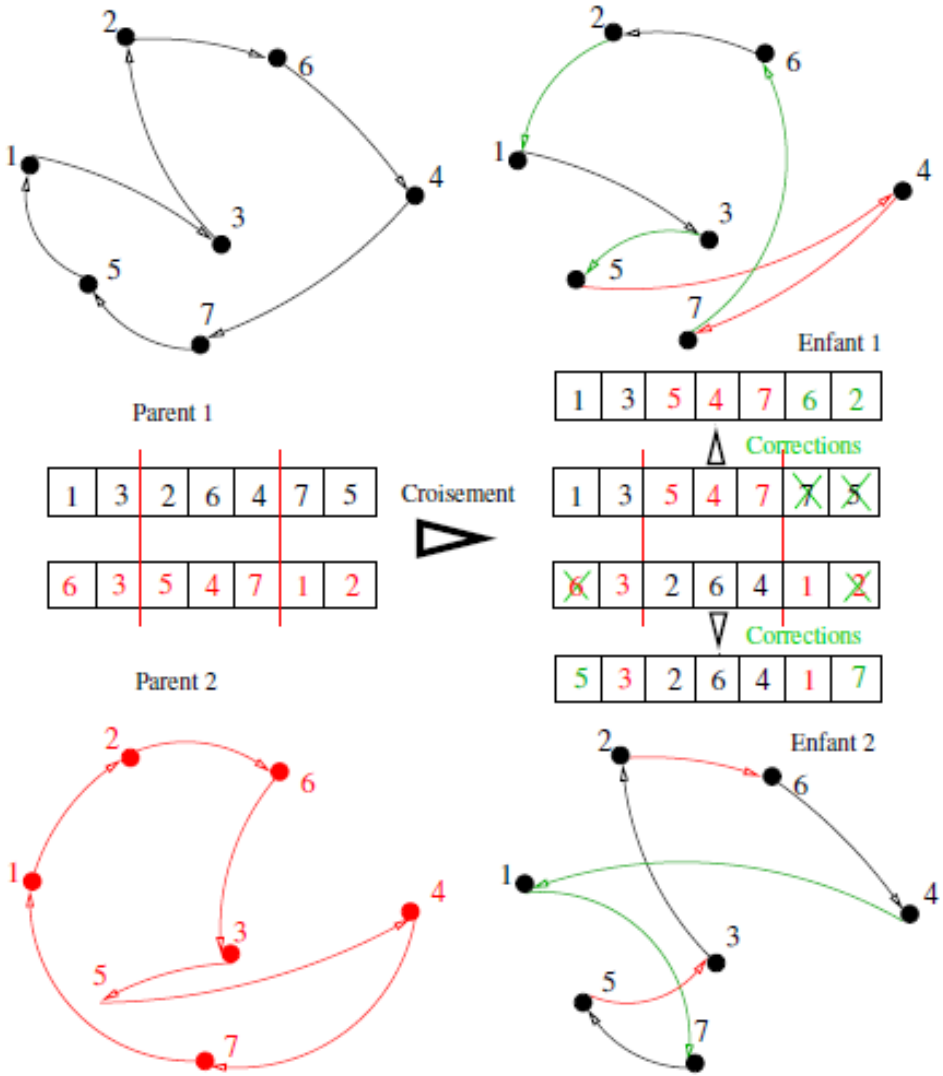

0: 0: 6 - Prevision 20 mn - Opti toutes les 4 mn EN COURS - Incertitude 5%
0.383436
4000
2000
0
-2000
-4000
-4000
-2000
0
2000
4000
6000
8000
10000


0: 0: 4 - Prevision 20 mn - Opti toutes les 3 mn EN COURS - Incertitude 3%
0.738450
8000
6000
4000
2000
0
-2000
-4000
-4000
-2000
0
2000
4000
6000
8000
10000

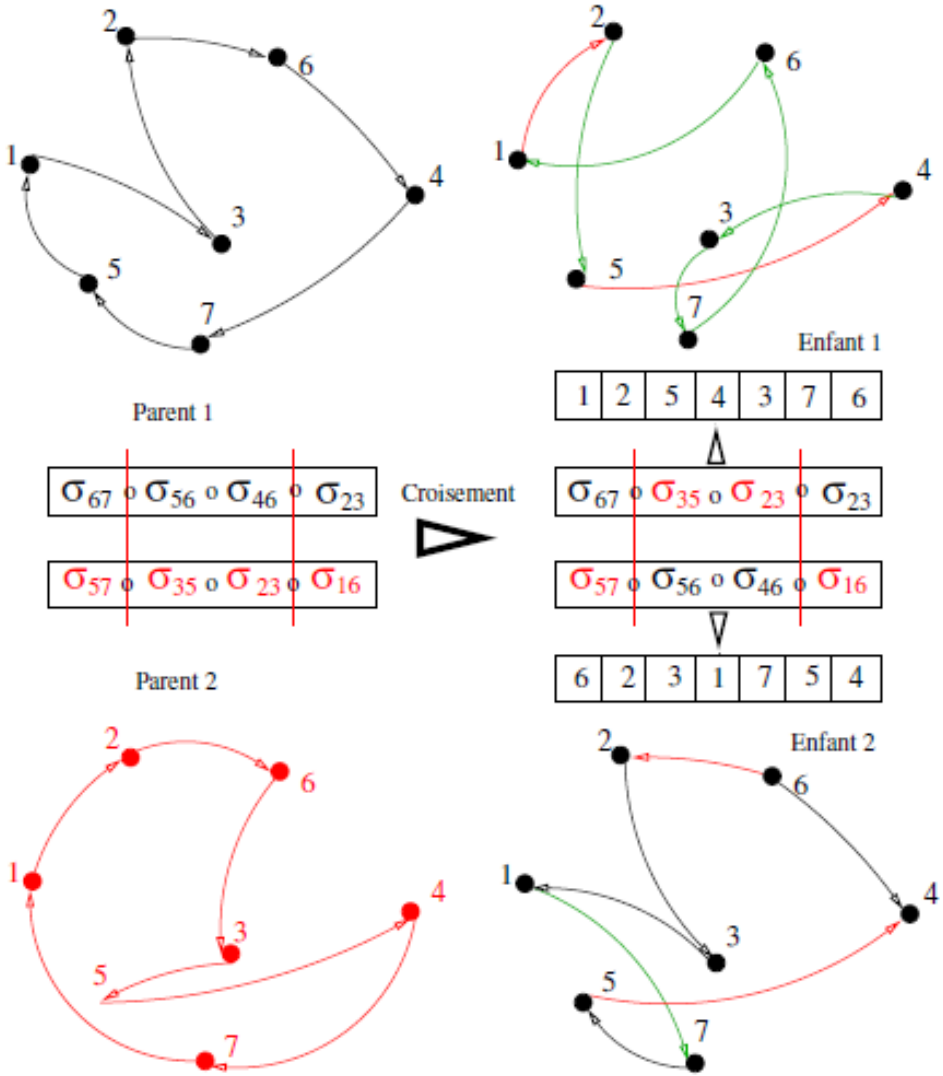
Le voyageur de commerce



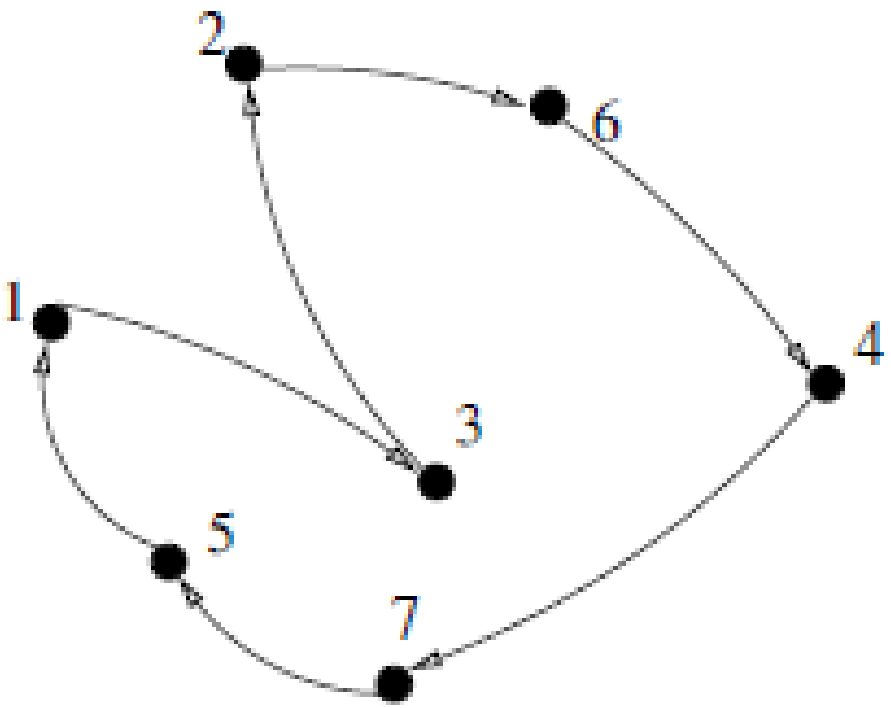
Le voyageur de commerce: croisement



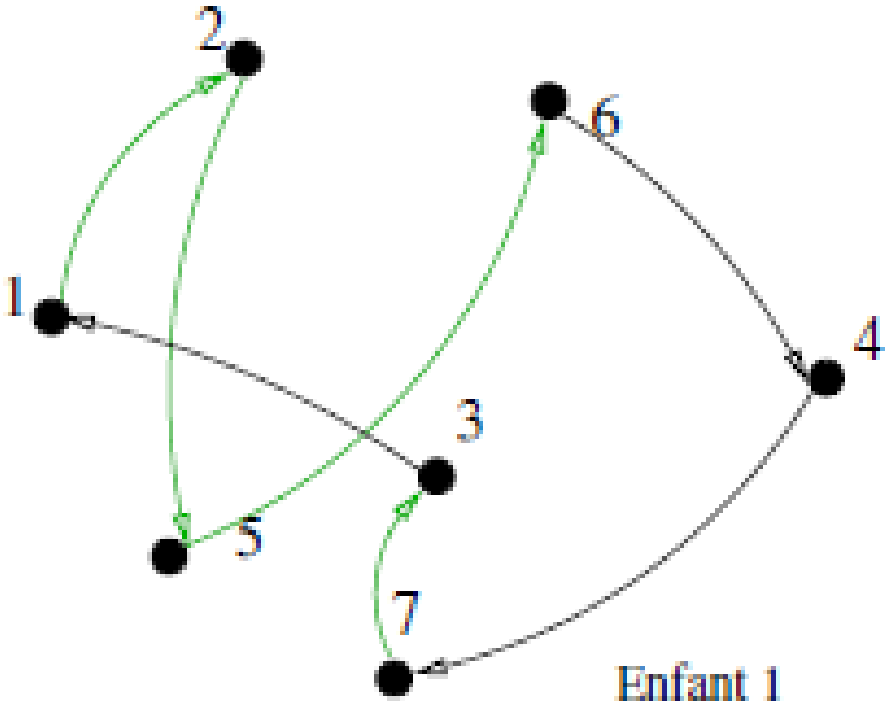
Nouveau croisement



Voyageur de commerce: mutation



Parent 1



Enfant 1

1	2	5	4	3	7	6

Mutation

$\sigma_{67} \circ \sigma_{56} \circ \sigma_{46} \circ \sigma_{23}$ → $\sigma_{67} \circ \sigma_{56} \circ \sigma_{46} \circ \sigma_{35}$

B&B et intervalles (méthodes globales déterministes)

- Soit la fonction:

  $f(x,y) = 333.75\ y^6 + x^2\ (11\ x^2y^2 - y^6 - 121\ y^4 - 2) + 5.5\ y^8 + x\ /\ (2y)$

- Si l'on calcule f(77617,33096), on obtient environ 1.172603.
- La valeur correcte est -0.827396.
- La programmation par intervalle a été utilisée au départ pour éviter les erreurs d'arrondi.

Opérations élémentaires

- Soit deux intervalles X=[a,b] et Y=[c,d]
- X+Y=[a+c,b+d] et X-Y=[a-d,b-c]
- X*Y=
  - [ac,bd] si a>0 et c>0
  - [bc,bd] si a>0 et c<0<d
  - [bc,ad] si a>0 et d<0
  - [ad,bc] si a<0<b et c>0
  - [bd,ad] si a<0<b et d<0
  - [ad,bc] si b<0 et c>0
  - [ad,ac] si b<0 et c<0<d
  - [bd,ac] si b<0 et d<0
  - [min(bc,ad),max(ac,bd)] si a<0<b et c<0<d

La division

- Il faut étendre R en lui ajoutant $+\infty/-\infty$
- X/Y=
  - $[b/c,+\infty]$ si $b<0$ et $d=0$
  - $[-\infty,b/d]$ et $[b/c,+\infty]$ si $b<0$ et $c<0<d$
  - $[-\infty,+\infty]$ si $a<0<b$
  - $[-\infty,a/c]$ si $a>0$ et $d=0$
  - $[-\infty,a/c]$ et $[a/d,+\infty]$ si $a>0$ et $c<0<d$
  - $[a/d,+\infty]$ si $a>0$ et $c=0$

Autres opérations

- Toutes les opérations peuvent être étendues à la programmation par intervalle.
- Pour les fonctions monotones:
  - F([a,b])=[f(a),f(b)] si f croissante
  - F([a,b])=[f(b),f(a)] si f décroissante
  - Exemple: Exp([a,b])=[e^a,e^b]
- Les compositions de fonction se traitent en composant les fonctions d'extension sur les intervalles.

Le problème de dépendance

- Soit X=[a,b], X-X = [a-b,b-a]<>[0,0]!
- De la même façon (X-1)(X+1) <> X^2-1
- ([0,2]-1)([0,2]+1)=[-1,1]*[1,3]=[-3,3]
- $[0,2]^2$-1=[0,4]-1=[-1,3]
- L'associativité est conservée:
  - A+(B+C)=(A+B)+C
  - A(BC)=(AB)C
- Distributivité perdue: A(B+C)<>AB+AC

Branch and bound

- Nom générique recouvrant des techniques de recherche divisant l'espace de recherche et éliminant des parties de l'espace en fonction de bornes.
- Dans le cas présent, c'est la division de l'espace en sous intervalle qui crée les branches, et le calcul d'un estimateur par intervalle qui crée les bornes.

Minimisation

- Initialiser: L<-{[a,b]} et e=estimateur de f sur [a,b]
  - Extraire I=[c,d] de L. Si e<c, réitérer. Si l'intervalle vérifie le test de fin, réitérer. Si L est vide: fin.
  - Construire I_1=[c,(c+d)/2] et I_2=[(c+d)/2,d].
  - Calculer F(I_1)=[x_1,y_1], F(I_2)=[x_2,y_2], e_1 et e_2.
  - Prendre e=min(e,e1,e2)
  - Si x_1<e alors insérer I_1 dans L
  - Si x_2<e alors insérer I_2 dans L
  - Réitérer.

Calcul d'un estimateur

- Soit X=[a,b]. Plusieurs méthodes:
  - Moyen le plus simple: e=f((a+b)/2)
  - Technique de sampling: prendre n points régulièrement répartis dans X
  - Technique stochastique: tirer au hasard n points dans X
  - Calculer f'(x) et F'(X) et regarder si le signe de la dérivée est constant sur X, auquel cas f est monotone

Insertion d'un élément

- Plusieurs possibilités pour insérer:
  - First In First Out
  - En fonction de la taille de l'intervalle (le plus grand en tête)
  - En fonction de l'estimateur (le plus petit en tête)
  - En fonction de la borne inférieure (la plus petite en premier)
  - etc...

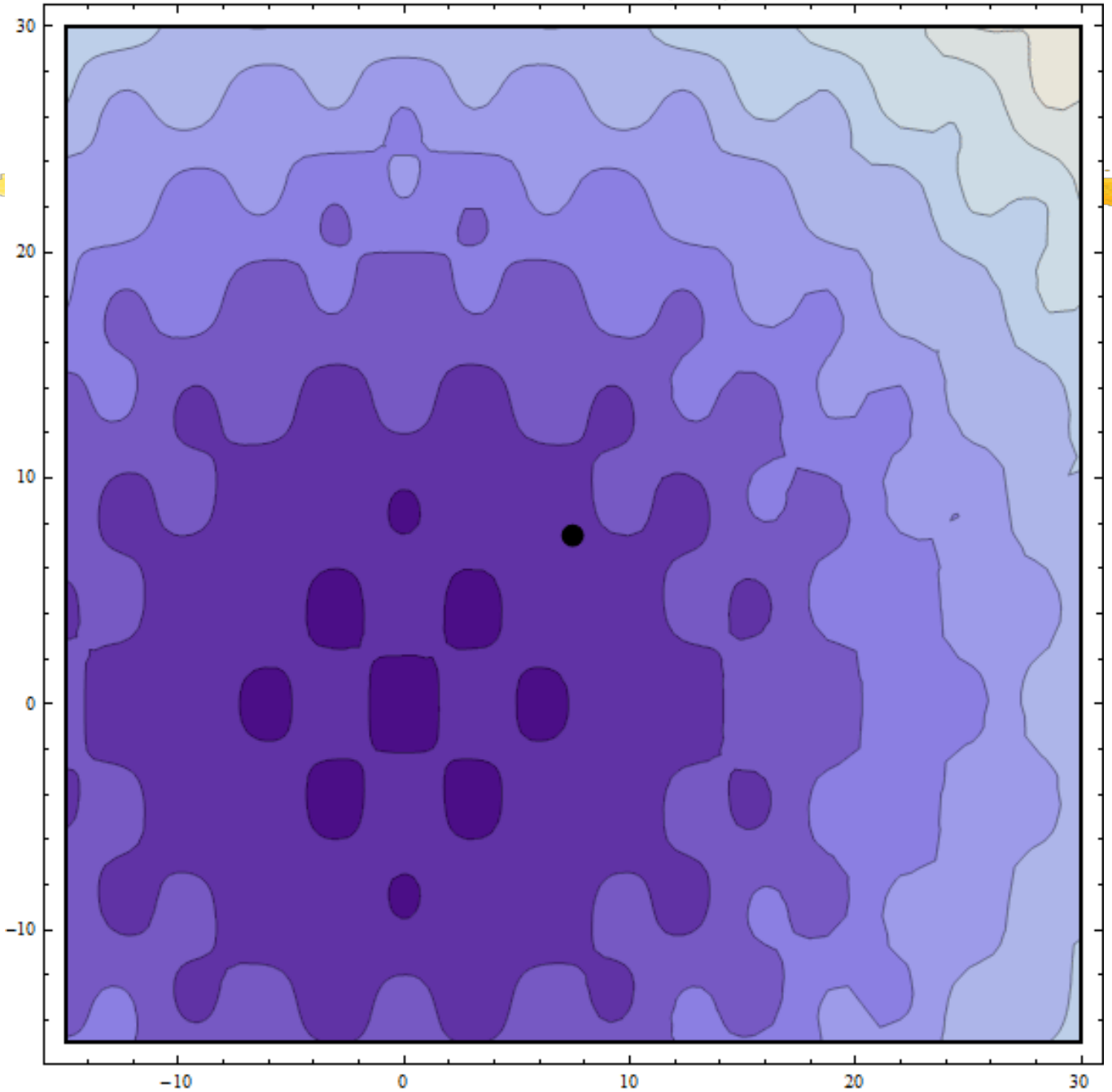
Test de fin

- Plusieurs possibilités:
  - La taille de l'intervalle est inférieure à une quantité fixée.
  - La taille de l'image de l'intervalle est inférieure à une quantité fixée...
  - Etc...

Extension à plus d'une dimension

- Pour une fonction à plusieurs variables, la dichotomie se fait variable par variable.
- On coupera systématiquement en deux l'intervalle le plus grand parmi ceux décrivant toutes les variables.
- On fait évoluer en conséquence le test de fin.

$$\frac{1}{100}\left(x^2+y^2\right)-\cos(x)\cos\left(\frac{y}{\sqrt{2}}\right)$$



Quand l'utiliser

- Méthode extrêmement efficace lorsqu'il y a peu de variables.
- Le temps de calcul croit comme 2^N ou N désigne le nombre de variables.