Utiliser des Support Vector Machines pour apprendre un noyau de viabilité

G. Deffuant S. Martin L. Chapel

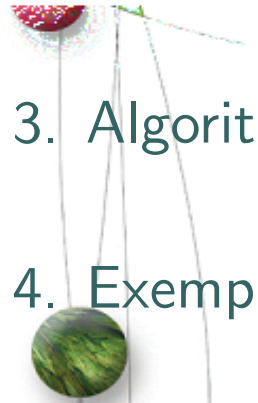
Laboratoire d'Ingénierie des Systèmes Complexes (LISC)
Cemagref

Majecstic, 16 - 18 novembre 2005

Introduction

- Dans de nombreuses applications en écologie ou économie, on veut contrôler un système afin de le maintenir dans un ensemble donné
- Pour cela, on recherche le noyau de viabilité du système
- Mais
  - la résolution de ce problème est difficile pour des espaces de grande dimension
  - la solution est difficile à manipuler
- Nous proposons un algorithme spécifique qui résout ces problèmes

Sommaire

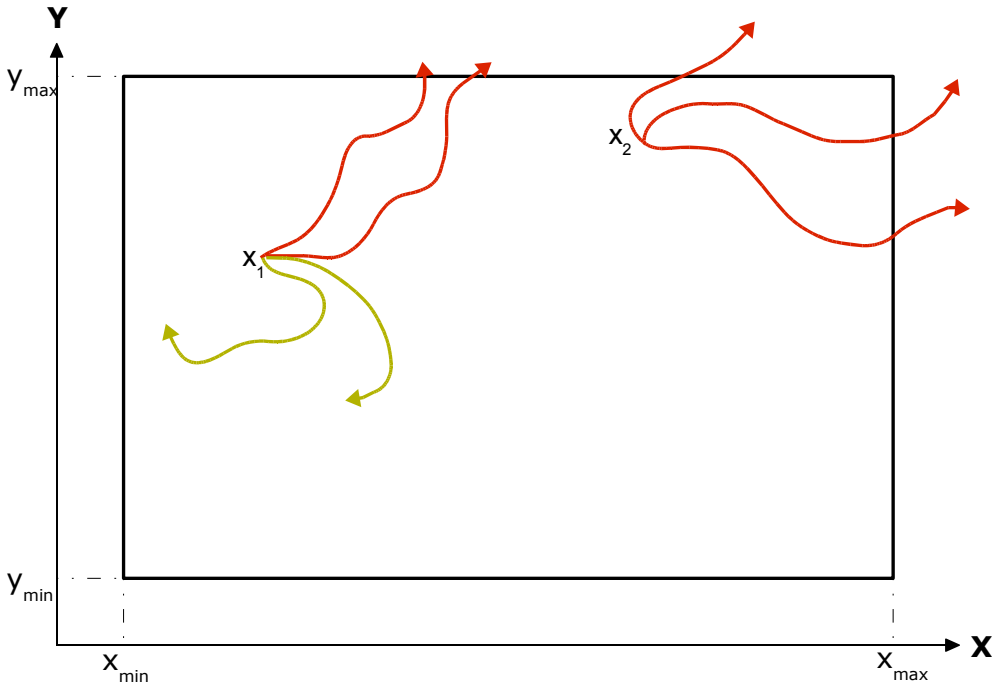


Sommaire

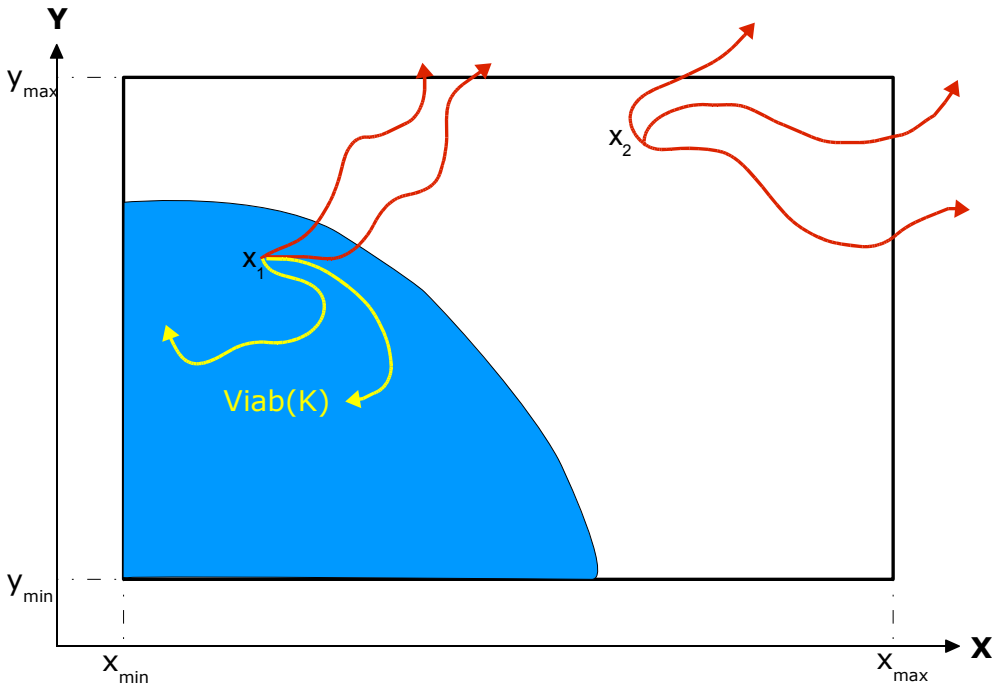
- But : contrôler un système dynamique afin qu'il survive dans un ensemble d'états admissibles, ou espace des contraintes
- Système dynamique

$$\begin{cases} x'(t) = \varphi(x(t), u(t)), \text{ pour tout t} \geq 0 \\ u(t) \quad U(x(t)) \quad \mathbb{R}^q \end{cases} \tag{1}$$

- État viable : il existe au moins une évolution qui permet de rester dans l'ensemble des contraintes de viabilité



- Noyau de viabilité : ensemble de tous les états viables



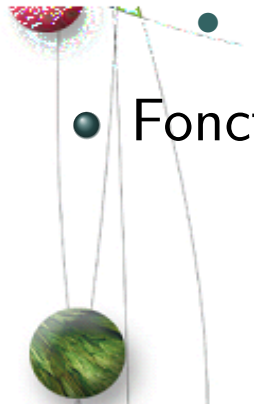
- Il existe un algorithme spécifique, basé sur la discrétisation de l'espace

P. Saint-Pierre
Approximation of viability kernel.
Applied Mathematics & Optimisation, 29:187-209, 1994.

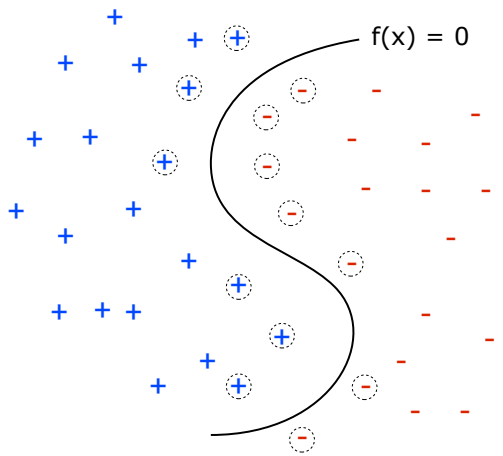
- En général, il n'y a pas de définition explicite du noyau
- Cet algorithme très rapide mais la taille de la grille croît avec la dimension du problème
- Le noyau est alors défini comme un ensemble de points

Sommaire

Support Vector Machines



- Construction d'un hyperplan séparateur dans un espace déployé
- $f(x) = \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b$ avec
  - $\alpha_i > 0$ vecteurs de supports
  - $K(x_i, x) = \exp\left(-\frac{\|x_i - x\|^2}{2\sigma^2}\right)$
- Fonction SVM : fonction telle que $f(x) = 0$

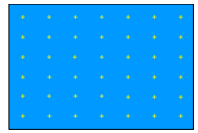
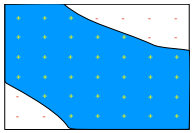
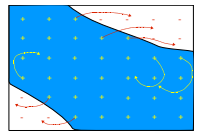
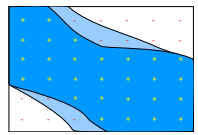


Sommaire

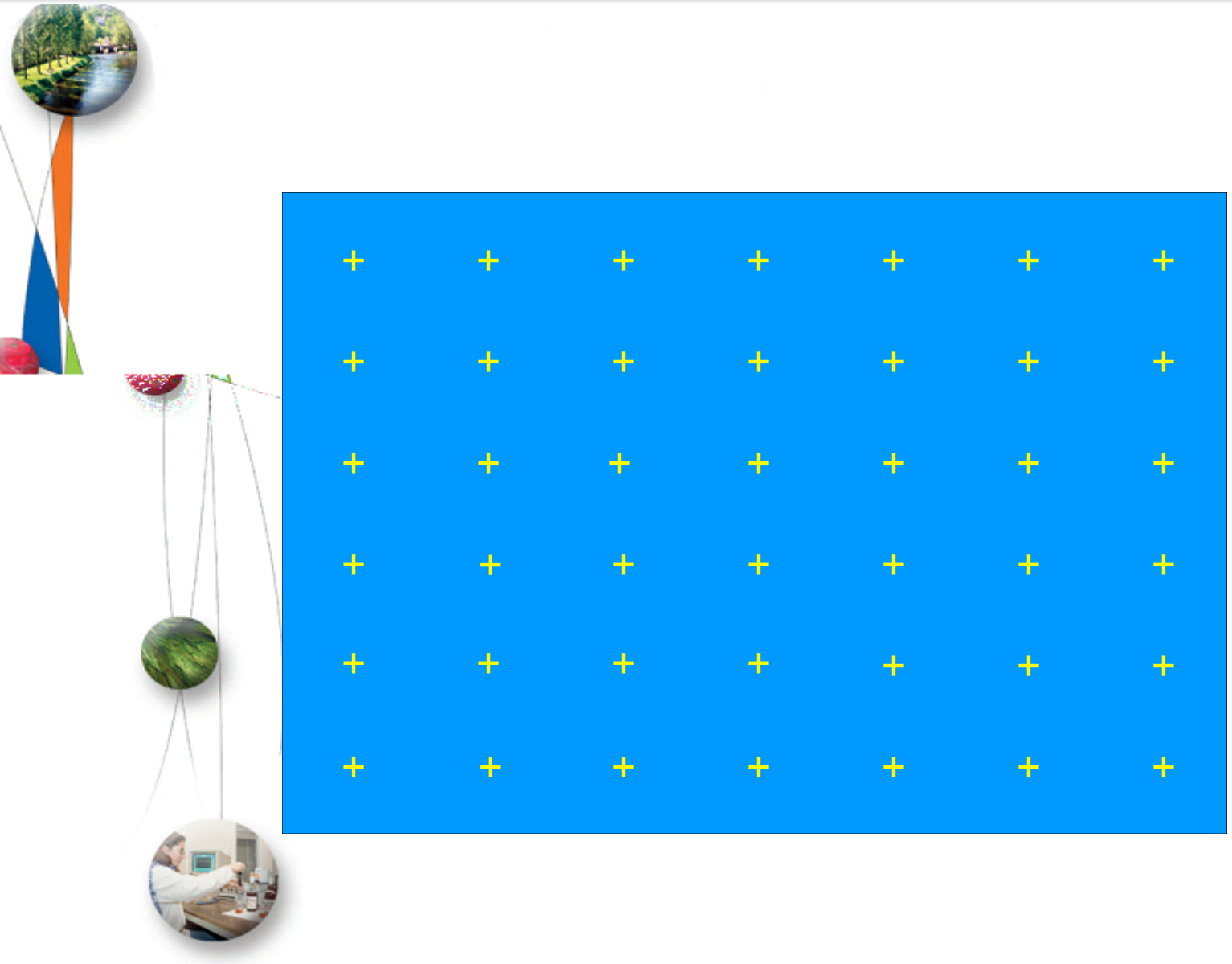
Algorithme proposé

- Utilisation des SVMs pour approximer un noyau de viabilité
- Permet d'avoir une expression analytique du noyau
- Avantages
  - permet l'utilisation d'une méthode d'optimisation (descente de gradient) pour trouver le meilleur contrôle
  - permet de travailler avec plusieurs pas de temps à la fois
  - la fonction SVM peut servir de contrôleur
  - l'expression dépend d'un nombre de points réduit → moins de points en mémoire ?
- Un algorithme spécifique a été développé et un théorème montre, sous certaines conditions, la convergence du noyau estimé vers le vrai noyau de viabilité

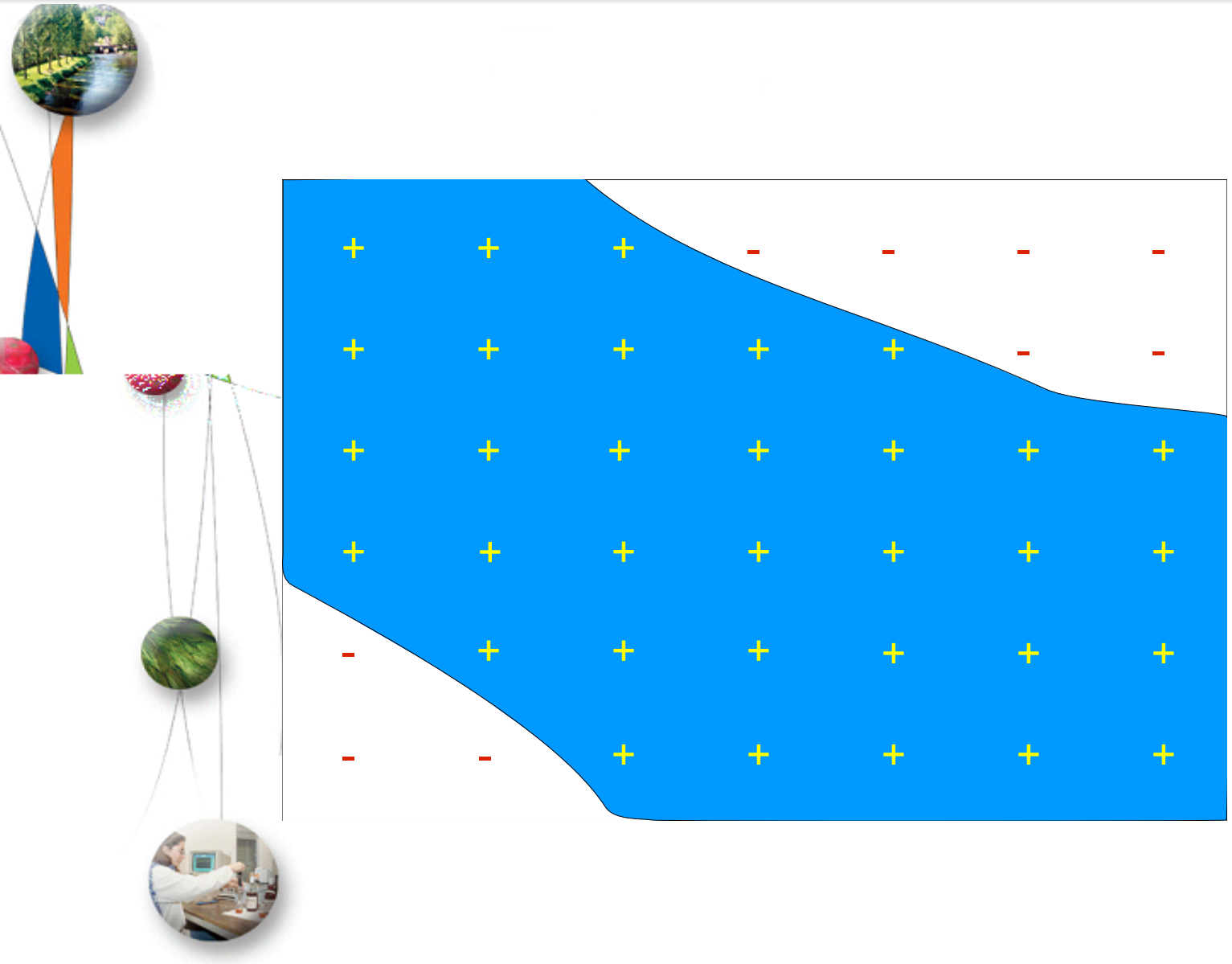
Algorithme proposé

- Initialisation : discrétisation de l'espace des états 
- Itération 1 : optimisation pour trouver le contrôle dans K
- Itération $n+1$ 
  - optimisation pour trouver le meilleur contrôle 
  - mise à jour des labels à partir de SVM_n
  - calcul de SVM_{n+1} 
- Jusqu'à ce qu'il n'y ait plus de modification de label

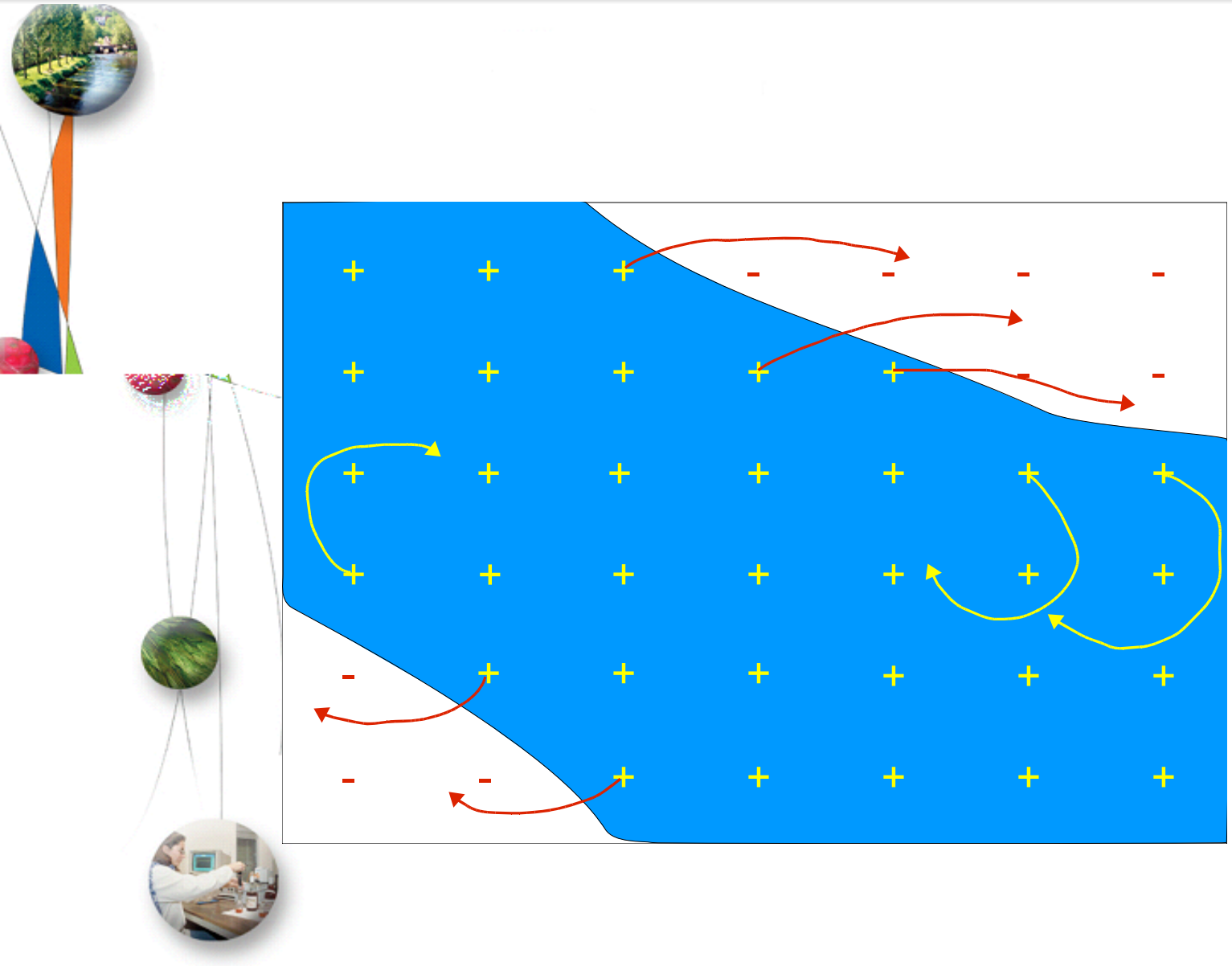
Algorithme proposé



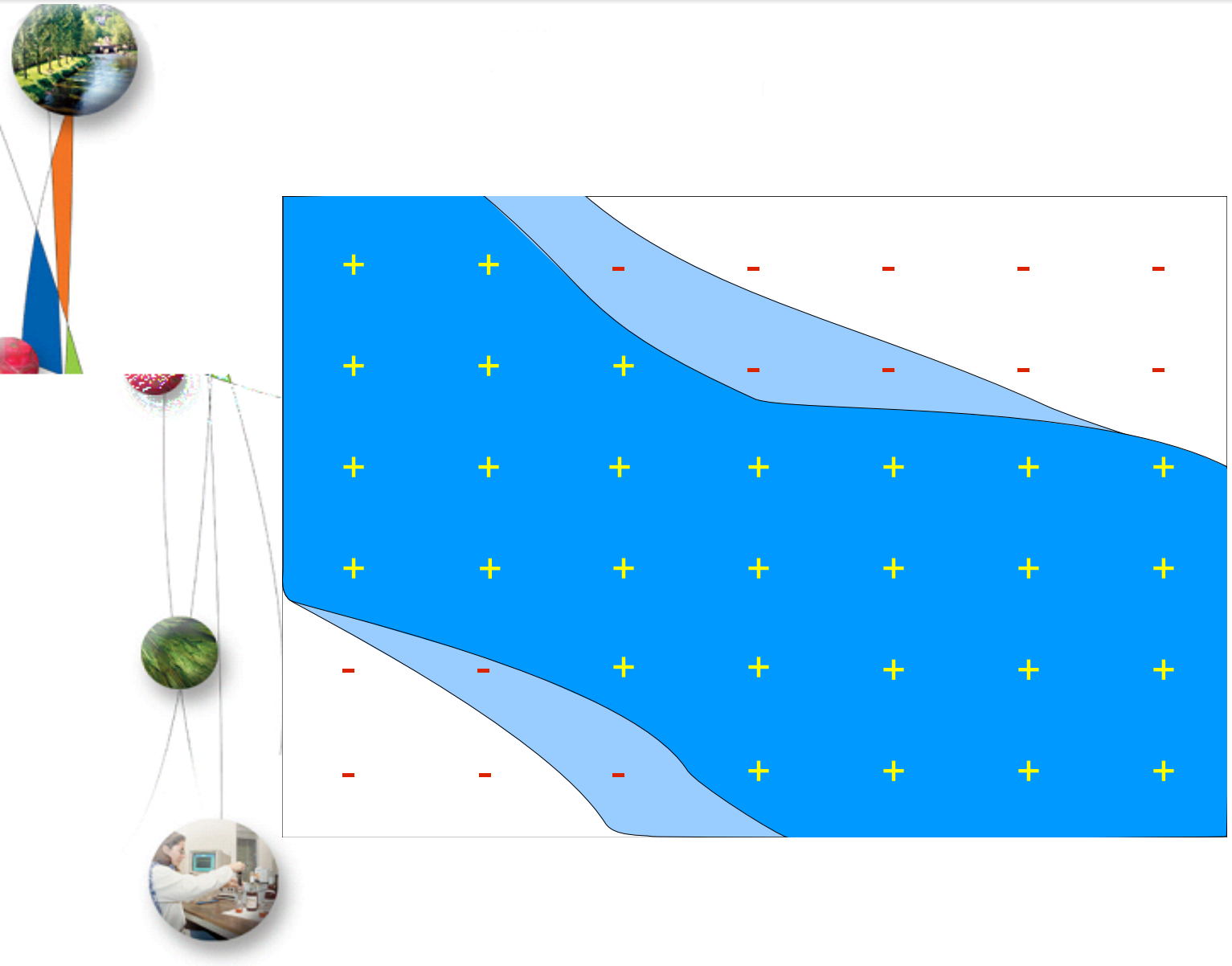
Algorithme proposé



Algorithme proposé



Algorithme proposé



Sommaire

Exemple d'application

- Modèle simplifié de la croissance d'une population dans un espace limité
- Système dynamique

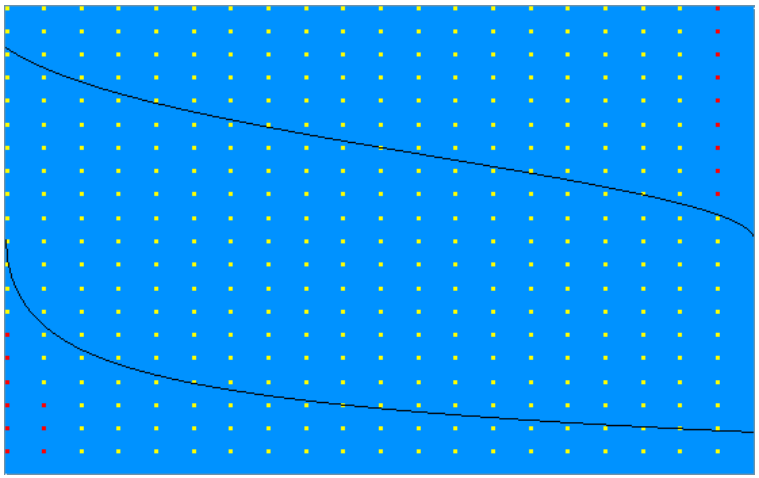
$$\begin{cases} x_1(t+dt) = x_1(t) + x_1(t)y(t)dt \\ x_2(t+dt) = x_2(t) \\ \dots \\ x_{n-1}(t+dt) = x_{n-1}(t) \\ y(t+dt) = y(t) + u(t)dt \end{cases} \quad (2)$$

- Sous contraintes
  - x_i $[a, b]$
  - y $[d, e]$
  - u $[-c, c]$

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

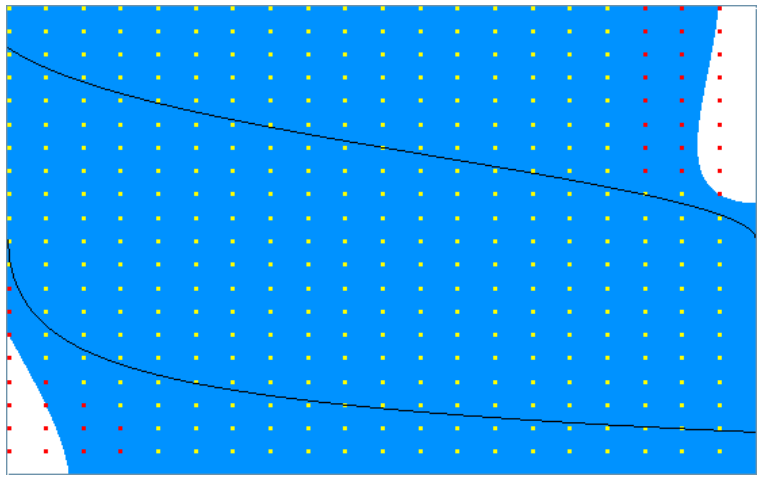


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

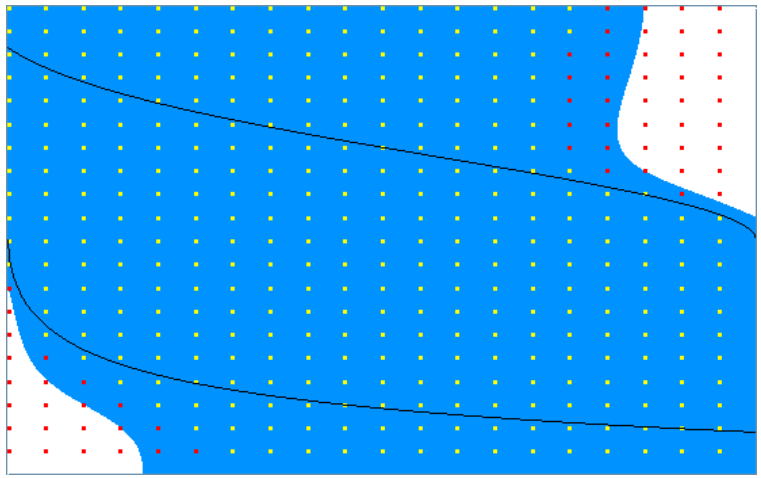


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

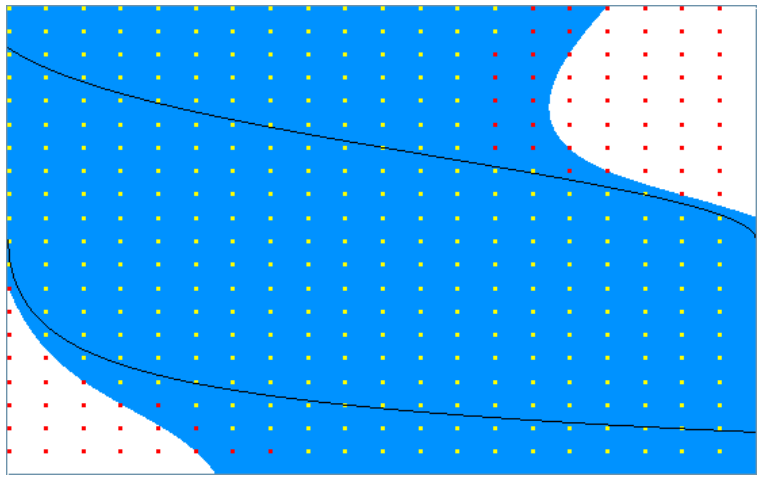


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

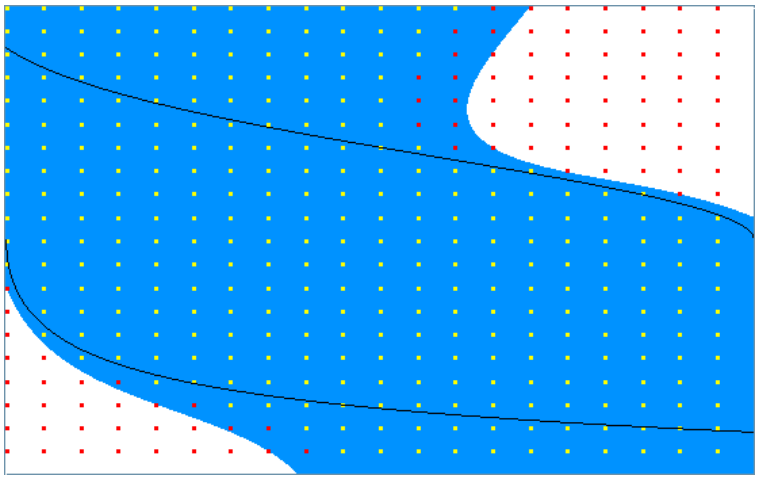


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

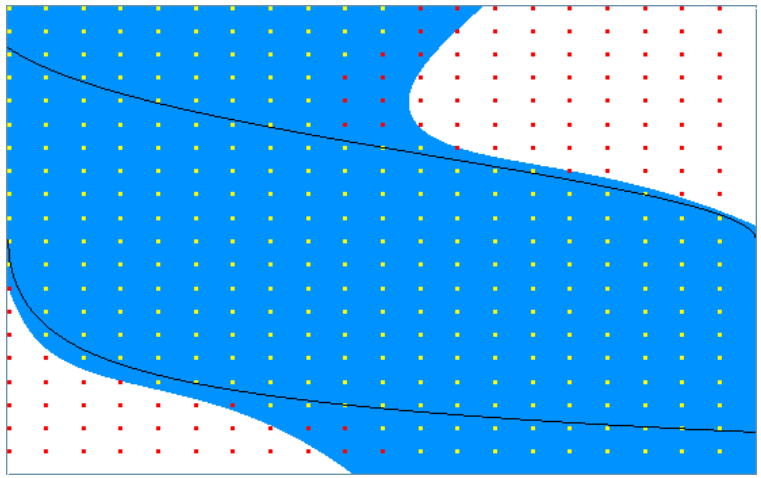


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

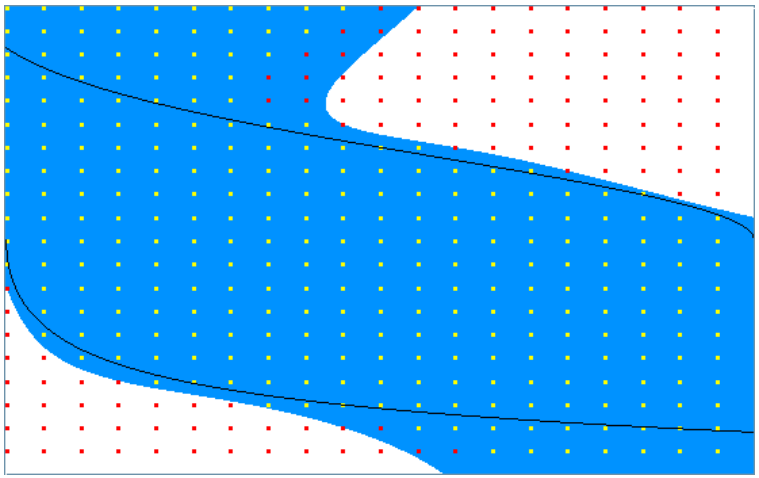


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

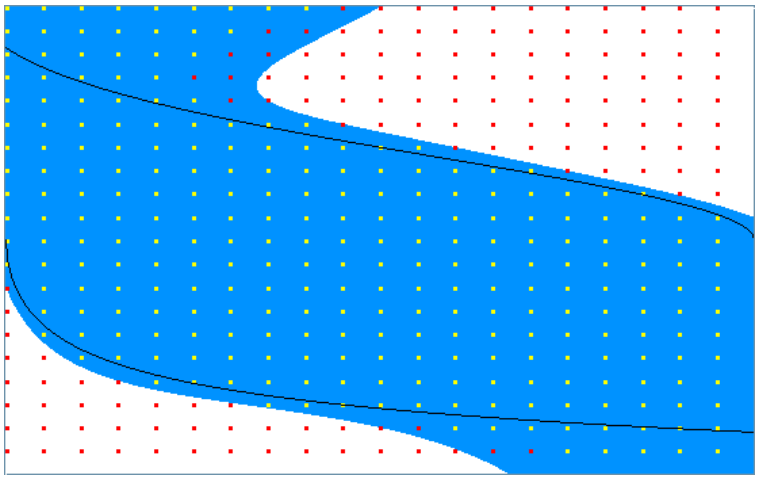


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

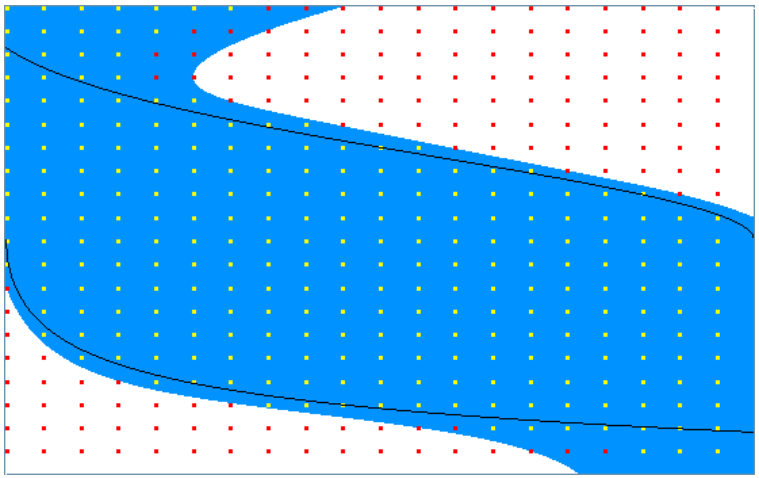


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

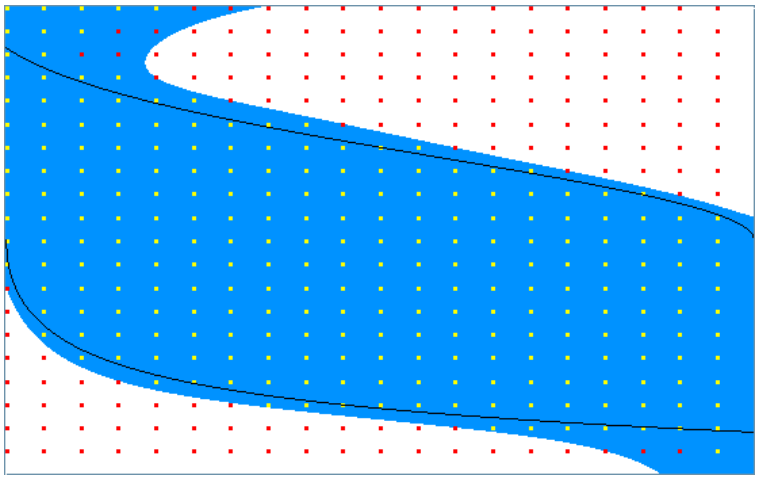


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

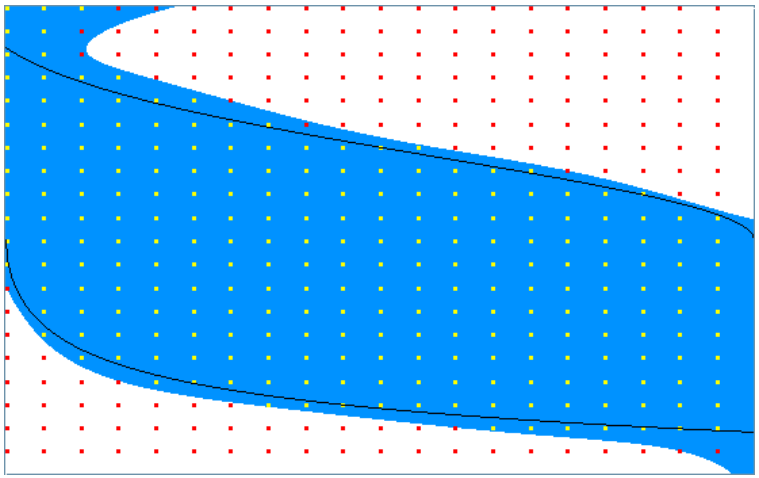


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

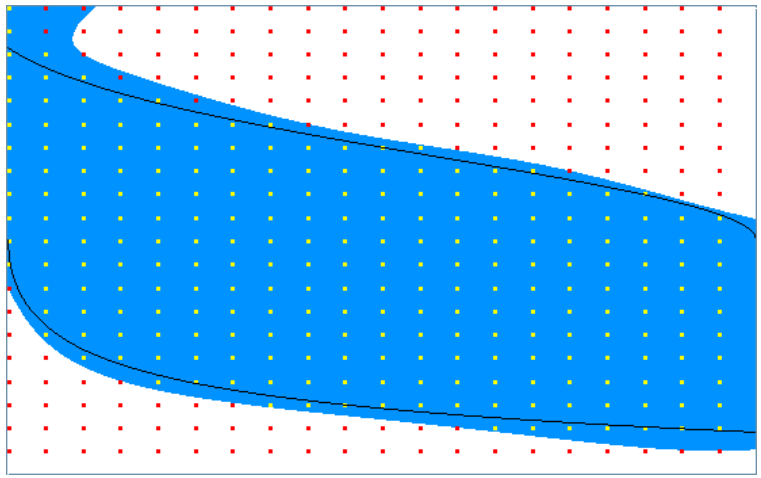


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

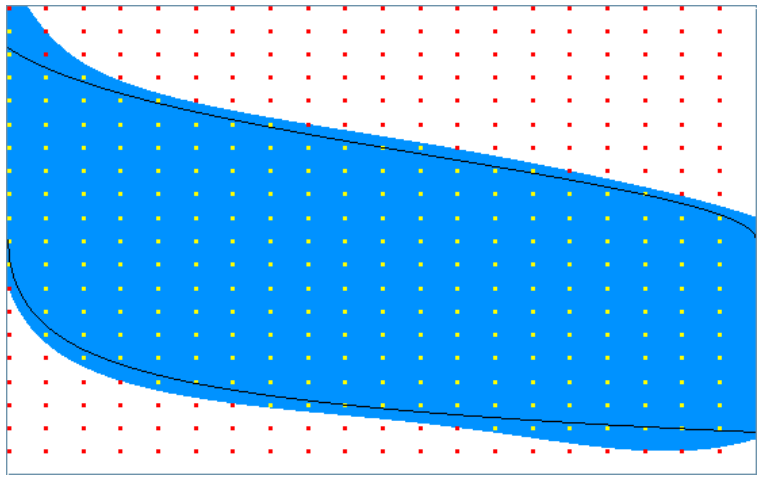


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

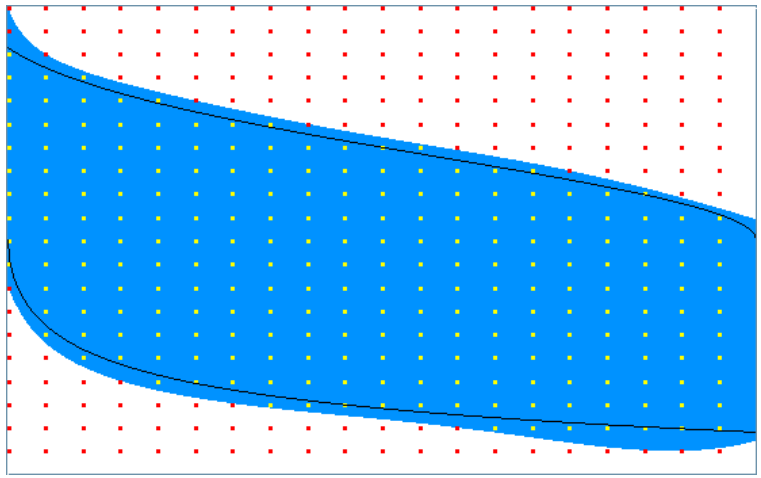


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas

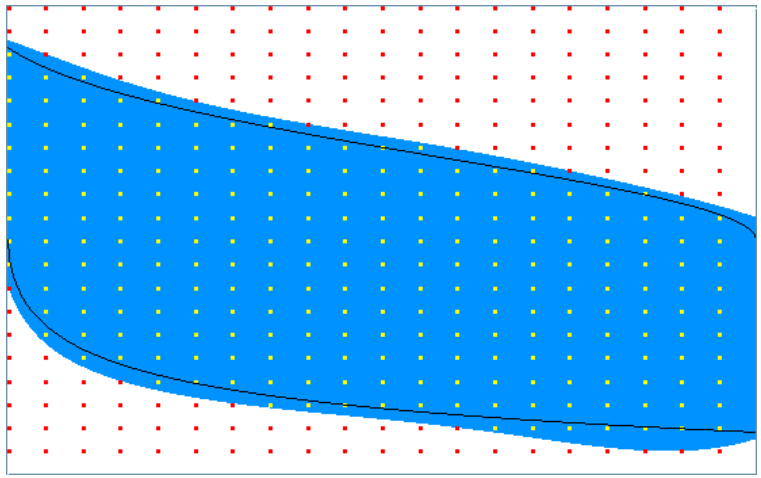


- 14 itérations, 16 SV

Exemple d'application

Déroulement de l'algorithme

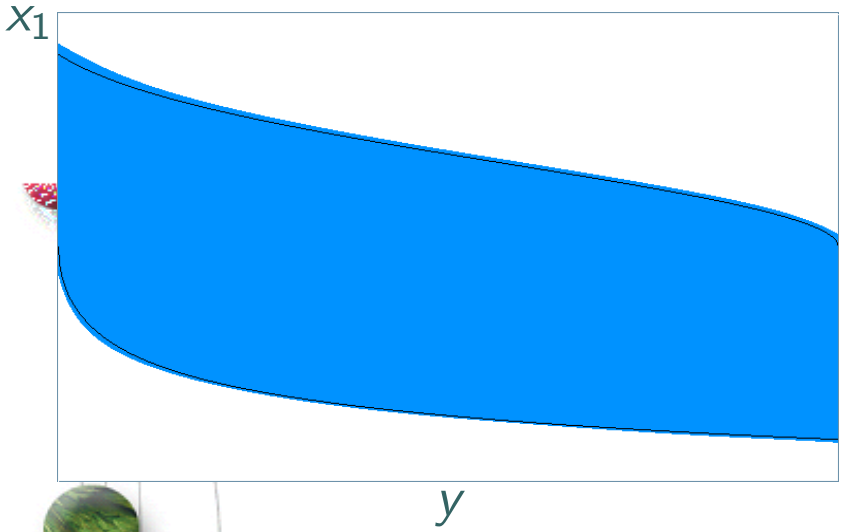
- Problème en 2 dimensions, grille de 21 points par dimension, 3 pas



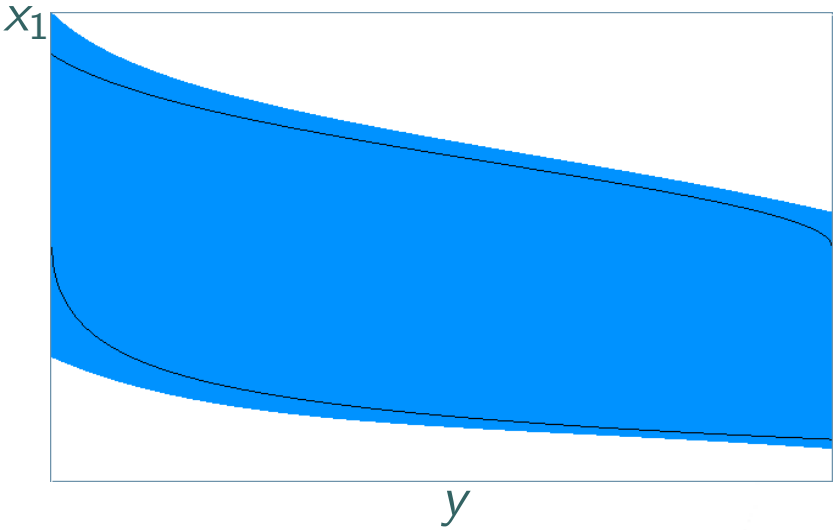
- 14 itérations, 16 SV

Exemple d'application

Exemple de résultats



Un exemple en dimension 2
(71 points par dimension)



Un exemple en dimension 5
(13 points par dimension
400 000 points au total)

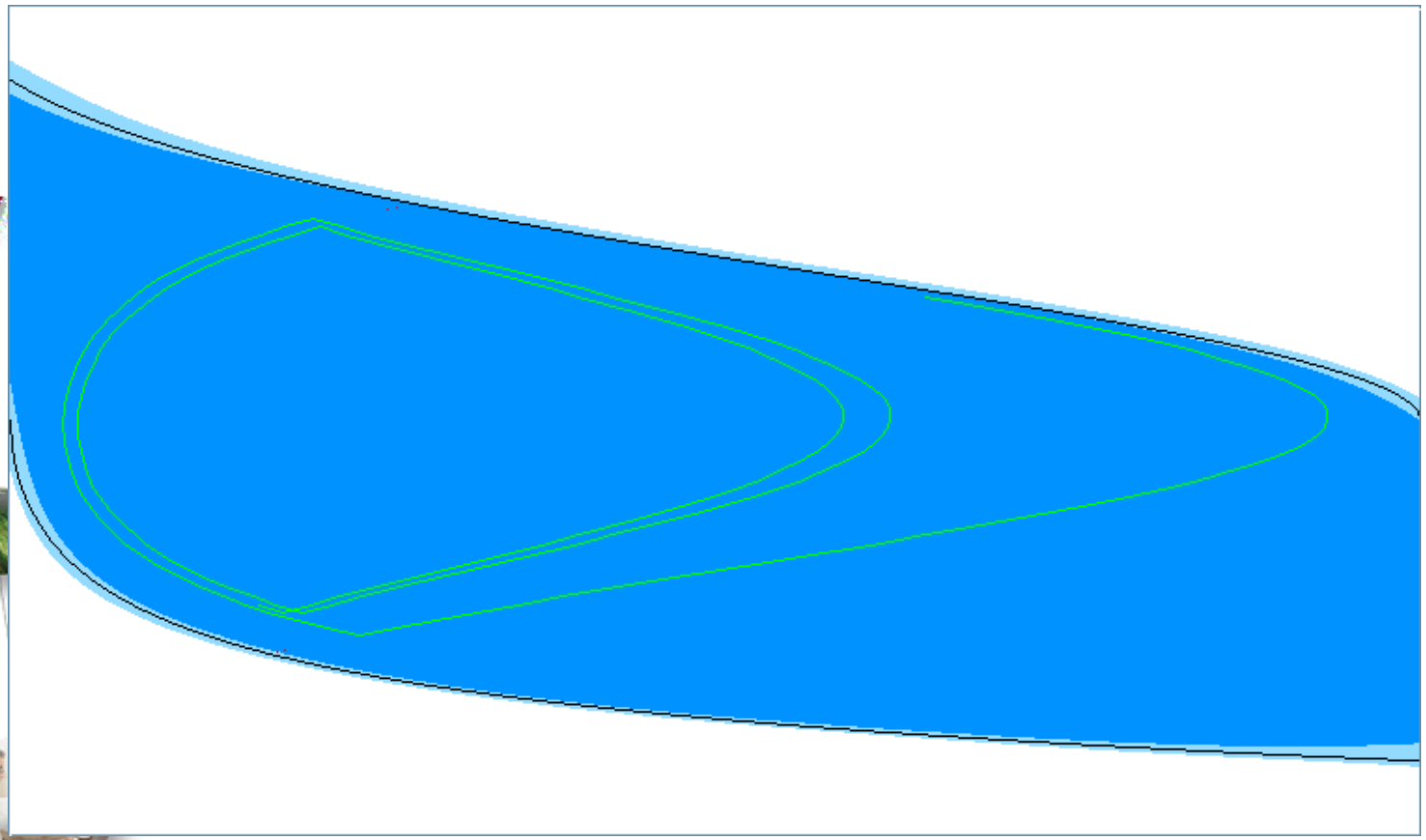
Exemple d'application

Contrôleur

- On part d'un sous-noyau, contenu dans le noyau estimé
- Algorithme
  - Si on ne traverse pas la frontière du sous-noyau à n pas, on conserve le contrôle précédent et on avance d'un pas (avec un contrôle constant)
  - Sinon, on optimise pour trouver le meilleur contrôle sur n pas et on avance d'un pas (avec un contrôle optimisé)
- Le contrôleur est plus ou moins prudent (vision à n pas)

Exemple d'application

Exemple de contrôleur (vision à 5 pas)



Sommaire

Conclusion

Utiliser des SVMs pour approcher un noyau de viabilité permet :

- d'obtenir une définition analytique du noyau
- d'utiliser des algorithmes d'optimisation pour calculer le meilleur contrôle à un moment donné
- d'améliorer la précision de l'approximation en optimisant sur plusieurs pas
- de définir des contrôleurs plus ou moins prudents

Perspectives

- Sélectionner les points les plus pertinents pour le calcul de la SVM
- Travailler avec une grille de résolution variable