Assessing the Resilience of Socio-Ecosystems: Coupling Viability Theory and Active Learning with kd-Trees. Application to Bilingual Societies*

Isabelle Alvarez(1,3)
(1) LIP6, UPMC
F-75005 Paris, France
isabelle.alvarez@lip6.fr

Ricardo de Aldama (2)
(2) ISC-PIF
F-75005 Paris, France
ricardo.de.aldama@iscpif.fr

Sophie Martin (3)
(3) Irstea, LISC
F-63172 Aubiere Cedex, France
sophie.martin@irstea.fr

Romain Reuillon (2)
(2) ISC-PIF
F-75005 Paris, France
romain.reuillon@iscpif.fr

Abstract

This paper proposes a new algorithm to compute the resilience of a social system or an ecosystem when it is defined in the framework of the mathematical viability theory. It is applied to the problem of language coexistence: Although bilingual societies do exist, many languages have disappeared and some seem endangered presently. Mathematical models of language competition generally conclude that one language will disappear, except when the relative prestige of the languages can be modified. The viability theory provides concepts and tools that are suitable to study the resilience, but with severe computational limits since it uses extensive search on regular grids. The method we propose considers the computation of the viability output sets as an active learning problem with the objective of restraining the number of calls to the model and information storage. We adapt a kd-tree algorithm to approximate the level sets of the resilience value. We prove that this algorithm converges to the output sets defined by the viability theory (viability kernel and capture basin). The resilience value we compute can then be used to propose a policy of action in risky situations such as migration flows.

1 Introduction

Assessing the resilience of an ecological or social system is becoming a challenge in the context of sustainable development (as stated in [Perrings, 2006]). Traditionally resilience measures the ability of a system to recover after a perturbation (see [Martin, 2004] for an analysis of operational definitions of resilience). In this context the mathematical theory of viability is an interesting framework since it studies the compatibility of dynamical systems and constraints [Aubin *et al.*, 2011]. This framework is used in sustainability studies in order to find control policies that keep the system in a given constraint set, such as the concept of tolerable windows [Bruckner *et al.*, 2003] for climate change studies. When the system evolves outside the desirable constraint set, viability theory can assess whether and how the system can be driven back to desirable states. In particular it can provide the returning time [Doyen and Saint-Pierre, 1997]. In this paper we quantify the resilience as the inverse value of this returning time, as it is done in [Martin, 2004] for ecological systems.

Viability analysis is based on the computation of the viability kernel and its capture basin. The viability kernel gathers all the states from which there exists a control function that allows the evolution to stay in the constraint set. Its capture basin gathers the states from which it is possible to reach the viability kernel in finite time. Unfortunately, the complexity of the computation task is exponential with space or time when using a discrete approximation on a grid [Saint-Pierre, 1994]. Therefore this method is limited to very low dimension (at most 3) or to linear models, so its applicability is very much impaired. Moreover, the complexity of the model in real application can limit the practical use of the method because of computation problems when running the model. Dimension 4 was exceptionally reached in a real food processing application with the help of grid computing (see [Sicard *et al.*, 2012], [Reuillon *et al.*, 2010]).

In the particular case of finite time horizon, [Bonneuil, 2006] proposes a method based on simulated annealing to generate trajectories near the boundary. This method works in higher dimensions but it only produces a set of trajectories so it cannot be used to compute the viability kernel nor resilience level sets. Following the same idea of focusing on the boundary in the general case, [Deffuant *et al.*, 2007] introduces classification functions in Saint-Pierre's viability algorithm in order to reduce the number of calls to the model and to represent the viability set in a more compact way than the traditional regular grid. But the Support Vector Machine (SVM) functions used as classification function in [Deffuant *et al.*, 2007], although very attractive, don't fulfill the conditions of the convergence theorem. So the output sets produced

*The research leading to these results has received funding from the E.C.'s FP7/2009-2013 under grant agreement DREAM n. 222654-2.

with the SVM method are not reliable. To overcome these limitations, we also consider the computation of the boundary at each step as an active learning problem, but we use kd-trees as it is done in [Rouquier *et al.*,] to limit the number of calls to the model. Using kd-trees as classification functions as well as oracle functions to store the successive approximations of the viability sets, it is possible to converge towards the true viability kernel or capture basin.

The paper is organized as follows: Section 2 presents the language competition problem and its translation in the viability framework. We define the resilience and show how it is related to the capture basin defined by the theory. Section 3 is devoted to the computation of the capture basin. It explains how it can be seen as an active learning problem, and it describes the kd-tree algorithm that is used in that purpose. It also gives the proof of convergence of the algorithm towards the output sets of the viability theory. Section 4 presents the result of the computation for the language competition problem and how the computed resilience value can be used to propose action policies that guarantee the language coexistence.

2 Resilience of bilingual societies

Bilingual societies exist, but history tells us that many languages have disappeared. Presently, many languages seem endangered. Some researchers have proposed mathematical models to study language competition (see for instance [Abrams and Strogatz, 2003]). These models are consistent with historical data from past or present endangered languages (such as Welsh, Scottish, Gaelic, Quechua). Models without bilingual population generally conclude that one language eventually becomes extinct. When a bilingual population exists, the survival of the language diversity depends on the relative prestige of languages [Bernard and Martin, 2012].

But the question of resilience of these societies is still pending. Real societies are subjected to migration: both immigration and emigration (due to economic, environmental, educational or safety reasons for instance) disturb the dynamics of the different language speakers and can jeopardize the diversity of a bilingual society. This is the reason why it is interesting to study the ability of a bilingual society to recover from such a perturbation. Here we propose to study this resilience in the viability framework.

2.1 Model of bilingual society

The model from [Bernard and Martin, 2012] describes a bilingual society considering three groups in the population: the monolingual speakers of language A, the monolingual speakers of language B, and the bilingual speakers AB. The model is described by σ_A the proportion of speakers of A, σ_B the proportion of speakers of B, and the proportion of bilinguals σ_{AB} with $\sigma_{AB} = 1 - \sigma_A - \sigma_B$. The rate at which speakers of one language switch to become speakers of the second language depends on the attractiveness of this second language, but transitions between groups concern A or $B \rightarrow AB$ and conversely. Since it is unlikely for one speaker to switch language from scratch, we have $P_{A\rightarrow B} = P_{B\rightarrow A} = 0$. We then have:

$\frac{d\sigma_A}{dt} = \sigma_{AB}P_{AB\rightarrow A} - \sigma_A P_{A\rightarrow AB}$, (with analogous equations for $\frac{d\sigma_B}{dt}$ and $\frac{d\sigma_{AB}}{dt}$).

Concerning transitions, monolinguals are sensitive to the size of the monolingual group of the other language when bilinguals are sensitive to the whole group of speakers of the other language. The proportionality factor is the prestige of one language compared to the other (we note s the prestige of language A, so $1 - s$ is the prestige of language B). A parameter a models how the attractiveness of a language scales with the proportion of speakers. So we have:

$P_{AB\rightarrow A} = (1 - \sigma_B)^a s$ and $P_{A\rightarrow AB} = \sigma_B^a(1 - s)$. The value of the prestige, s, can evolve with public action: education, advertising, incentive policy, arranged employment, positive discrimination, and so on. Since these kinds of policy take time to give results, $\frac{ds}{dt}$ is considered here as a control on the dynamics, bounded in some interval $U = [-\bar{u}; \bar{u}]$ with $\bar{u} > 0$. In this framework the model of a bilingual society is defined by the following equations:

$$\begin{cases} \frac{d\sigma_A}{dt} &= (1 - \sigma_A - \sigma_B)(1 - \sigma_B)^a s - \sigma_A \sigma_B^a (1 - s) \\ \frac{d\sigma_B}{dt} &= (1 - \sigma_A - \sigma_B)(1 - \sigma_A)^a (1 - s) - \sigma_B \sigma_A^a s \\ \frac{ds}{dt} &= u \in U \end{cases} \tag{1}$$

If s is constant in $]0; 1[$, the dynamics has three equilibria: $(0, 1)$, $(1, 0)$ which are stable, and an unstable one. Consequently, one language is doomed to become extinct. Therefore it is necessary to apply control policy on s to insure the coexistence.

2.2 Set of desirable states and viability domain

In order to ensure the sustainability of the language diversity, it is necessary to define what is considered as desirable. For instance, an easy way to define language diversity would be to consider that the proportion of monolingual speakers of each language has to be above a threshold $\underline{\sigma}$. The set of desirable set in $E = [0, 1]^3$ is then $K = \{(\sigma_A, \sigma_B, s)\}$ such that:

$$\begin{cases} 0 < \underline{\sigma} \leq & \sigma_A & \leq 1 \\ 0 < \underline{\sigma} \leq & \sigma_B & \leq 1 \\ 0 \leq & s & \leq 1. \end{cases} \tag{2}$$

We suppose now that this threshold is set to 0.2. We consider that the boundary of the control variation $\bar{u}$ is set to 0.1. Parameter a set to 1.31 according to the literature. (It is calibrated in [Abrams and Strogatz, 2003] from historical data).

In this framework the viability theory applies. It can be shown that there exists a subset of K which is a viability domain D: from each state $x = (\sigma_A, \sigma_B, s)$ in D there exists a control function $u(t)$ that allows the evolution of the dynamics 1 to stay inside D. Figure 1 shows the shape of D.

2.3 Capture basin and resilience value

Outside the viability domain D, the bilingual society may not be viable any longer. In particular, the proportion of one of the languages may fall under the threshold $\underline{\sigma} = 0.2$, which is not considered as a satisfying future. [1]

[1] Since there exists a viability domain in K, the viability theory states that there must be a viability kernel including D, but actually it

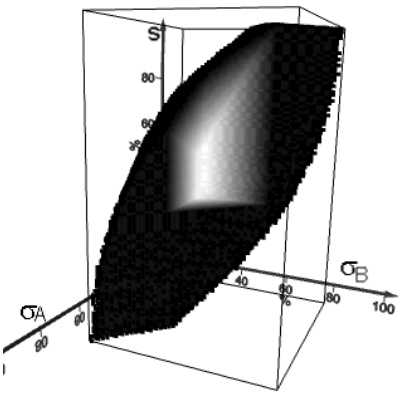

Figure 1: Viability domain D for the problem of language coexistence. The bounding box is the set of desirable states. Evolution starting in D can stay within forever.

We consider that without additional information, the viability domain D gathers the secure states, since from any starting point in D, we are sure that the evolution can remain in the set of desirable states K (that is, each language is spoken by at least 20% of the population) with the appropriate action policy.

In case of perturbation, the state of the bilingual society can jump from D to a state outside the viability domain. Here we consider that perturbations can affect each variable: Migrations change the value of the proportion of speakers of each language. But the prestige of one language may also be disturbed, for example by criminal cases or impressive achievement concerning one group. In a sustainability viewpoint, the issue is to know if it is possible to reach a secure position in the viability domain, starting from the disturbed state. The set of states from which there exists a control policy that drives the system back to D is called the capture basin of D in the viability theory.

$$Capt(D) = \{x \mid \exists u(.),\ \exists T \mid x(T) \in D\}$$

At each time horizon T, it is possible to consider the corresponding capture basin $Capt(D,T) = \{x \mid \exists u(.),\ x(T) \in D\}$. Its boundary $\partial Capt(D,T)$ is the level set of the returning time T to D. In particular we have by definition: $Capt(D,0) = D$.

Definition 1 (Resilience of a state) *The resilience of a state x outside the viability set D is $r(x) = 0$ if $x \notin Capt(D)$. Otherwise, it is $r(x) = \frac{1}{T}$ with $T = min\{t|\ x \in Capt(D,t)\}$.*

Definition 2 *The resilience of a bilingual society to a given perturbation set P is: $min\{r(x+y)|\ x \in D,\ y \in P\}$.*

When the capture basin is not greater than the viability domain, then the resilience of the bilingual society is zero: In case of a perturbation, the proportion of speakers of one language will never recover. If the capture basin is greater than the viability domain D, that means that if the state of the society is not as expected yet but still in the capture basin, it can return to the set of desirable states with an appropriate control policy. Computing the capture basin of D will allow to measure resilience and to find these control policies.

3 Computation of the resilience

3.1 Computation of the capture basin: active learning sampling with kd-tree

The computation of the capture basin of the viability domain D can be seen as a particular problem of viability [Aubin, 2001]. Actually, the capture basin can be defined as a projection of the viability kernel of an auxiliary viability problem with one additional dimension, the time dimension. So the same computational limits apply. The classical approach consists in sampling the state space along a grid, and store the coordinates of all viable points, but this is costly in terms of time and memory.

We consider the dynamical model as a black box F defined from $E = [0,1]^3$ to $E = [0,1]^3$. F is the discrete dynamical system corresponding to the model 1. Based on this black box function F, an exact function $f_{(S)}(x)$ from $E = [0,1]^3$ to $\{0,1\}$ is built, which can tell as an oracle whether there exists or not a control $u \in U$ such that the trajectory starting at point x with control u belongs to a given set S at the following time step. If this set S is $Capt(D,T)$ then $x \in Capt(D,T+dt)$ when $f(x) = 1$, and $x \notin Capt(D,T+dt)$ when $f(x) = 0$.

As in [Rouquier *et al.*,], we use kd-trees to discover the boundary of $Capt(D,T+dt)$. The kd-tree subdivides the search space E. It focuses on the boundary by construction, so it reduces the number of calls to the oracle f, compared to the extensive search on the grid.

We first build a kd-tree g_{dt} to approximate $f_{(D)}$. So g_{dt} is an approximation of the characteristic function of $Capt(D,dt)$. We then recursively build a kd-tree g_{T+dt} that approximates $f_{g_{(T)}}$. By construction $g_{(T+dt)}$ is an approximation of the characteristic function of $Capt(D,T+dt)$.

3.2 Algorithm

The kd-tree $g = g_T$ is built by refining its root node which contains the whole space E. To this root node is associated a point x_0 of D such that an evolution starting at x_0 is still in D at time step T. Since D is a viability domain, any point in D is suitable. The root node is labeled to 1, since its associated point x_0 is in $Capt(D,T)$.

Nodes that are refined are always divided in two halves along their largest dimension, unless they meet the stopping criterion: a node is not divided any more if the length of its largest side is already h. These are called terminal nodes. With this splitting method, nodes are represented by a vector of half-open intervals (except for border nodes). Since intervals are split in the middle, they can't overlap, they are either disjoint, adjacent or included in one another. Nodes that have to be refined are:

- Border nodes (nodes that are adjacent to the boundary of the search space E) with label 1.

could be D itself. In this latter case, outside D, one of the languages is doomed to be spoken by less than a proportion $\underline{\sigma} = 0.2$ of the population.

- Adjacent nodes with different labels (they are called critical pairs).

Two nodes are adjacent if and only if on each coordinate axis, one interval is included in (or equal to) the other, except for one coordinate where both intervals are adjacent (their closures have exactly one common point). When a node is refined, a point p is drawn at random (other method can be used) in each child node which is labeled with $f(p)$. .

The sketch of the algorithm is the following:

Algorithm 1 ***BuildG(***$root, h$***)***
0. `while` *count* $\neq 0$ `do` {
1. *count* $\leftarrow 0$
2. $A \leftarrow$ `NodeToDivide`$(root)$
3. `foreach` $node_i \in A$ `do` {
4. `if` $node_i$ *is not a terminal node* `then do` {
5. *count* $\leftarrow$ *count* $+ 1$
6. `Refine`*(*$node_i, h$*)*
7. $A \leftarrow A \backslash node_i$ *(a node is refined only once)* }}
8. `return` $g = root$

`Refine`$(node, h)$ creates two children at node $node$, by splitting it along its largest dimension if its greater than h. If $node$ is represented by the vector of intervals $([a_i, b_i[)_{1,..,n}$ and is divided along j, its children will be have the same vector values except for the j^{th} coordinate which is $[a_j, \frac{(b_j - a_j)}{2}[$ for one child and $[\frac{(b_j - a_j)}{2}, b_j[$ (if $b_j \neq 1$) for the other ($[\frac{(1-a_j)}{2}, 1]$ with $b_j = 1$). A call to the oracle is made for the child that need it in order to attribute a label to the node. (Other methods may require 2 calls).

Algorithm 2 ***NodeToDivide(***$node$***)***
0. `if` *node is a leaf* `then`
1. `if` $f(node) = 1$ *AND node is a border*
`then return` $\{node\}$
2. `else return`$(\emptyset)$
3. `else` { $(node_1, node_2) \leftarrow$ *node.children*
4. *result* $\leftarrow \emptyset$
5. *result* $\leftarrow$ *result* $\cup$ `NodeToDivide`*(*$node_1$*)*
6. *result* $\leftarrow$ *result* $\cup$ `NodeToDivide`*(*$node_2$*)*
7. *result* $\leftarrow$ *result* $\cup$ `PairsInNodes`*(*$node_1$*,* $node_2$*)*
8. `return` *result* }

Algorithm 3 ***PairsInNodes(***$node_1$***,*** $node_2$***)***
$node_1$ *and* $node_2$ *are necessarily adjacent*
0. *result* $\leftarrow \emptyset$
1. `if` *both* $node_1$ *and* $node_2$ *are leaves* `then`
2. `if` $f(node_1) \neq f(node_2)$ `then`
3. *result* $\leftarrow \{node_1, node_2\}$
4. `else if` *neither* $node_1$ *nor* $node_2$ *are leaves* `then`
5. `foreach` $node_i$ *child of* $node_1$ `do` {
6. `foreach` $node_j$ *child of* $node_2$ `do` {
7. `if` $node_i$ *and* $node_j$ *are adjacent* `then`
8. *result* $\leftarrow$ *result* $\cup$ `PairsInNodes`*(*$node_i$*,* $node_j$*)* }}
9. `else do` {
10. $leaf \leftarrow$ *the leaf (either* $node_1$ *or* $node_2$*)*
11. $node \leftarrow$ *the other node*
12. *border* $\leftarrow$ *axis and point of adjacency*
13. *label* $\leftarrow 1 - f(leaf)$
14. `foreach` $node_i \in$ `BoundaryNodes`*(node, border, label)* `do` {
15. `if` $leaf$ *and* $node_i$ *are adjacent* `then`
result $\leftarrow$ *result* $\cup \{leaf, node_i\}$ }
16. `return` *result*

`BoundaryNodes`$(node, border, label)$ returns all the leaves l within $node$ such that $f(l) = label$ and that share with $node$ the same boundary $border$.

The characteristic function is built by calling `NodeToDivide` on the search space E.

3.3 Convergence

The algorithm refines a subtree (generally the root) by splitting leaves in two, so it reaches the stopping criterion in finite time.

We show here that the series of sets defined by the kd-tree algorithm converges towards the viability kernel of the associated viability problem. The method we use verifies the conditions of the convergence theorem 1 proposed in [Deffuant *et al.*, 2007]. This theorem applies when a discrete viability kernel is approximated with a classification procedure.

A general viability problem is defined by a controlled dynamics in $E \subset \mathbb{R}^p$ modeled by a set of equations such as (1), and a compact set of constraints $K \subset E$, such as (2) in which we want the dynamics to evolve:

$$\left\{ \begin{array}{lcl} x'(t) & = & \varphi(x(t), u(t)) \\ u(t) & \in & U(x(t)) \subset \mathbb{R}^q \\ x(t) & \in & K \subset E \end{array} \right. \quad (3)$$

The viability kernel is the subset of states from which there exists a control function $u(t)$ that allows the evolution of the dynamics to stay inside K: $\text{Viab}(K) = \{x \in K \mid \exists u(.) \mid x(t) \in K \ \forall t \in [0, +\infty[\}$. Considering a time interval *dt*, the discrete dynamical system F associated to 3 is a correspondence $E \mapsto E$ which associates to a state x its set of successors. We assume that F is μ-Lipschitz, which means that the images of two vectors x and y can't diverge from more than $\mu d(x, y)$:

$$F(x) = \{x + \varphi(x, u)dt, u \in U(x)\}. \quad (4)$$

As for the language competition model, we consider that an oracle f is available, to tell whether at the following time step a state evolves towards a given set S. In order to guarantee the convergence toward the viability kernel it is necessary to consider an augmented set; The oracle is assumed to be defined from E to $\{0, 1\}$ in the following way: $\hat{f}_{(S)}(x) = 1$ if and only if there exists $u \in U(x)$ such that $d(F(x), S) \leq \mu\beta(h)$, otherwise $\hat{f}(x) = 0$ (where $\beta(h)$ depends on the grid and tends to 0 as h tends to 0). This means that the oracle $\hat{f}$ is based on the characteristic function of $S \cup (\cup_{x \in S} B(x, \mu))$. We have $\hat{f}_{(S)} = f_{S \cup (\cup_{x \in S} B(x, \mu))}$.

We want to prove that algorithm 1 converges to the characteristic function of the viability kernel $viab_F(K)$ of the discrete dynamic (4) when its stopping criterion h tends to 0 and when the viability kernel has sufficiently smooth properties.

Theorem [Deffuant *et al.*, 2007] assures that the series of sets built with a learning procedure verifying conditions (5)

and (6) converges towards the viability set. In order to apply this theorem, we consider a discrete grid K_h induced by the stopping criterion h. For example, the points of the grid can be centered on each terminal leaf of a totally refined tree. We obviously have: $\forall x \in K, \exists x_h \in K_h$, such that $d(x, x_h) \leq \beta(h)$, with $\beta(h) \to 0$ when $h \to 0$. (Actually $\beta(h) = \frac{h}{2}\sqrt{p}$). In the following we omit the stopping criterion index h when it is not necessary. At each time step we define a discrete set $K^{n+1} \subset K^n \subset K_h$, and we build a kd-tree g on the sets K^{n+1}, $K_h \backslash K^{n+1}$ with algorithm 1. K^{n+1} gathers the points of K^n such that the dynamics (4) send them near $g(K^n)$ (at a distance less than $\mu\beta(h)$). We note g^n the kd-tree built with the subsets of K^n and $K_h \backslash K^n$ that correspond to points x_h for which the oracle $\hat{f}$ has been called when refining the tree. We note $G^n = \{x \in K \mid g^n(x) = 1\}$. We have $K^0 = K_h$ and $G^0 = K$. Then:

$$K^{n+1} = \left\{ x_h \in K^n \mid \hat{f}_{G^n}(x_h) = 1 \right\}.$$

Since $K^{n+1} \subset K^n \subset K_h$ it converges to a limit L_h. We note g_h the kd-tree built on the subsets L_h and $K_h \backslash L_h$, and $G_h = \{x \in K \mid g_h(x) = 1\}$.

Theorem 1 (fact) *If the learning procedure g verifies the following conditions (5) and (6), then G_h converges to Viab(K) when $h \to 0$.*

$$\exists \lambda \geq 1,\ \forall n > 0,\ (g^n(x) = 1) \Rightarrow d(x, K^n) \leq \lambda\beta(h) \quad (5)$$

$$\forall n > 0,\ (g^n(x) = 0) \Rightarrow d(x, K_h \backslash K_h^n) \leq \beta(h) \quad (6)$$

We consider that the viability kernel $viab_F(K)$ (and its complementary set in K) verify the following conditions: We note $V_\epsilon(S) = \{x \in S \mid B(x, \epsilon) \subset S\}$ (V is an open space in S)

$$\exists \epsilon > 0 \mid V_\epsilon(viab_F(K)) \text{ and } V_\epsilon(K \backslash viab_F(K)) \text{ are path-connected.} \quad (7)$$

$$\begin{cases} \forall x \in viab_F(K), & d(x, V_\epsilon(viab_F(K))) \leq \epsilon d \\ \forall x \in viab_F(K), & d(x, V_\epsilon(viab_F(K))) \leq \epsilon d \end{cases} \quad (8)$$

The path-connectivity of the viability kernel and its complementary are insured when K is simply connected and sufficiently regular. Conditions 8 insure that when the stopping criterion h is small enough, then there are no more thin tentacle that can't be seen through the grid.

Theorem 2 *The series of characteristic functions defined by the kd-trees build with algorithm 1 converges to the characteristic function of the viability kernel when the stopping criterion tends to* 0.

Proof: The learning procedure g based on algorithm 1 verifies the conditions of Theorem 1.

We first prove condition (6). Let x be a state in K such that $g^{n+1}(x) = 0$. This means that x belongs to a leaf L whith labeling point $x_h \in K \backslash K^{n+1}$. If L is a terminal node, then it is divided to the stopping criterion, and we have: $d(x, x_h) \leq \beta(h)$. If the leaf L of x is not a terminal node, let us consider the terminal node N to which x would belong if the node L was divided into a subtree to the stopping criterion. Let y_h be the labeling point of N: we have $d(x, y_h) \leq \beta(h)$. We also have $N \subset L$. Since L is not a terminal node, the closest labeling point y of a terminal node with label 1 is at least separated from L by another terminal node with label 0. So necessarily $y_h \neq y$ and so $y_h \in K^{n+1}$ and condition (6) is fulfilled. Since the kd-tree algorithm 1 proceeds in the same way whatever the label of the nodes, the condition (5) is also fulfilled. •

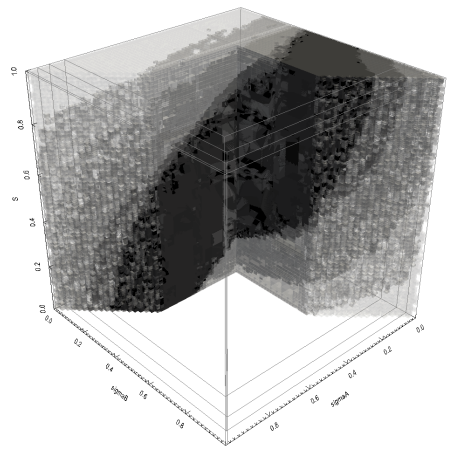

Figure 2: Level set of the resilience at 0.5; 0.2; 0.1; 0.08; 0.05; 0.025, corresponding to time level 2, 5 ,10, 15, 20, 40. Stopping criterion $h = 2^{-18}$. Lighter colors correspond to less resilient states.

The oracle $\hat{f}$ is defined with the following procedure: Let g be a kd-tree build with algorithm 1. g is the characteristic function of G. We build a kd-tree $\hat{g}$ by changing the label of critical leaves to 0, then refining the tree again with algorithm 1. This adds a level of new critical leaves with label 0 (refined from larger nodes with label 0), that are adjacent to the leaves whose label have been modified. If we note $G = \{x \in K, g(x) = 1\}$ and $H = \{x \in K, \hat{g}(x) = 0\}$, we have: $h\sqrt{p} \geq d(G, H) \geq h$. This operation is repeated k times until $kh \geq \mu\beta(h)$. So the use of $\hat{g}^(k)$ is a bounded approximation of the oracle $\hat{f}$. This guarantees the convergence of the process to the viability kernel.

4 Preservation of language diversity

4.1 Full resilience of bilingual societies

We used algorithm 1 in order to compute an approximation of the capture basin of the viability domain D of the language competition problem. [2]

From the viability domain D, the algorithm computes the successive approximations of the capture basin. Figure 2 shows the capture basin of D at different time horizons. The boundaries are the level sets of the resilience.

[2] For this experiment we use directly f as the oracle function and not $\hat{f}$, and the points associated to the leaves are drawn at random. It reduces the computation time (but the convergence is no longer strictly decreasing).

Figure 3 shows the evolution of the volume of the different approximation of the capture basin in language competition problem when increasing the time horizon. Since we necessarily have $\sigma_A + \sigma_B \leq 1$, the volume of the state space is 0.5. We can see that the volume of the capture basin actually converges to 0.5. This means that bilingual societies are fully resilient: the capture basin of the viability domain covers the entire space. So whatever the perturbations, there

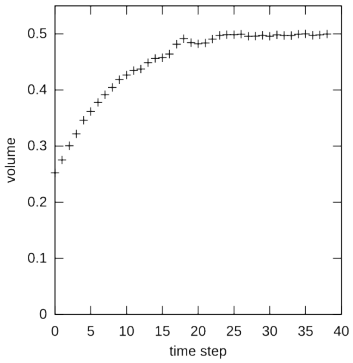

Figure 3: Evolution of the volume of the capture basin with time horizon. Convergence to 0.5 occurs around time step 25. (In this experiment the splitting direction in refined node is selected randomly, so the convergence is not monotonous)

always exists a control policy that allows to return in the set of desirable states in less than 25 time steps, so the global resilience is $r = 0.04$.

With this information it is possible to accept temporary undesirable situations from the language diversity viewpoint, such as receiving refugees, which can shift the present state in D to a state outside D. For instance if the initial state is $(\sigma_A = 0.28, \sigma_B = 0.48, s = 0.40)$ it can be decided to accept a disturbance to the state $(\sigma_A = 0.22, \sigma_B = 0.54, s = 0.40)$ since its returning time step is only 4.

4.2 Choosing a policy after a disturbance

If a disturbance is inescapable (for instance aging of population, etc.), the computation of the capture basin provides control functions in order to return to the set of desirable states D. For example, we consider a bilingual society which state is supposed to be $(\sigma_A = 0.35, \sigma_B = 0.35, s = 0.2)$. This means that the proportion of bilingual speakers is 30%. This state is inside D, which means that an appropriate control of s can maintain the bilinguism property. We suppose now that a perturbation occurs, like a large immigration of speakers of language A. It changes the proportion of speakers of language B from 0.35 to 0.41 and the proportion of speakers of language A to 0.30. The new state $P = (0.30, 0.41, 0.2)$ is not in D. P is very close to D but even with the maximal control available $\frac{ds}{dt} = 0.1$, the dynamics drives the system outside the constraint set K (at time step 3), as it can be seen on figure 4. Since the system is resilient, we already know that it is possible to return to the set of desirable states K. The resilience of P is $r(P) = 0.14$, which means that it can return in 7 time steps to point $Q = (0.2, 0.61, 0.90)$.

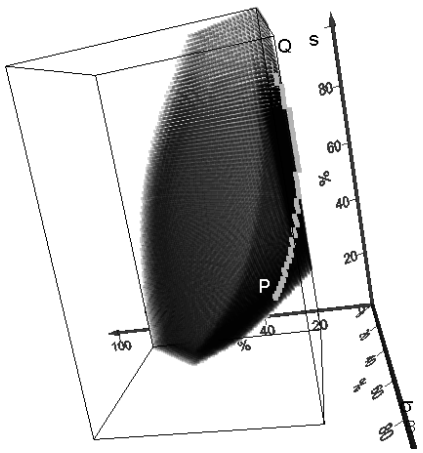

Figure 4: Resilient trajectory from the disturbed state P towards the reentry point Q. The model provides the continuous trajectory, while the kd-tree provides the control values.

5 Conclusion

We have presented in this article a new method to compute the level sets of the resilience defined in the framework of the viability theory. This method uses kd-trees with a stopping criterion h to store the characteristic function of the capture basin of the viability kernel (which provides the level set of the resilience). It uses a slightly modified kd-tree to approximate the oracle function that allow to compute these sets. The number of calls to the oracle function is limited to the number of nodes of the kd-trees. The building function concentrates these calls at points near the boundary of the viability set. We have proved that this algorithm converges towards the viability kernel when the stopping criterion h tends to 0. Moreover, the algorithm that codes the dynamics is a black box for the algorithm that build the viability kernel (or the capture basin). This modular aspect guarantees the reusability of our approach to other dynamics (the code is available as free software). We have shown how this method can be used in a language competition problem. We have shown how a resilience value can be defined. We have computed the level sets of the resilience and we have shown how it can be used to propose action policy in order to guarantee the language coexistence. This method focuses on the boundary of the viability sets rather than on the sets themselves, so it allows to consider state space with one additional dimension But it still suffers the curse of dimensionality. Further work is in progress in order to parallelize the refining part of the algorithm. Another limit is that in order to provide efficient and realistic action policies it is necessary to know the distance of a state to the boundary of the viability kernel. The algorithm can easily provide an approximation of this distance, so further work also concentrates on this point.

References

[Abrams and Strogatz, 2003] D.M. Abrams and S.H. Strogatz. Modelling the dynamics of language death. *Nature*, 424(6951):900, 2003.

[Aubin *et al.*, 2011] J.-P. Aubin, A. Bayen, and P. Saint-Pierre. *Viability Theory: New Directions*. Springer, 2011.

[Aubin, 2001] J.-P. Aubin. Viability kernels and capture basins of sets under differential inclusions. *SIAM Journal Control Optim.*, 40(3):853–881, 2001.

[Bernard and Martin, 2012] C. Bernard and S. Martin. Building strategies to ensure language coexistence in presence of bilingualism. *Applied Math. and Comp.*, 218(17):8825 – 8841, 2012.

[Bonneuil, 2006] Noel Bonneuil. Computing the viability kernel in large state dimension. *Journal of Mathematical Analysis and Applications*, 323(2):1444 – 1454, 2006.

[Bruckner *et al.*, 2003] T. Bruckner, G. Petschel-Held, M. Leimbach, and F. L. Toth. Methodological aspects of the tolerable windows approach. *Climatic Change*, 56:73–89, 2003.

[Deffuant *et al.*, 2007] G. Deffuant, L. Chapel, and S. Martin. Approximating viability kernels with support vector machines. *IEEE T. Automat. Contr.*, 52(5):933–937, 2007.

[Doyen and Saint-Pierre, 1997] L. Doyen and P. Saint-Pierre. Scale of viability and minimum time of crisis. *Set-Valued Anal.*, 5:227–246, 1997.

[Martin, 2004] S. Martin. The cost of restoration as a way of defining resilience: a viability approach applied to a model of lake eutrophication. *Ecol. Soc.*, 2(9):8, 2004.

[Perrings, 2006] C. Perrings. Resilience and sustainable development. *Environ. Devel. Econ.*, 11:417–427, 2006.

[Reuillon *et al.*, 2010] Romain Reuillon, Florent Chuffart, Mathieu Leclaire, Thierry Faure, Nicolas Dumoulin, and David Hill. Declarative task delegation in OpenMOLE. In *HPCS*, pages 55–62, 2010.

[Rouquier *et al.*,] J.-B. Rouquier, I. Alvarez, R. Reuillon, and P.-H. Wuillemin. A kd-tree algorithm to discover the boundary of a black box hypervolume. Technical Report hal-00816704, LIP6.

[Saint-Pierre, 1994] P. Saint-Pierre. Approximation of the viability kernel. *Applied Mathematics & Optimisation*, 29:187–209, 1994.

[Sicard *et al.*, 2012] M. Sicard, N. Perrot, R. Reuillon, S. Mesmoudi, I. Alvarez, and S. Martin. A viability approach to control food processes: Application to a camembert cheese ripening process. *Food Control*, 23(2):312 – 319, 2012.