Chapter 4

From clusters to compartments

The goal now is to use a clustering algorithm on an advection-diffusion problem, and ultimately use the partitions to build a box-model. The underpinning idea is that communities found on the dynamic of the flow could be relevant compartments for the box-model. The idea makes sense although it is far from being obvious that the communities are indeed the compartments that we seek. In a first instance, we should only check that, at least in some cases, community detection leads to relevant partitions. Hence, we first build a test problem for which we know in advances what compartments to expect. Then we apply the method on the less obvious, though still very naive overturner model of the Atlantic ocean.

Remark In the context of interpreting the communities as compartments for a box-model, we should require that the communities have vertical and horizontal boundaries. When dealing with marine problems, the goal of a box-model is to provide a simple and intuitive description of the problem. Complex shaped compartments are neither simple nor intuitive for marine models, hence the requirement.

4.1 Methodology

4.1.1 Description of the method

In order to apply a clustering algorithm on a physical advection-diffusion problem, we have to define how the problem can be considered as a graph. For the next, we consider a two-dimensional problem in the coordinate system (y, z). Let us partition the domain into $n_{box,\,y} \times n_{box,\,z}$ boxes, and denote $N_{box} = n_{box,\,y} n_{box,\,z}$ the total number of boxes. Figure 4.1 represents an example of such a domain decomposition of the overturner problem with $n_{box,\,y} = 15$ and $n_{box,\,z} = 10$. For any time T, the corresponding directed graph is build as follows: each node represents a box, and the weight of the edge between nodes i and j is the probability $m_{ij}(T)$ that a particle ends up in box j after a time T if it was initially in box i. If $m_{ij}(T) = 0$, one can equivalently consider that there is no edge between nodes i and j. Since the problem is stationary, $m_{ij}(T)$ depends only on the elapsed time T, not on the initial time. Hence, the initial time can indifferently be considered as being zero. The adjacency matrix $\mathbf{M}(T)$ of the graph is build from the weights $m_{ij}(T)$: $[\mathbf{M}(T)]_{ij} = m_{ij}(T)$. For any time T, $\mathbf{M}(T)$ is row-stochastic, i.e. $\mathbf{M}(T)\mathbf{1} = \mathbf{1}$, where $\mathbf{1}$ is the N_{box}-dimensional unit column vector. The latter has a straightforward physical interpretation: every particle remains in the domain.

To estimate the probabilities $m_{ij}(T)$, a Lagrangian simulation is run for a time T with each

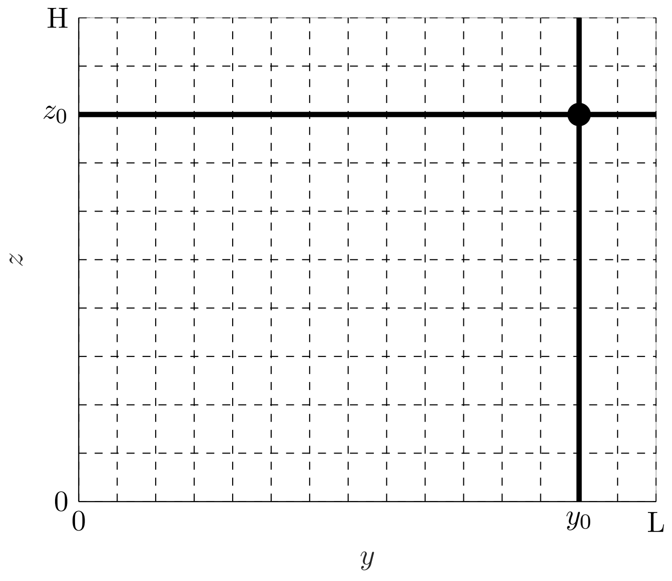

Figure 4.1 – Illustration of the decomposition of the domain into boxes with $n_{box,\,y} = 15$ and $n_{box,\,z} = 10$.

box containing initially J uniformly distributed particles. $m_{ij}(T)$ is then numerically estimated as the number of particles having started in box i and ending up in box j, divided by J. This *box counting* method has been extensively used to estimate the concentration in studies using random walk modeling, see e.g. [28]. Nevertheless, this method suffers some drawbacks; the most important of them are pointed in [3], but we recall them here for the sake of completeness. First, the estimated transition probability depends on the choice of the boxes, in particular of their size and their center. Moreover, the number of boxes cannot be chosen to be too large; otherwise the estimated concentration tends to become very irregular or noisy. Finally, the resolution is limited to the size of the boxes, as the concentration cannot be described in a box more precisely than a constant. But it is the perfect method for our problem since the volume average over such boxes (the nodes) is precisely what we want. Note however that other methods exist for estimating the concentration, that might be better suited for other studies. For example, the *kernel estimation* method allows to reduce drastically the number of particles, and does not suffer from the resolution limit inherent to the box counting method. However, the kernel estimation method introduces some difficulties. A first difficulty is that this method depends on a parameter called the *bandwidth* for which finding a relevant value is not trivial. Second, the method as such does not perform well at the boundaries, and a specific treatment of the boundaries must be introduced. This kernel estimation method is briefly presented in [3]. Classical references are [29] and [30].

4.1.2 Dealing with the time scales

An important feature of the stability method for detecting community structures is that it is *dynamic*: community structures are revealed as a function of the Markov time t_M. For the problems that we consider, this Markov time is intrinsically linked to the physical time: for a given time T, suppose that the stability method is applied on the adjacency matrix $\mathbf{M}(T)$. In the discrete framework, a particle jumps from one node to another at every integer Markov time. Hence, a Markov time step of 1 corresponds to a physical time step of T.

From the above discussion, two possibilities arise for dealing with the time scales: either we compute the adjacency matrix at one unique time T and then compute the stability on the desired range of Markov times, or we compute the adjacency matrix at different times and then compute the stability on each adjacency matrix but for the Markov time $t_M = 1$ only. The advantage of the first method is that we do not have to fix *a priori* the time scales at which we compute clusterings: such times scales arise naturally as plateaux in the community curve, and a low corresponding variation of information. Hence the relevant time scales are deduced from the stability curve as being the ones at which robust clusterings arise. At the contrary, the second method imposes that we choose the time scales beforehand; doing so, we lose one of the most appealing features of the stability approach. Furthermore, the first method is computationally lest costly. Even for a relatively coarse partitioning of the domain, say of about 300 boxes, if we release 10 000 particles in each box there is a total of 3×10^6 trajectories to simulate. For long T, the simulation time might become restrictive, especially in the case where one does not have access to supercomputers to run the code in parallel. The same situation leads to a 300 nodes network, which is a relatively small network size that can easily be handled by the stability software. The first method has however one important drawback: the errors in the adjacency matrix are spread and even amplified across the Markov times. If those errors become too important, the community structures found at large Markov times might become irrelevant. The ideal methodology is thus probably to use the first method to detect the interesting time scales and compute the corresponding community structures, and then to check that we get similar community structures at the same time scales using the second method.

4.1.3 Use of the stability software

We present here briefly how the *PartitionStability* software is used to compute the partitions. Every concept appearing here has been presented in section 3.1. The `stability` function is simply called as follows :

```
[S,N,VI,C] = stability(M,Markov_T,'directed','plot','teleport',tau);
```

Here, `M` is the matrix $\mathbf{M}(T)$ at the desired time T; `Markov_T` is the vector containing every Markov times at which the optimal stability partition has to be computed (ideally, the sampling should be exponential); the `'directed'` option specifies that we consider a directed graph; `'plot'` asks the program to plot the stability, number of communities and variation of information as a function of the Markov time; and `'teleport',tau` allows to specify the value of the teleportation probability τ to `tau`, the default value being 0.15. In most cases, we will choose `tau` $= 0$. This choice is motivated by the fact that if our approximation of the transition probability matrix is close enough the the exact one, then if the diffusivities are everywhere strictly positive the graph is ergodic (notice that there can be no dangling node whatever the precision of our approximation). Further, one example where the graph is not ergodic will be encountered. In that case, the value of `tau` must be chosen strictly positive in order to ensure ergodicity. A small value is then preferred, in order to minimize the impact of random teleportations on the dynamics of the graph. We will typically choose `tau` $= 10^{-3}$ in such a case.

Unfortunately, the software does not handle discrete-time stability. Instead, it allows to choose which type of laplacian should be used to calculate the (continuous-time) stability. However, the question does not arise here since both laplacians are equivalent in our case. Indeed, the total outgoing weight is the same at every node and is precisely equal to the number of particles J released in each box. Hence, $k_i = J$ for every node i and $\langle \mathbf{k} \rangle = J$, so that $\boldsymbol{\lambda}_{combi}(\mathbf{k}) = \mathbf{k}/\langle \mathbf{k} \rangle = \mathbf{1} = \boldsymbol{\lambda}_{norm}(\mathbf{k})$. We let thus the program run with the default normalized Laplacian, since it does not make any difference in our case.

The output arguments `S`, `N`, `VI` and `C` contain respectively the stability, the number of communities, the variation of information, and the optimal partition for each Markov time contained in `Markov_T`. If the latter is of size n, then `S`, `N` and `VI` are n-dimensional vectors and

`C` is a $N_{box} \times n$ matrix. At the jth Markov time, communities are labeled by consecutive integers between 0 and `N(j)`-1 such that `C(i,j)` $= k$ means that node i belongs to community k at Markov time `Markov_T(j)`.

4.2 Application of the method on a simple class of problems

This section can be considered as a kind of *sanity check* of the method: we define a class of problems for which the box decomposition is intuitively obvious and we check that a community detection algorithm applied on the problem allows to find back that box-decomposition. We call that class of problems the *bi-overturner* problems. The domain that we consider is $\Omega = [-L,\, L] \times [0,\, H]$. Bi-overturner problems basically consist of two *overturner-like* circulations models side-by-side. Let $\Omega^- = [-L,\, 0[\times[0,\, H]$ and $\Omega^+ =]0,\, L] \times [0,\, H]$. If $\psi(y, z; y_0, z_0)$ denote the streamfunction defined in (1.20) with parameters $y_0 \in]0,\, L[$ and $z_0 \in]0,\, H[$, then the streamfunction ψ_2 of the bi-overturner problems is defined as

$$\psi_2(y,z) = \begin{cases} \psi(L+y, z; y_0, z_0) & \text{if} \quad (y,z) \in \Omega^-, \\ 0 & \text{if} \quad (y,z) \in \{(0,z) \mid z \in [0,\, H]\}, \\ -\psi(y, z; L-y_0, z_0) & \text{if} \quad (y,z) \in \Omega^+, \end{cases} \tag{4.1}$$

The streamfunction ψ_2 has two extrema of equal strengths: a maximum at (y_0^-, z_0) with $y_0^- \triangleq -L + y_0 = -y_0^-$ and a minimum at (y_0^+, z_0) with $y_0^+ \triangleq L - y_0$. The overturner-like circulation is clockwise in Ω^- and counterclockwise in Ω^+. The horizontal and vertical velocities v and w are given by

$$v = -\frac{\partial \psi_2}{\partial z}, \quad w = \frac{\partial \psi_2}{\partial y}. \tag{4.2}$$

Isolines are shown in figures 4.2 and 4.3 respectively. The key feature is that $v(0, z) = 0$, namely the horizontal velocity is zero on the whole segment $y = 0$. Hence, if there is no horizontal diffusion, a passive tracer's particle starting in Ω^- can never reach Ω^+ and conversely. In that case, we can imagine that there is a vertical wall implying no-through boundary conditions at $y = 0$ and the graph is not ergodic. But if the horizontal diffusivity is nonzero in some area near $y = 0$, then exchange of particles between Ω^- and Ω^+ can happen in that area. Now we suppose that the diffusivity tensor $\mathbf{K}$ is diagonal:

$$\mathbf{K}(y,z) = \begin{pmatrix} K_h(y,z) & 0 \\ 0 & K_v \end{pmatrix}. \tag{4.3}$$

We assume that the vertical diffusivity K_v is constant and equal to 10^{-3} [m^2/s]. Now we introduce the parameter $\alpha \in [0, 1]$ and define $z^* = \alpha H$. We choose an horizontal diffusivity K_h of the form

$$K_h(y,z) = \begin{cases} 10^4 \text{ [m}^2\text{/s]} & \text{if} \quad y_0^- \le y \le y_0^+ \quad \text{and} \quad z^* \le z \le H, \\ 10^3 \text{ [m}^2\text{/s]} & \text{if} \quad -L \le y < y_0^- \quad \text{or} \quad y_0^+ < y \le L, \\ 0 \text{ [m}^2\text{/s]} & \text{otherwise.} \end{cases} \tag{4.4}$$

Now, exchange between Ω^- and Ω^+ is possible but only above z^*.[1] For the next, we call *exchange zone* the area where $K_v = 10^4$ [m^2/s] (dark gray zone in figure 4.4) Hence, we can imagine that there is a vertical, no-through wall of height z^* at $y = 0$. The situation is depicted on figure 4.4. Making α vary between 0 and 1 defines a class of bi-overturner problem where the vertical wall's height z^* at $y = 0$ vary between 0 and H. Two examples of trajectories with the same initial condition are shown for $\alpha = 0.75$ in figures 4.5 and 4.6.

[1] From a numerical, random-walk, point of view, we should also ensure that $y_0^+ = |y_0^-|$ is sufficiently large. If not, it would be numerically possible for particles lying below z^* and before y_0^- (resp. after y_0^+) to jump from Ω^- (resp. Ω^+) to Ω^+ (resp. Ω^-).

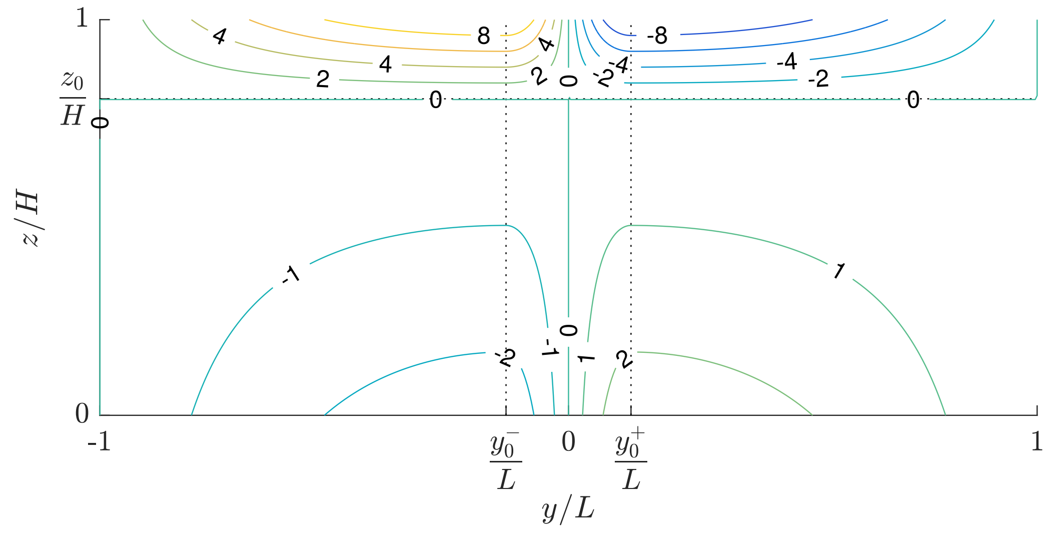

Figure 4.2 – Isolines of the horizontal velocity v for bi-overturner problems.

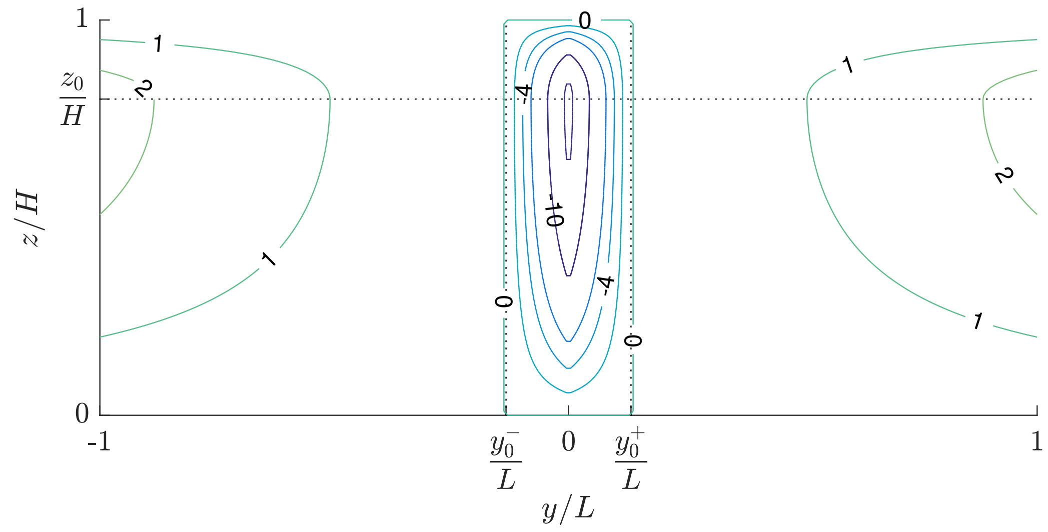

Figure 4.3 – Isolines of the horizontal velocity w for bi-overturner problems.

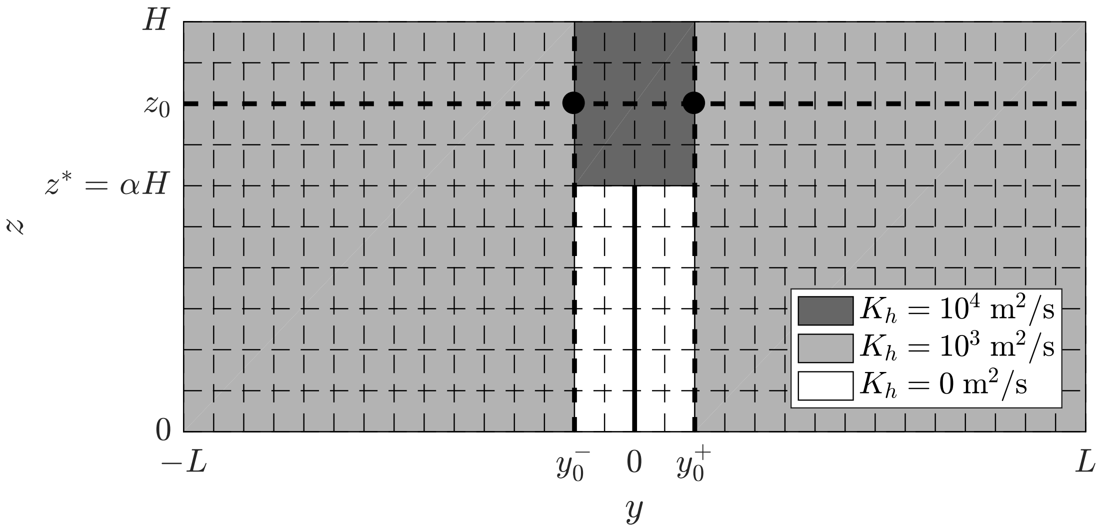

Figure 4.4 – Illustration of the decomposition of the domain into boxes corresponding to the nodes of the directed graph. The values of K_h are also shown for $\alpha = 0.6$, and the fictitious wall is represented by the black continuous line. Here the particle enters the exchange zone but finally stays in Ω^-.

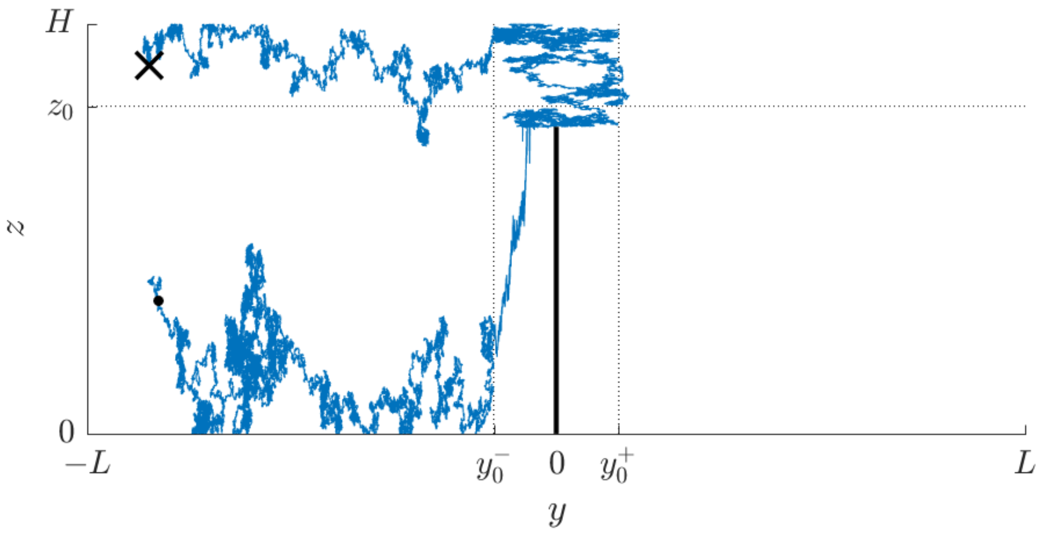

Figure 4.5 – Example of a particle trajectory in the bi-overturner model with $\alpha = 0.75$. The black cross represents the initial position whereas the black dot shows the final position. The simulation time is 200 years.

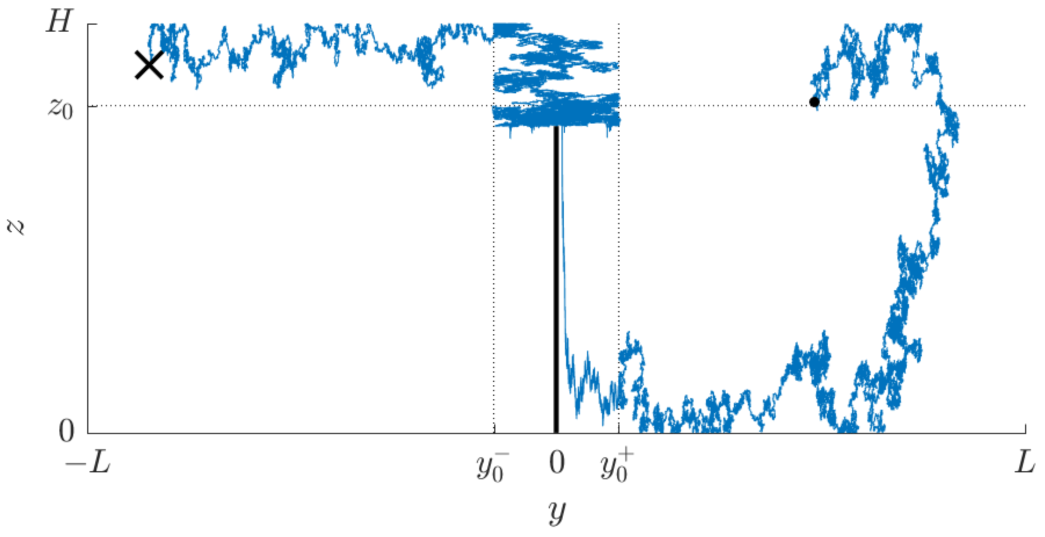

Figure 4.6 – Example of a particle trajectory in the bi-overturner model with $\alpha = 0.75$. The black cross represents the initial position whereas the black dot shows the final position. The simulation time is 200 years.

When $\alpha = 1$, the obvious compartmental model is made of the two compartments Ω^- and Ω^+ which do not communicate with each other. Suppose that different amounts of passive tracer are released into Ω^- and Ω^+ at a given time; the concentration in each compartment tends to become uniform in time due to diffusion, but the concentration in Ω^- depends only on the initial quantity of tracer released in Ω^- and similarly for the concentration in Ω^+. At the contrary, when $\alpha = 0$ the concentration tends to become uniform over the whole domain when time goes to infinity. Hence, we can expect three types of partitioning: a three-communities partitioning with left and right compartments and an exchange zone in between; a two-communities partitioning corresponding to Ω^- and Ω^+; and finally a trivial partitioning with one single community for very long time scales. For intermediate values of α, we expect a behavior similar to the case $\alpha = 1$ when α is close to 1 and similar to the case $\alpha = 0$ when α goes to 0.

4.2.1 Results

The partitioning results are presented here for $\alpha = 1$, $\alpha = 0.75$, $\alpha = 0.5$, $\alpha = 0.25$ and $\alpha = 0$. For every value of α, the transition probability matrix is computed at $T = 1$ year on a discretization like the one presented in figure 4.4, namely with $n_{box,\,y} = 30$ and $n_{box,\,z} = 10$. 10 000 particles are initially released in every box. The stability software is run using a vector of Markov times taking values between 10 and 10^3. Since we have computed the transition probability matrix for $T = 1$ year, one unit of Markov time correspond here to one year. Notice that in the case where $\alpha = 1$, the graph is not ergodic (it is composed of two ergodic classes) and a random teleportation probability $\texttt{tau} = 10^{-3}$ is used when running the stability software. When $\alpha < 1$, $\texttt{tau} = 0$ is used. The stability, number of communities and variation of information curves are shown in figures 4.7, 4.8, 4.9, 4.10 and 4.11 for different values of α. Most robust communities correspond to plateaux in the community curve together with a low variation of information. Whatever the value of α, the number of communities goes to 2 after maximum 50 years, together with a variation of information that is almost zero. In every case, the two-communities partitioning corresponds as expected to Ω^- and Ω^+. It is shown in figure 4.12 for the case $\alpha = 0.25$. In that case, the boundary is perfectly straight which corresponds to the intuition and is conform to the remark made on page 36. In some other cases, like when $\alpha = 0.5$, the boundary is not exactly a straight line. The situation is depicted in figure 4.13. However, we have to remember that neither the transition probability matrix nor the stability partitioning is solved exactly. Hence, we can consider that the irregularity in the boundary of the communities is due to those numerical artifacts: if a box-model has to be build from the partitioning shown in figure 4.13, the compartments should of course be chosen with a vertical boundary. This illustrates the fact that when using a community-detection algorithm to build compartments for a box-model, the communities should not be blindly interpreted as being the relevant compartments. In particular, if the boundaries of the communities are almost but not exactly vertical and horizontal, one should consider straight boundaries for the compartment. Community detection should thus be considered as a guide towards choosing relevant compartments, rather than as an exact method.

When $\alpha = 0.75$, a small three-communities plateaux starts appearing around 40 years. This plateau grows as α decreases. The corresponding clusterings are shown in figures 4.14 and 4.15, where a community corresponding to the exchange zone appears, as expected.

In figure 4.11 corresponding to the case $\alpha = 0$, peaks corresponding to oscillations between two and three communities are observed around 300 and 400 years. As stated in section 3.1, the number of communities should decrease with time, and those oscillations are thus due to the fact that the stability partitioning is only solved approximately. However, this shows that the two- and three-communities clusterings have similar stabilities at those times. Notice that in that case, we expected to find a single-community partitioning for very long Markov times, which does not appear here. Such a clustering would probably appear if we run the stability software for longer Markov times.

Hopefully this introductory example shows how a community-detection algorithm could be use to build compartment models, and provide intuition about why it could work. The communities found depend on the time scale considered but this is not a problem since it could also be the case for the compartments. Notice that we do not to build compartmental models for the bi-overturner class of problems because the compartments are obvious in this case, and it is thus not the goal of this section. In the next section, the method is applied on the overturner problem and we should try to build a compartmental model for that problem.

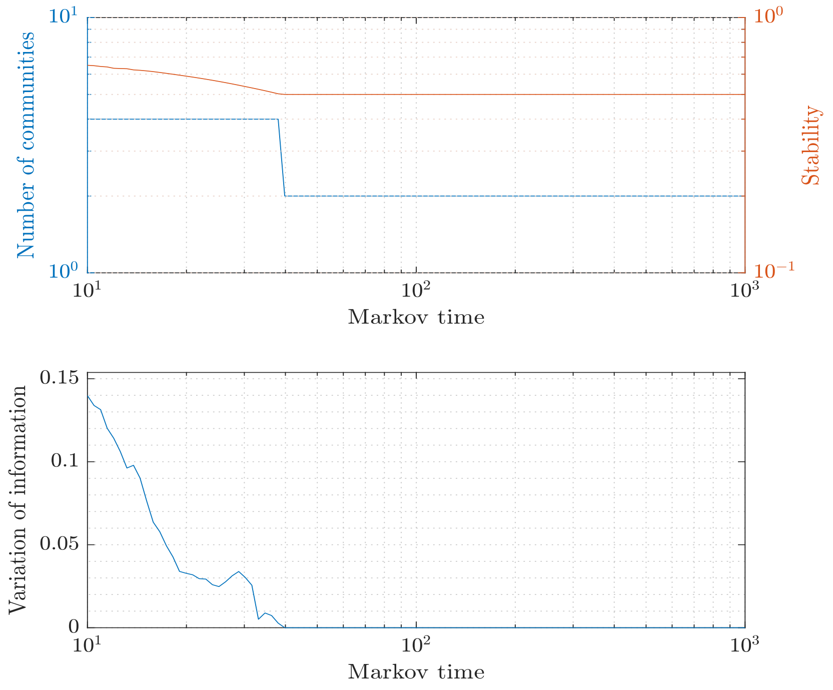

Figure 4.7 – Stability, number of communities and variation of information as a function of the Markov time for $\alpha = 1$.

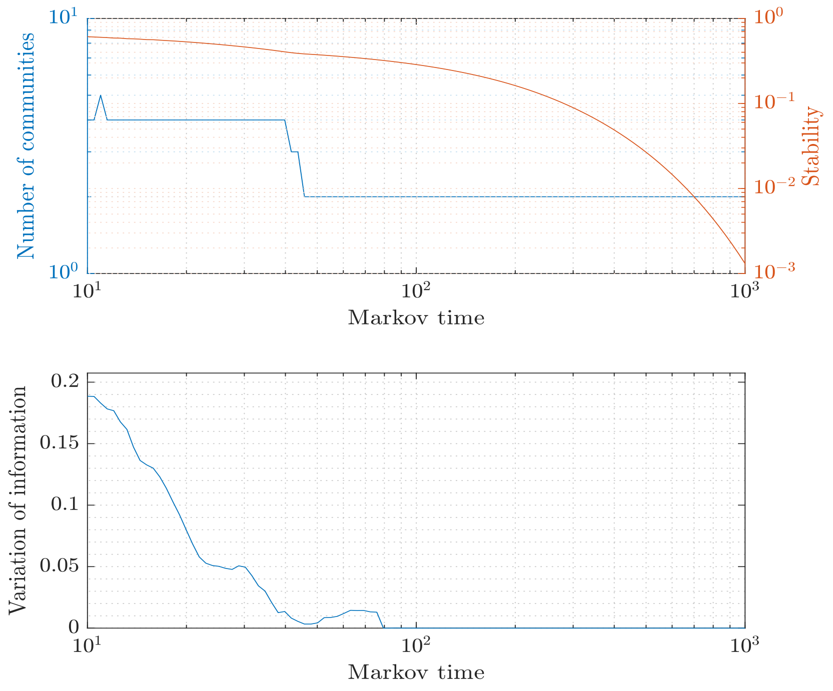

Figure 4.8 – Stability, number of communities and variation of information as a function of the Markov time for $\alpha = 0.75$.

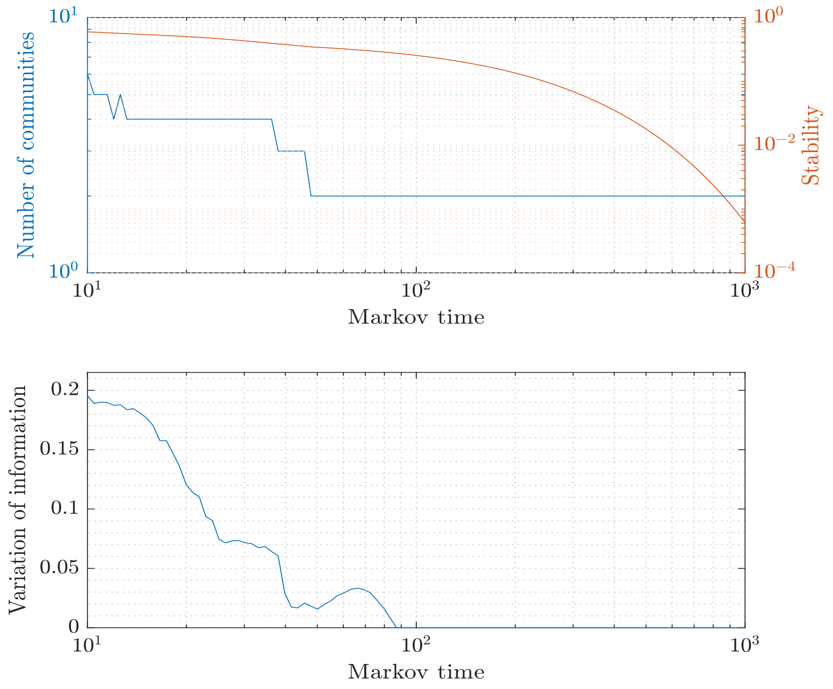

Figure 4.9 – Stability, number of communities and variation of information as a function of the Markov time for $\alpha = 0.5$.

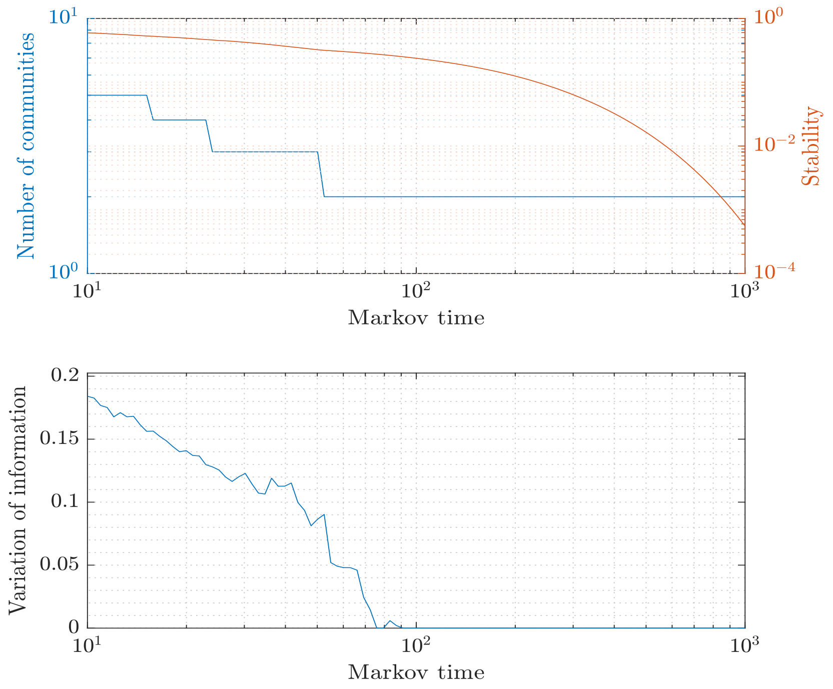

Figure 4.10 – Stability, number of communities and variation of information as a function of the Markov time for $\alpha = 0.25$.

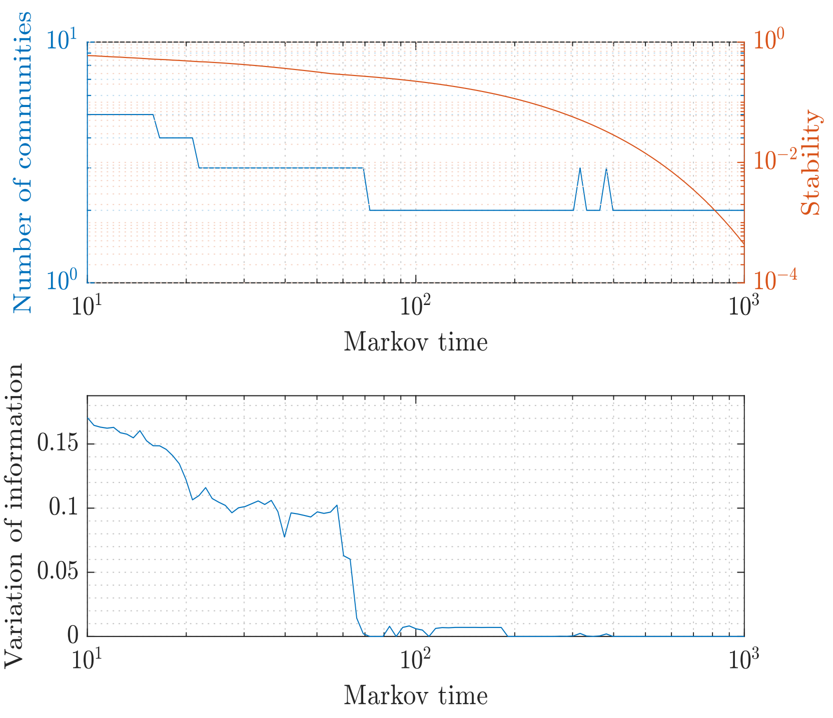

Figure 4.11 – Stability, number of communities and variation of information as a function of the Markov time for $\alpha = 0$.

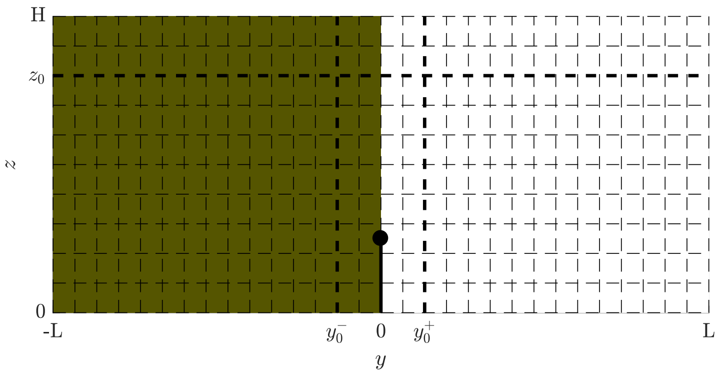

Figure 4.12 – Illustration of the two-communities partitioning for $\alpha = 0.25$.

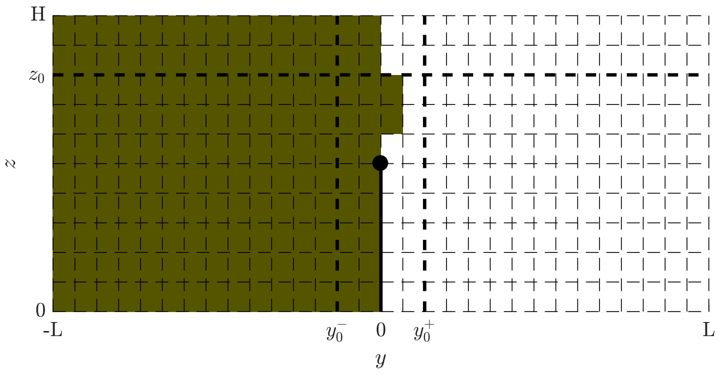

Figure 4.13 – Illustration of the two-communities partitioning for $\alpha = 0.5$.

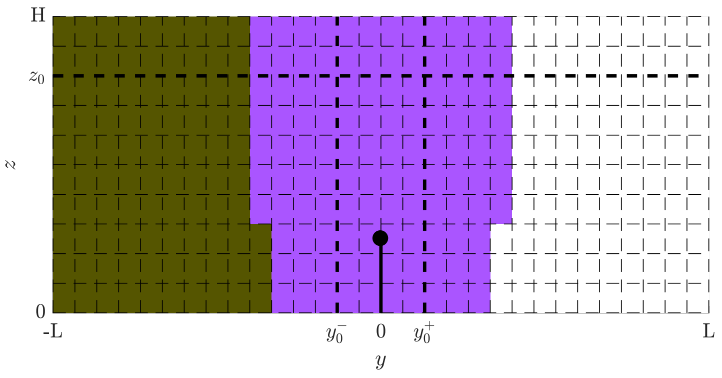

Figure 4.14 – Illustration of the three-communities partitioning for $\alpha = 0.25$.

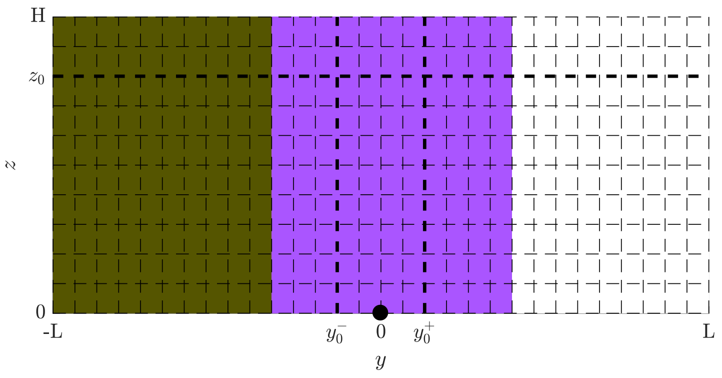

Figure 4.15 – Illustration of the three-communities partitioning for $\alpha = 0$.