NetBioDyn

Comment ça marche ?

Guide de l'utilisateur

2025

Rédigé par Morgane Dellozcour, étudiante de première année de master Biologie-Santé

Sous la direction de Michael Théron et Pascal Ballet

Sommaire

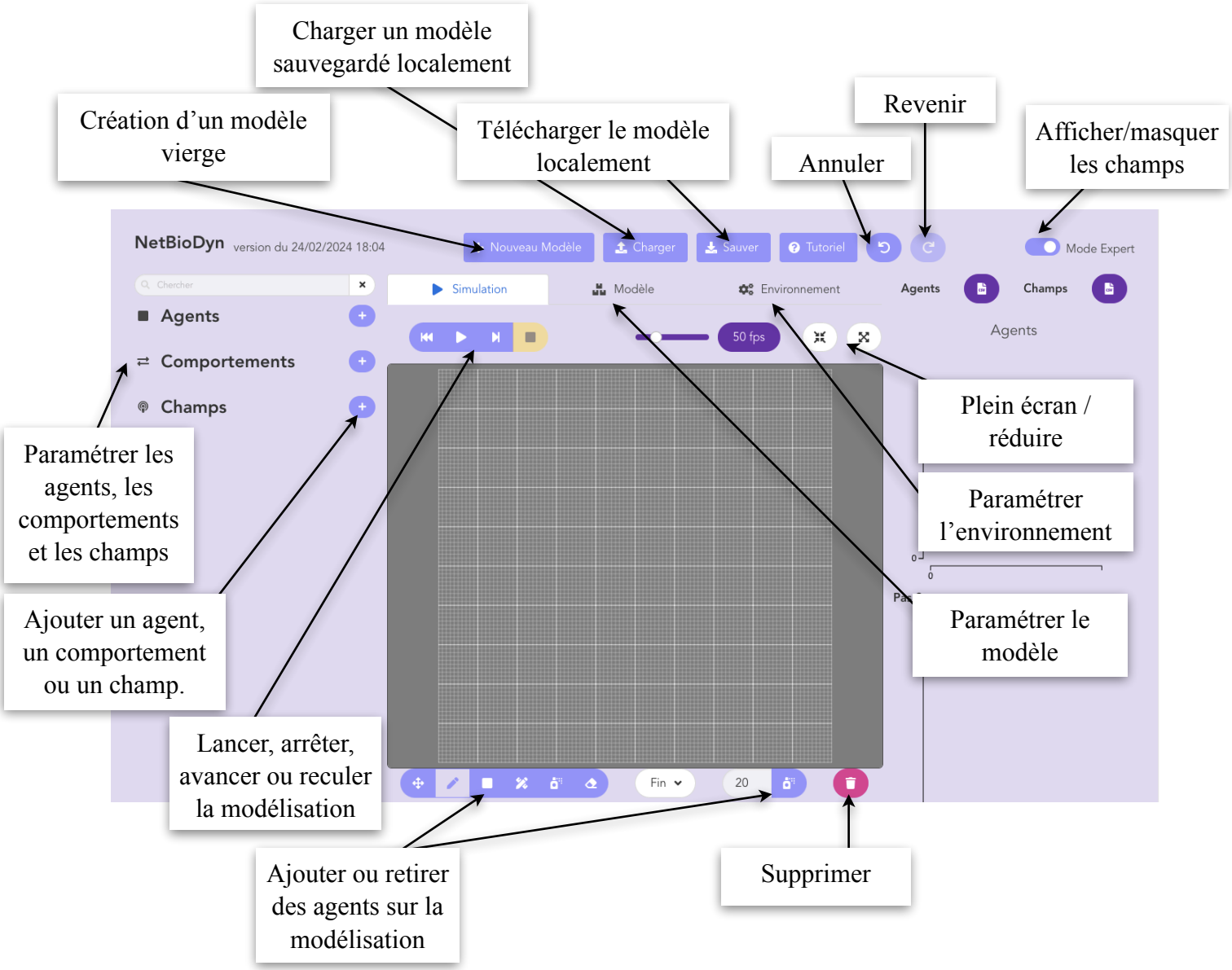
Introduction

NetBioDyn est un outil informatique qui permet de représenter, visualiser et simuler la dynamique des réseaux biologiques. La modélisation avec NetBioDyn se déroule sur un environnement discret. Elle est modélisée sur dans un cadre, sur une grille composée de petits carreaux. Chaque petit carreau est vide au début.

Ce cadre, c'est l'environnement dans lequel le modèle évoluera. Les paramètres de l'environnement peuvent être modifiés. On peut déterminer la taille de 10 à 100 (nombre de carreaux par côté du carré). On peut aussi définir une couleur de fond ou intégrer l'URL d'une image à mettre en arrière plan pour personnaliser ou rendre plus lisible le modèle. On peut aussi choisir de ne pas afficher la grille.

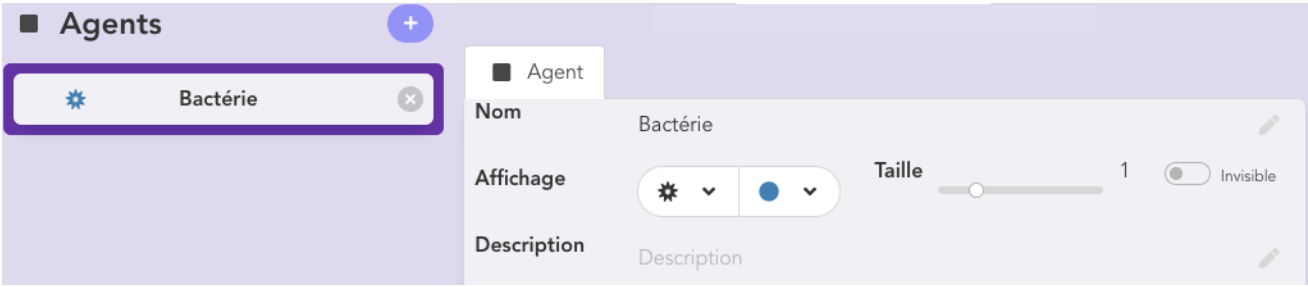
Sur NetBioDyn, on retrouve trois catégories de paramètres qui vont définir notre modèle : les agents, les comportements et les champs. Chacun paramètre aura son importance dans la modélisation.

La modélisation se déroule au fil du temps, temps que l'on appelle Pas. Un Pas correspondra à une unité de temps en fonction de la modélisation réalisée. Ainsi, on retrouve :

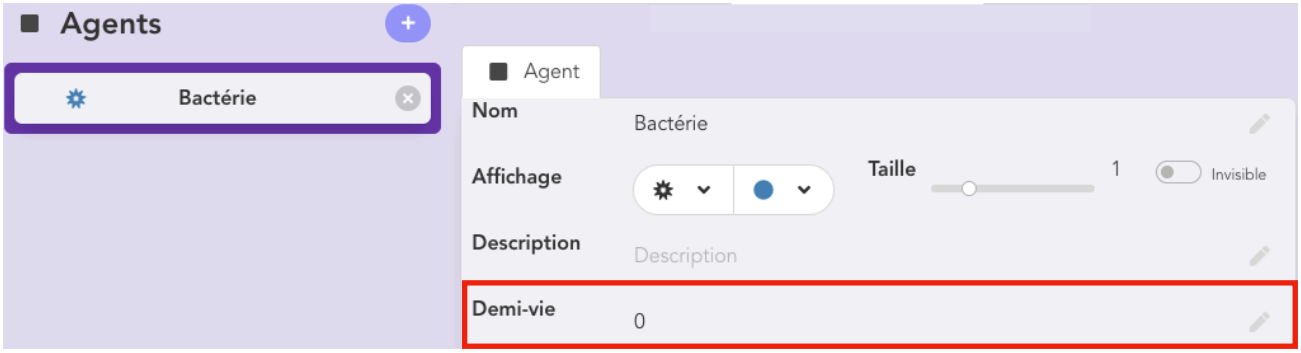


Les Agents

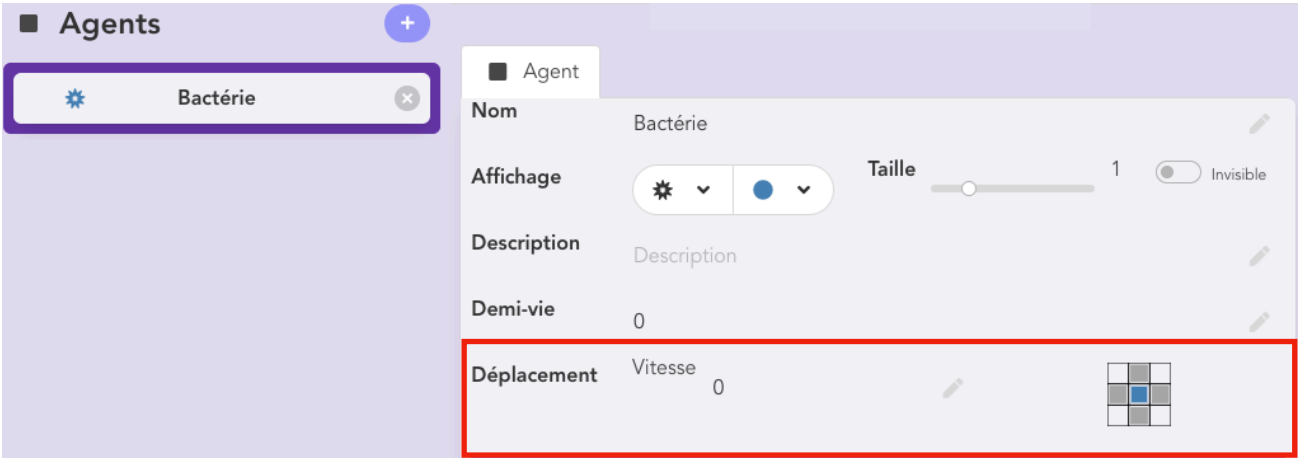
Les agents définissent les éléments qui composent votre modélisation, comme les bactéries dans une fonction bactérienne mais aussi les tissus et les défenses immunitaires. Les agents regroupent toutes les entités qui vont agir entre elles dans votre modèle, mais aussi les structures fixes par exemple.



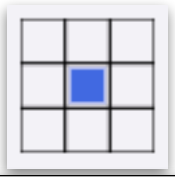
Chaque agent correspond à un type ou groupe d'éléments. Par exemple l'agent « bactérie » désignera l'ensemble des bactéries du modèle. On peut aussi être plus précis et créer un agent par espèce de bactérie. On peut lui choisir une icône et une couleur pour les identifier plus facilement dans la liste des agents (sur la modélisation, seule la couleur sera visible et non l'icône). La taille permet de rendre l'agent plus ou moins haut sur la vue 3D (et non plus large). L'onglet invisible permet de rendre l'agent invisible sur la modélisation.



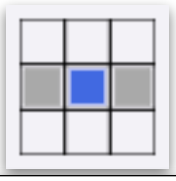
On peut donner à ces agents une demi-vie. Si on regarde pour un seul agent (individuellement), une fois la demi-vie passée, l'agent aura eu une chance sur deux de mourir, de disparaître. Il peut donc être toujours présent sur la modélisation mais il y a une chance sur deux qu'il ai disparu. Si on regarde le nombre total d'agent présents sur la modélisation, à la fin de la demi-vie, la moitié des agents auront disparus, seront morts. Si n est le nombre d'agents, alors il y a aura n/2 agents qui auront disparu.



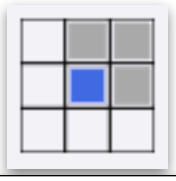
On lui accorde une vitesse entre 0 et 1, 1 étant la plus rapide. On peut aussi définir le sens de déplacement de l'agent en sélectionnant les cases souhaitées. Si par exemple, je veux modéliser des agents qui ensemble se déplacent selon un flux, je peux sélectionner uniquement les cases qui vont dans la direction souhaitée. Les agents pourront par exemple se déplacer vers la droite en haut, ou exclusivement vers la gauche. En fait, lorsqu'il se déplace, l'agent va choisir une case au hasard parmi les cases sélectionnées.



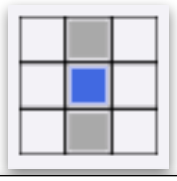
Aucun déplacement



Déplacement des deux côtés



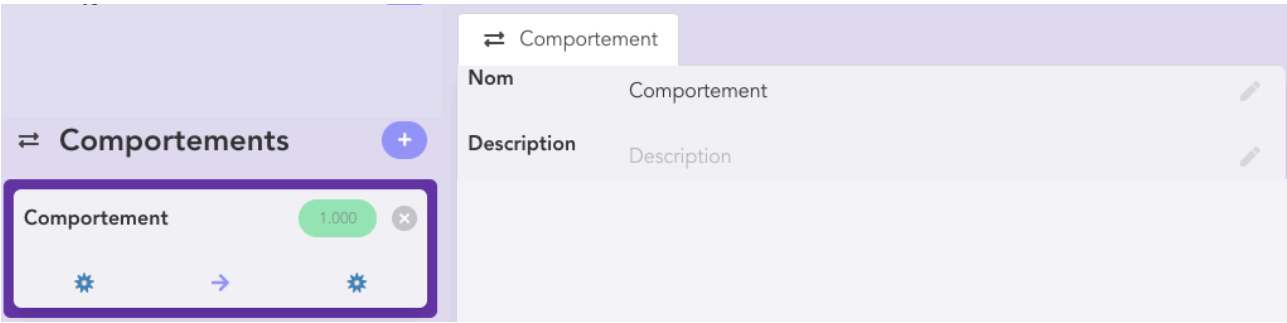
Déplacement en haut et à droite



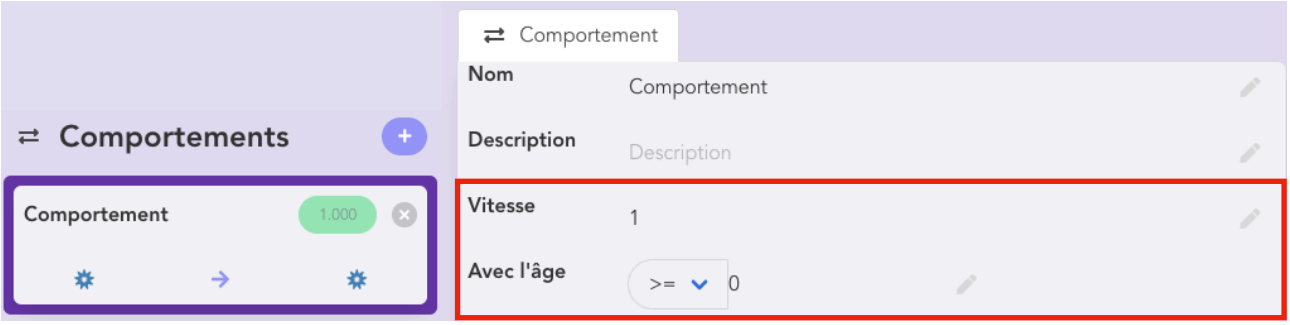
Déplacement en haut et en bas

Les Comportements

Les comportements définissent les interactions entre les agents. On peut ainsi modéliser une transformation, une production ou encore une absorption. Le comportement peut décrire la rencontre entre deux agents, ou un évènement qui survient à un moment donné.

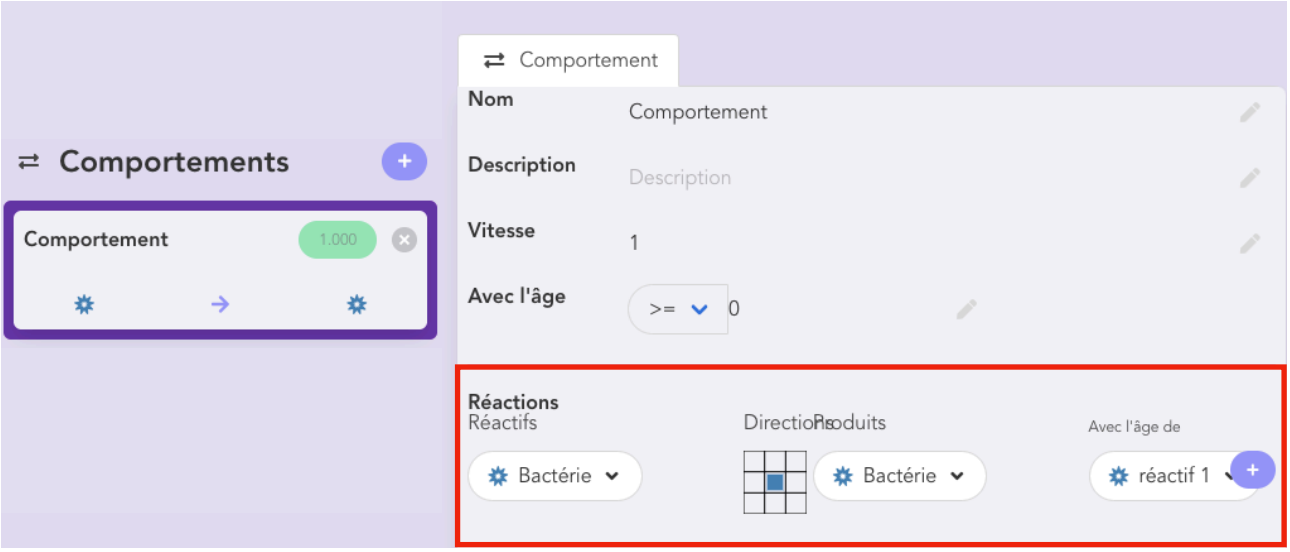


On peut ainsi nommer les comportements pour mieux les identifier, et en ajouter autant que l'on souhaite. On peut aussi y ajouter une description pour plus de précision. Par exemple, on crée un comportement appelé « Prolifération bactérienne », que l'on décrit par la multiplication des bactéries du modèle.



On peut définir une vitesse, qui correspond à la probabilité que le comportement, l'évènement, se produise. Si par exemple, on a 20% de chances que l'évènement arrive, on mettra une valeur de 0.2 pour la vitesse.
On peut ajouter une valeur dans la case « Avec l'âge », nous permettant de déterminer un âge que doivent avoir les agents pour que le comportement se réalise.

=	Egal, le comportement se déclenche à l'âge exact défini.
!=	Différent de (≠), le comportement peut avoir lieu à tous les âges de l'agent sauf celui défini.
<	Strictement inférieur, le comportement peut avoir lieu à tous les âges strictement inférieurs à celui défini.
<=	Inférieur ou égal (≤), le comportement peut avoir lieu à tous les âges inférieurs ou à l'âge égal à celui défini.
>	Strictement supérieur, le comportement peut avoir lieu à tous les âges strictement supérieurs à celui défini.
>=	Supérieur ou égal (≥), le comportement peut avoir lieu à tous les âges supérieurs ou à l'âge égal à celui défini.

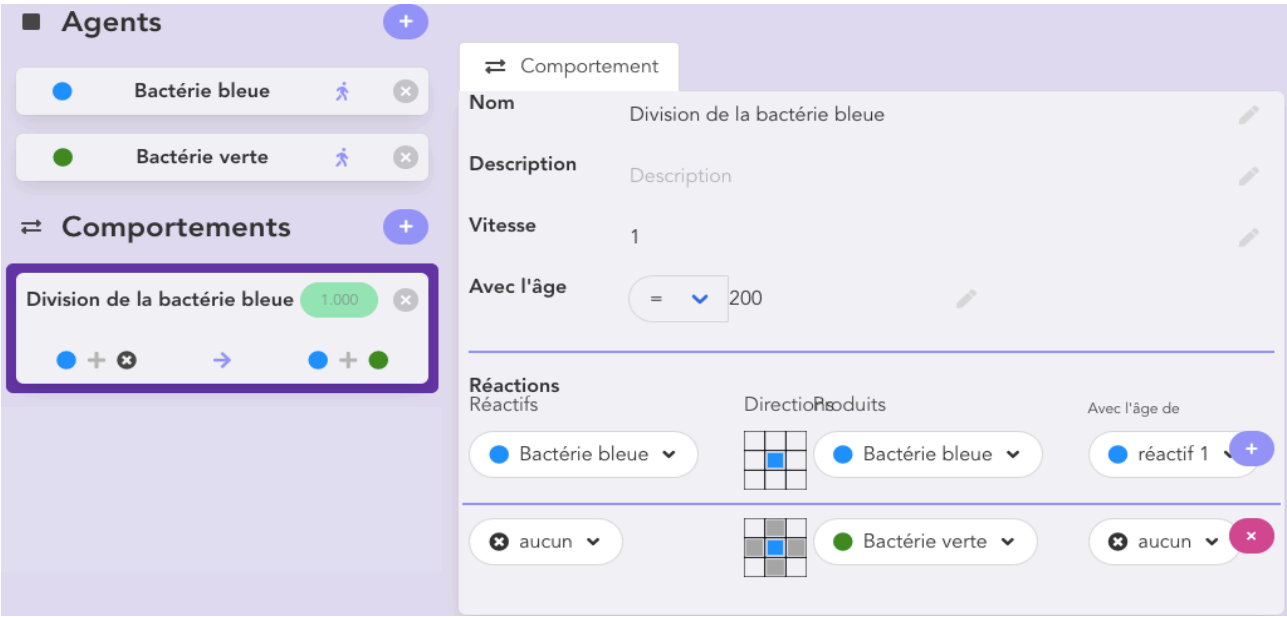


Enfin, on définit le comportement suivant plusieurs étapes :

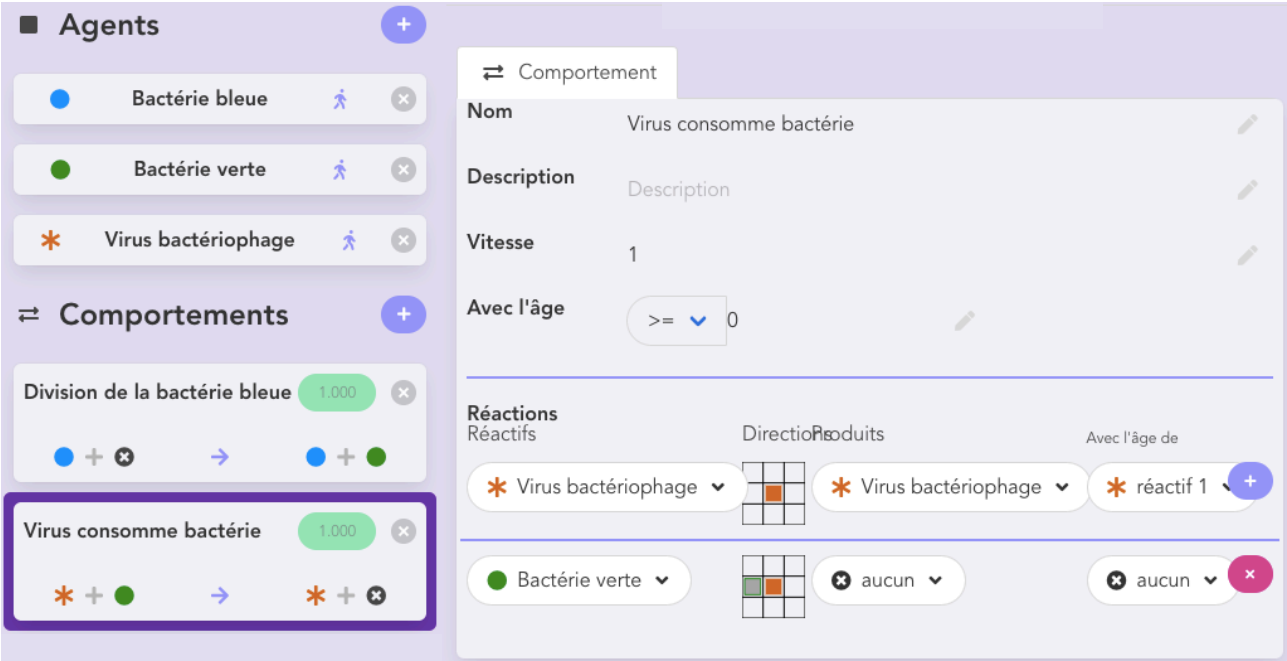
Etape 1 : Réactifs	**Etape 2 : Direction**	**Etape 3 : Produits**	**Etape 4 : avec l'âge de**
Ici, on détermine quel agent préexistant (qui est déjà placé ou présent sur la modélisation) va effectuer le comportement. On peut n'en choisir qu'un, par exemple une bactérie qui grandit à un âge précis. On peut aussi choisir deux agents pour modéliser leur rencontre dans le modèle, une collision. Lorsque ces deux agents se rencontrent, le comportement se produit.	Dans le cas d'un seul agent, elle permet de déterminer de quel coté le ou les produits seront formés. Dans le cas de 2 agents ou plus, elle permet de déterminer de quel côté les agents vont devoir se rencontrer pour que le comportement ai lieu.	Ici, on va déterminer les conséquences de la réaction. On peut par exemple déterminer qu'une bactérie va donner naissance à une nouvelle bactérie ou à deux bactéries, trois bactéries… On peut aussi déterminer que la rencontre entre 2 agents donne naissance à un nouvel agent, un produit.	Lorsque le produit apparait, on peut déterminer un âge qu'il aura à sa naissance. Par exemple, on modélise la transformation d'une bactérie en une autre. Celle-ci peut garder l'âge qu'elle avait au moment du comportement.

Exemples concrets :

On a une bactérie que l'on va appeler *bactérie bleue*. Au bout de 200 heures (pas), celle-ci se divise et forme une nouvelle bactérie, la *bactérie verte*. Ainsi, le comportement sera :



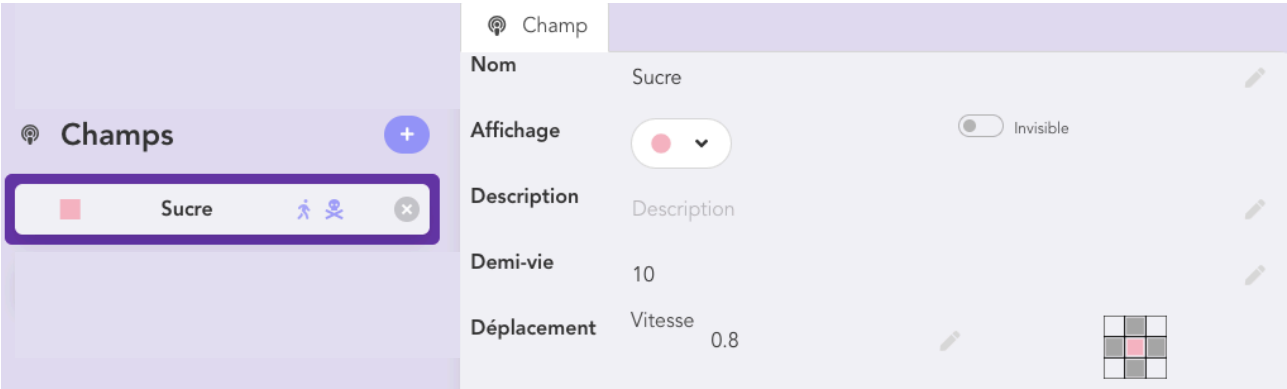
Maintenant, lorsque la *bactérie verte* se balade, elle rencontre un *virus bactériophage*, celui-ci va manger la bactérie. Mais, le virus ne peut avaler la bactérie que si celle-ci se présente sur sa gauche. Le comportement sera :



On peut ajouter autant de réactifs et de produits que l'on souhaite, par exemple 1 réactifs et 10 produits ou 10 réactifs et un seul produit.

Les Champs

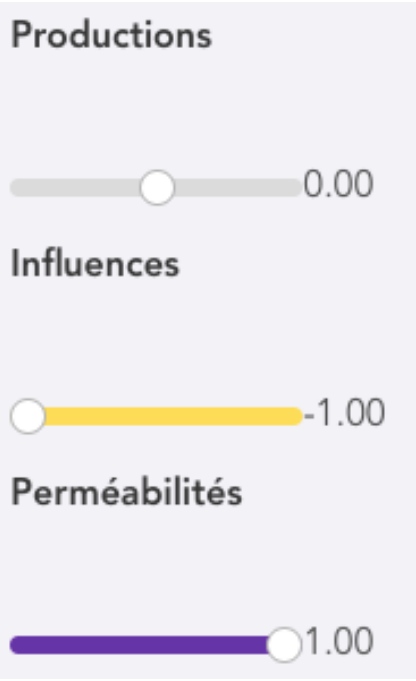
Les champs permettent de diffuser un élément dans le modèle. Les champs servent à influencer les agents ou représenter des gradients qui peuvent affecter le comportement des agents. On peut par exemple créer des gradients de concentration, de température ou d'autres paramètres. Ils peuvent aussi délimiter une zone dans laquelle les agents seront attirés ou au contraire repoussés. Ils peuvent aussi représenter des forces comme la gravité.



Comme pour les agents et les comportements, les champs peuvent être nommés et décrits. On peut leur attribuer une couleur, une demi-vie, une vitesse de déplacement et une direction. Ensuite, les réglages se font au niveau des agents et des comportements.

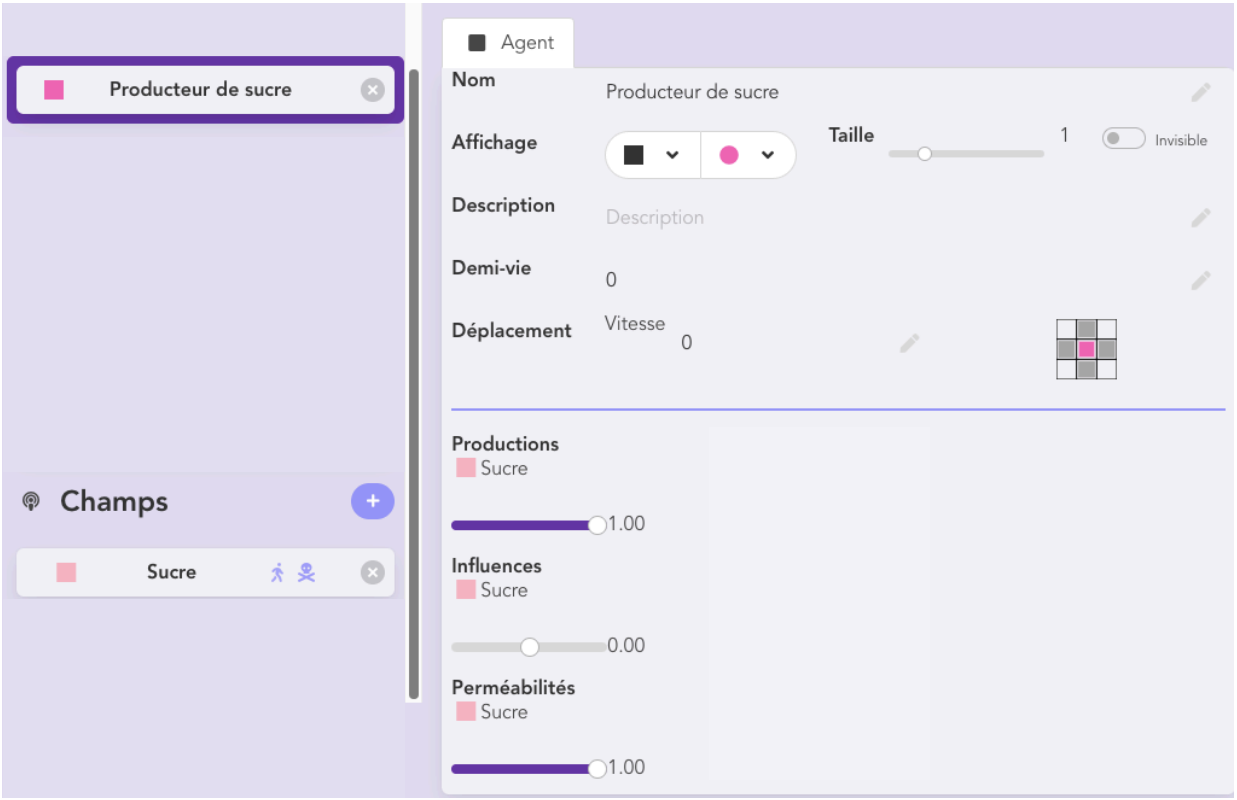
Trois paramètres au niveau des agents permettront son intéraction avec le champ.

Productions	Le paramètre production permet à l'agent de produire lui même le champ. Lorsque la production est à zéro pour un champ, l'agent ne produit pas ce champ. Lorsqu'elle est à 1.00, la production du champ par l'agent est au maximum.
Influences	Le paramètre influence permet à l'agent d'être plus ou moins attiré par le champ. Lorsque l'influence est à -1.00, le champ repousse l'agent. Lorsqu'elle est à 0, le champ n'a aucune influence sur l'agent. Lorsqu'elle est à 1.00, l'agent est attiré au maximum par le champ.
Perméabilité	Le paramètre perméabilité permet à l'agent de passer au travers du champ. Lorsque la perméabilité est à 0, le champ ne passe pas au travers de l'agent (permettant ainsi de délimiter une zone où le champ se répand). Lorsqu'elle est à 1, le champ passe au travers de l'agent.

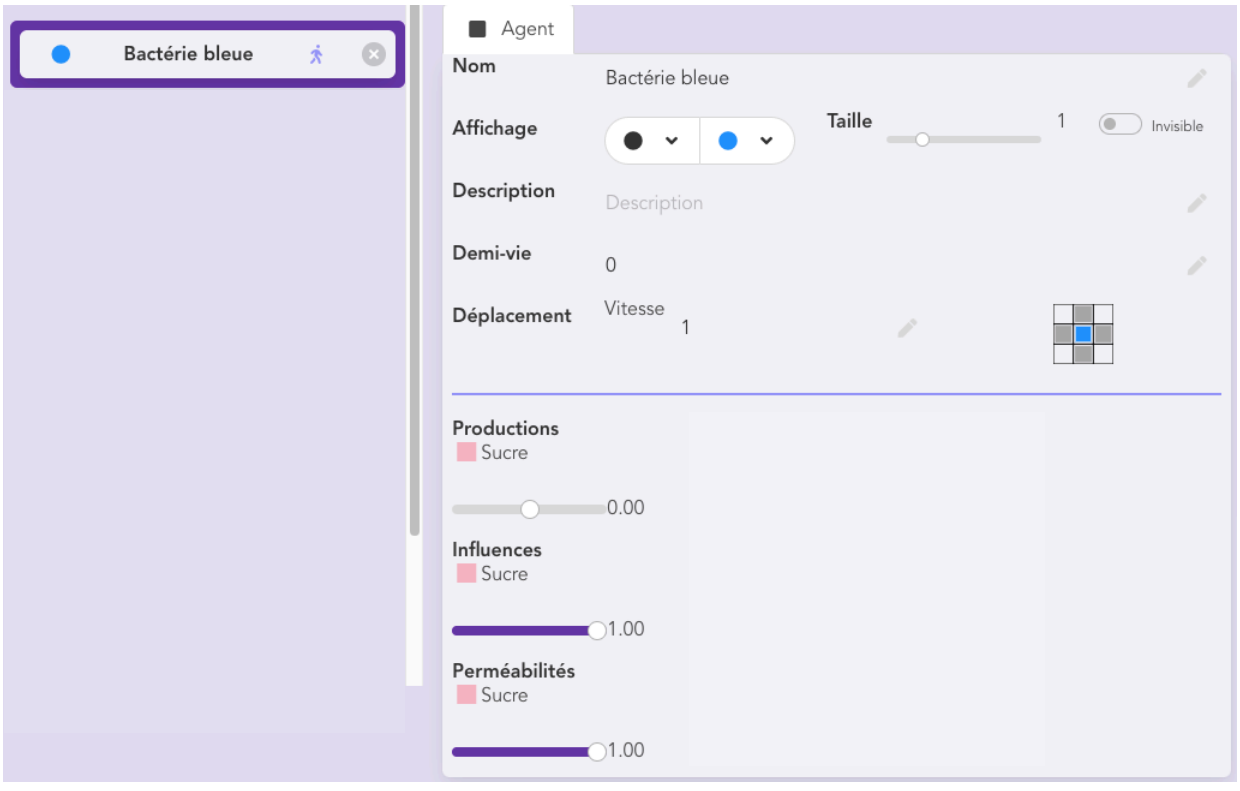


Exemples concrets :

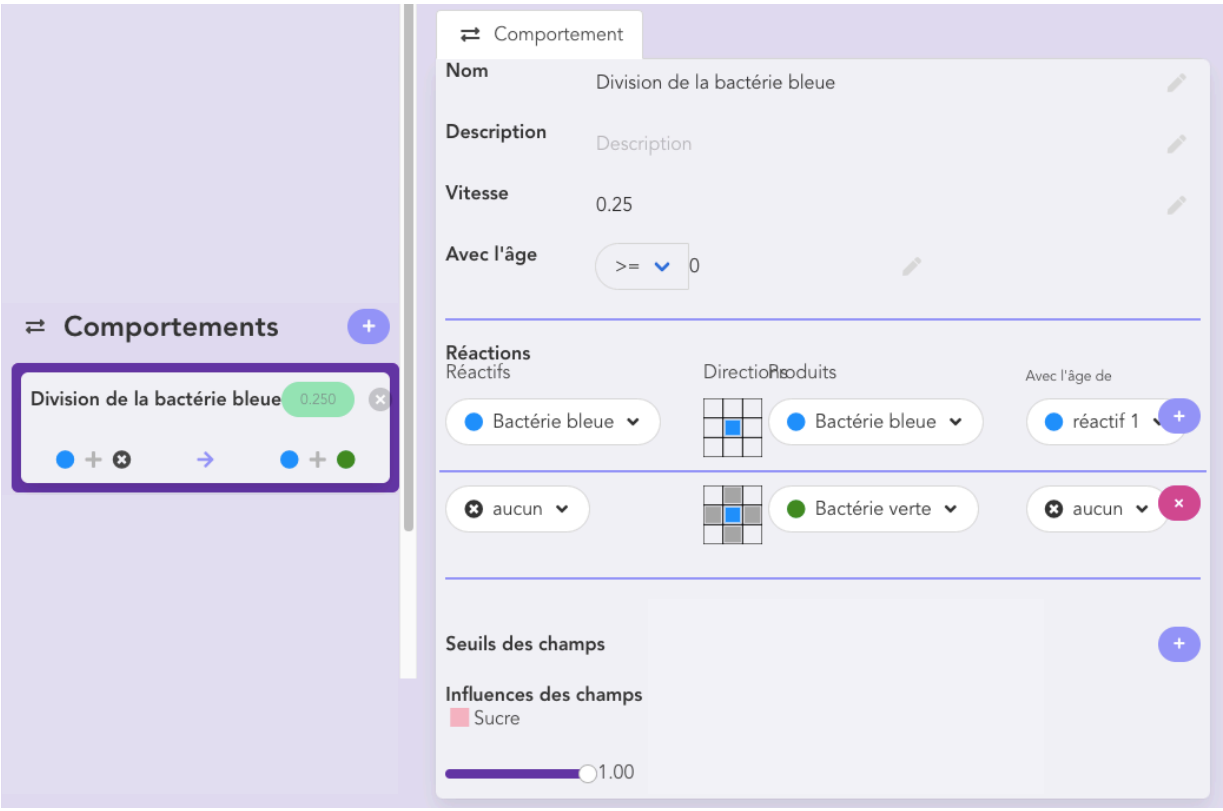
Reprenons les agents *bactérie bleue* et *bactérie verte*. On garde les comportements précédents. Pour se diviser et produire la *bactérie verte*, la *bactérie bleue* a besoin de sucre. On va donc créer l'agent *producteur de sucre* qui produira le champ *Sucre*. On met donc la production de *Sucre* à 1.00.



La *bactérie bleue* doit elle aussi être attirée par ce champ, par le sucre. On va donc mettre l'influence de *Sucre* sur l'agent *bactérie bleue* à 1.00.

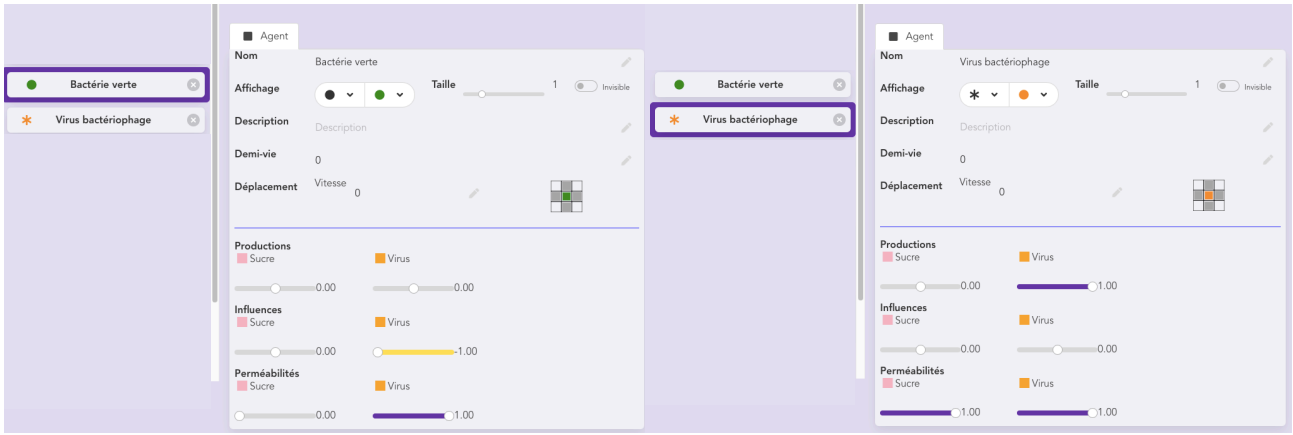


Si on veut faire en sorte que le sucre stimule la division de la bactérie bleue en une bactérie bleue et une bactérie verte, il faut aussi modifier le comportement. On va donc faire en sorte que le champ *Sucre* influe le comportement *Division de la bactérie bleue.*



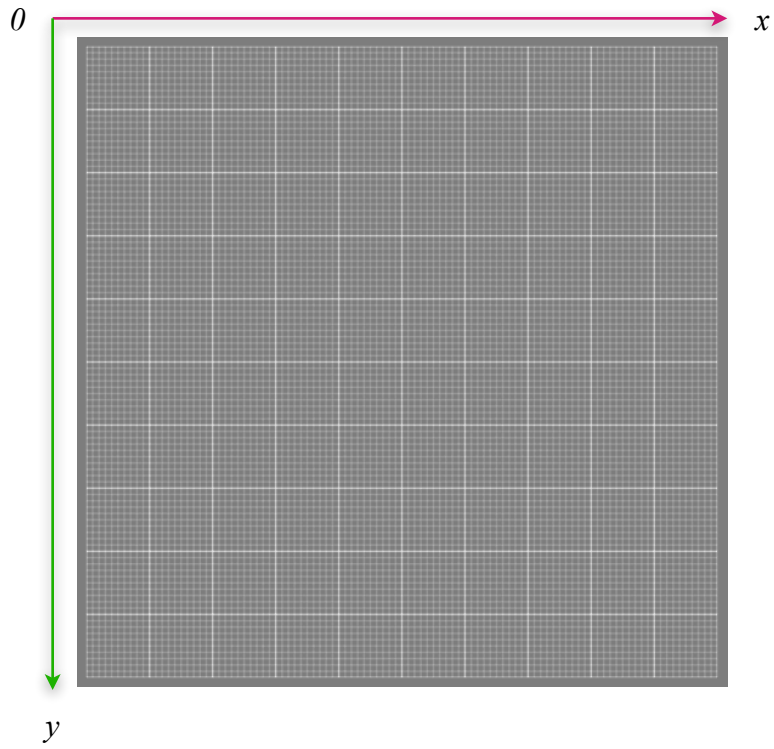
On a maintenant une *bactérie bleue*, attirée par le *Sucre* (produit par le *Producteur de sucre*), qui se divise mieux lorsqu'il consomme ce sucre et produit plus de *bactérie verte.*

Mais maintenant, je veux que cette *bactérie verte* soit repoussée par les *virus bactériophages* qui la dévorent. Je vais donc créer un champ, produit par l'agent *virus bactériophage*, qui repousse l'agent *bactérie verte*.



Je mets donc l'influence du champ *Virus* sur la *bactérie verte* au minimum, et la production du champ *Virus* par l'agent *virus bactériophage* au maximum.

La propagation du champ peut être définie grâce à une formule. On définit *y* l'axe des ordonnées, et *x* l'axe des abscisses. Cet axe commence à zéro et dépend de la taille de la grille : si la taille est de 100 x 100, les axes iront de 0 à 100.



Ainsi, plusieurs éléments peuvent entrer en jeu dans la définition de la formule.

Paramètres de la formule :

step	Représente le temps écoulé depuis le début de la simulation
x, y	Position de x et y dans la case courante. indispensables pour les modèles spatialisés où la valeur d'une cellule dépend de ses voisines ou de sa position.
current	Valeur courante , la valeur par défaut du champ, utilisée si aucune formule n'est définie ou si la formule ne peut être évalué.

Fonctions du modèle :

agent (index)	Permet d'accéder aux propriétés d'un agent spécifique en fonction de son index
filed0 (index), filed1 (index)	Permettent d'accéder aux valeurs précédentes d'un champ spécifique pour un agent donné.

Opérateurs et fonctions mathématiques logiques :

+, -, *, /		Opérateurs mathématiques de somme, soustraction, multiplication et division
%		Reste d'une division entière
Opérateurs logiques	**&&**	"Et", renvoie "vrai" si les deux conditions sont vraies
	||	"Oui", renvoie "vrai" si au moins une des deux conditions est vraie
	!	"Non", inverse la valeur booléenne
==, !=, <, >, <=, >=		Opérateurs de comparaison
a ? b : c		Opérateur ternaire, si a est vrai alors la valeur est b, sinon c
pi, e		Constantes π et $\mathcal{E}$
abs (x)		Valeur absolue
avg (a1, …, an)		Moyenne de n valeurs
cos(x), sin (x), tan(x)		Fonctions trigonométriques
Fonctions d'arrondis	**floor(x)**	Arrondi à l'entier inférieur le plus proche
	ceil (x)	Arrondi à l'entier supérieur le plus proche
	round (x)	Arrondi à l'entier le plus proche
ln (x), log(x, base)		Fonctions logarithmiques
min (a1, …, an) ; max(a1, …, an)		Fonctions minimum et maximum
sum (a1, …, an)		Fonctions de somme

Exemple concret : utilisation des formules de champs

Imaginons que l'on souhaite modéliser une marée montante et descendante grâce à un champ. Le champ montera et descendra en fonction de la marée. On va donc définir la formule *ch* pour le champ 1 :

$$ch_1 = \cos\left(\frac{y}{100}.\frac{\pi}{2}\right)$$

On a donc :

- y : Coordonnée verticale de la grille (axe des ordonnées).
- 100 : Sert à normaliser y car la grille fait environ 100 cases de hauteur. On a une valeur de y pour 100 cases, on passe à une valeur qui ne peut dépasser 1 (pour l'utilisation de la fonction cosinus).
- $\frac{\pi}{2}$: Facteur d'échelle pour que l'argument du cosinus varie entre 0 et $\frac{\pi}{2}$. Il représente l'angle entre l'axe des ordonnées (y) et l'axe des abscisses (x).
- cos(x) : Fonction cosinus, qui varie entre 1 et 0 sur l'intervalle $[0, \frac{\pi}{2}]$.

Le champ se déplace donc sur une intervalle compris entre 0 et 1 sur l'axe des ordonnées, 0 étant le bas de la grille et 1 le haut de la grille. Si l'on souhaite que le champ se déplace de gauche à droite, on remplace y par x (il se déplacera sur l'axe des abscisses).

On peut compléter la formule pour qu'elle dépende du pas et que le champ se déplace à une vitesse et une distance définie :

$$ch_1 = \cos\left(\frac{y}{100}.\frac{\pi}{2}\right) + \sin\left(\frac{step}{10}\right)$$

- step est le temps (nombre de pas dans la simulation).
- sin(x) oscille entre -1 et 1, donc ce terme fluctue périodiquement avec le temps.
- La division par 10 ralentit l'oscillation pour éviter qu'elle change trop vite. Si l'on modifie cette valeur, on modifie la vitesse d'oscillation du champ.

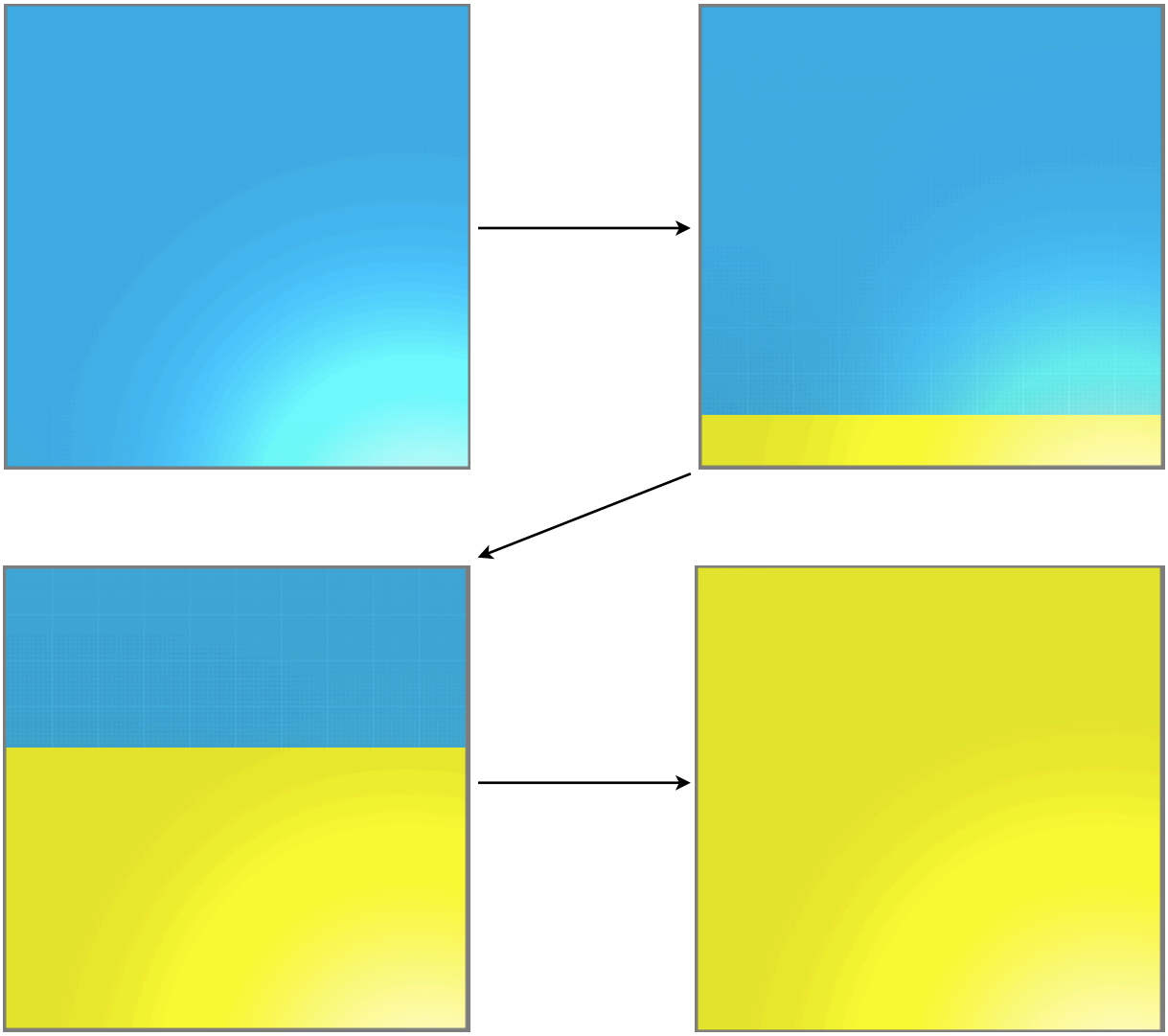
On peut ensuite ajouter un 2e champ, qui représentera le banc de sable. Ce champ sera découvert lorsque la marée sera basse (que le 1e champ sera haut) et sera réduit lorsque la marée sera haute (que le 1e champ sera bas). On peut utiliser la formule suivante pour le 2e champ :

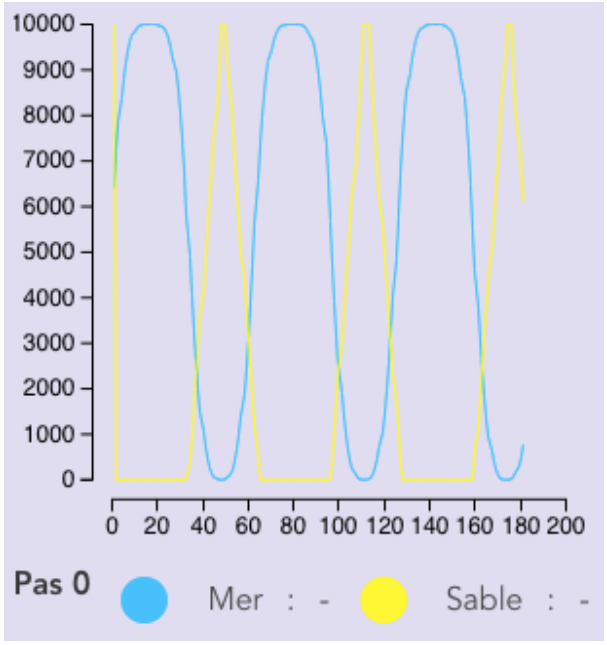
$$ch_2 = \text{field } 0\ (x, y) < 0.01\ ?\ 1 : 0$$

- field 0 (x, y) : Reprend les valeurs du 1e champ ch1
- < 0.01 ? 1 : 0 : Opérateur conditionnel *if… else*. Si les valeurs du champ 1 selon x et y sont inférieures à 0.01, alors la valeur du champ 2 est 1, sinon 0.

Ici, le champ 2 dépend du champ 1, et peut ainsi osciller de la même façon.

On obtient donc 2 champs qui oscillent en même temps, avec un champ *Mer* en bleu et un champ *Sable* en jaune :

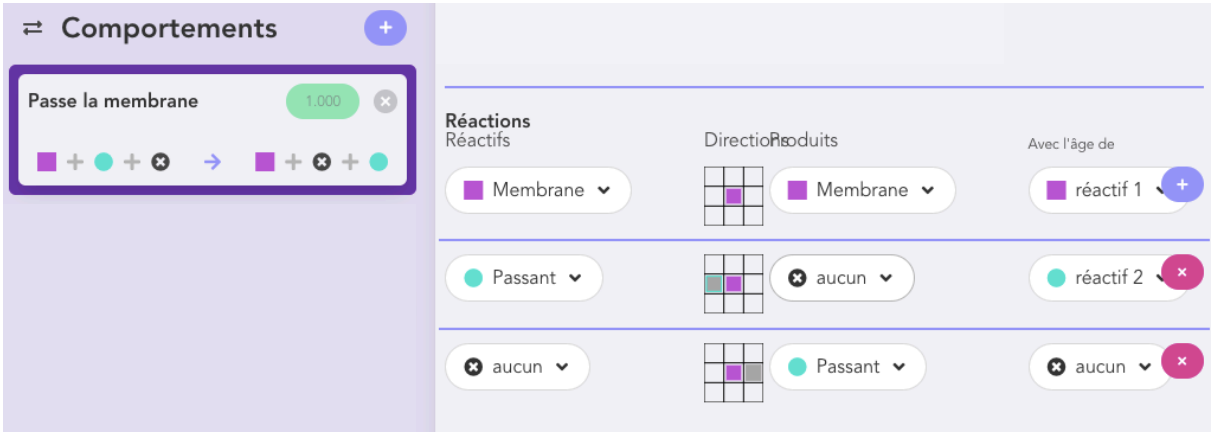




Astuces

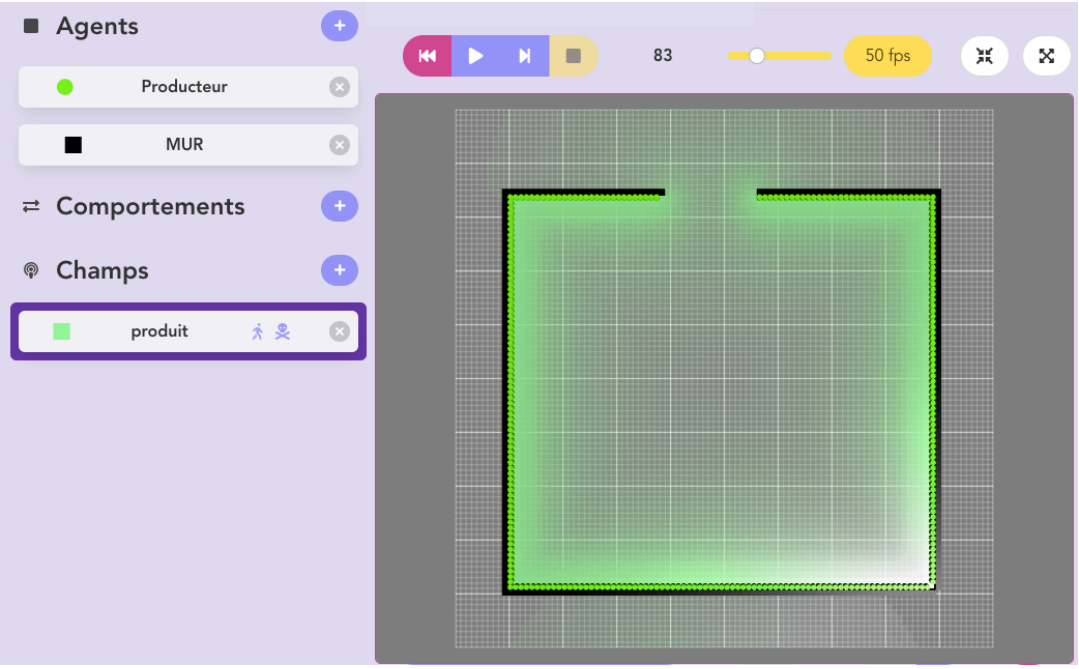
Créer une membrane sélective (un seul sens de passage) :

- Créer un agent *Membrane*, avec une vitesse et une demi-vie de 0. Créer une ligne droite sur la modélisation avec cet agent.
- Créer un champ *membrane,* produit par l'agent *Membrane*.
- Régler l'influence de ce champ pour chaque agent (on mettre l'influence à 1.00 pour un agent qui pourra traverser la membrane, et une influence à -1.00 pour ceux qui ne traversent pas).
- Créer un comportement, avec l'agent qui traverse que l'on appellera *Passant* :



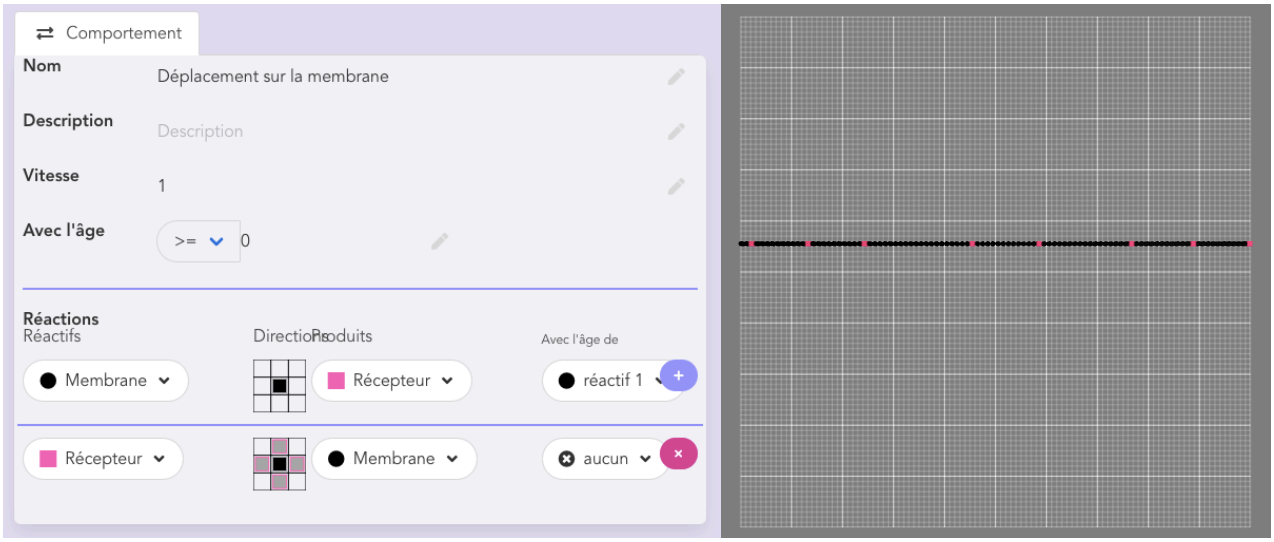
Créer une zone fermée pour un champ (le champ ne s'étend que dans ce compartiment) :

- Créer un agent qui produira le champ *Producteur*
- Créer un champ produit par l'agent *produit*
- Créer un agent qui ne laissera pas passer le champ *Mur*
- Régler la production par *Producteur* de *produit* à 1.00 et la perméabilité de *Mur* à *produit* à 0.
- Tracer le contour du compartiment avec le *Mur*, puis placer l'agent *Producteur* dans le compartiment. Le contour de celui-ci doit être continu et imperméable si on ne veut pas que le champ sorte du compartiment. On peut créer une ouverture qui permettra de laisser passer les agents, mais attention à l'encombrement.



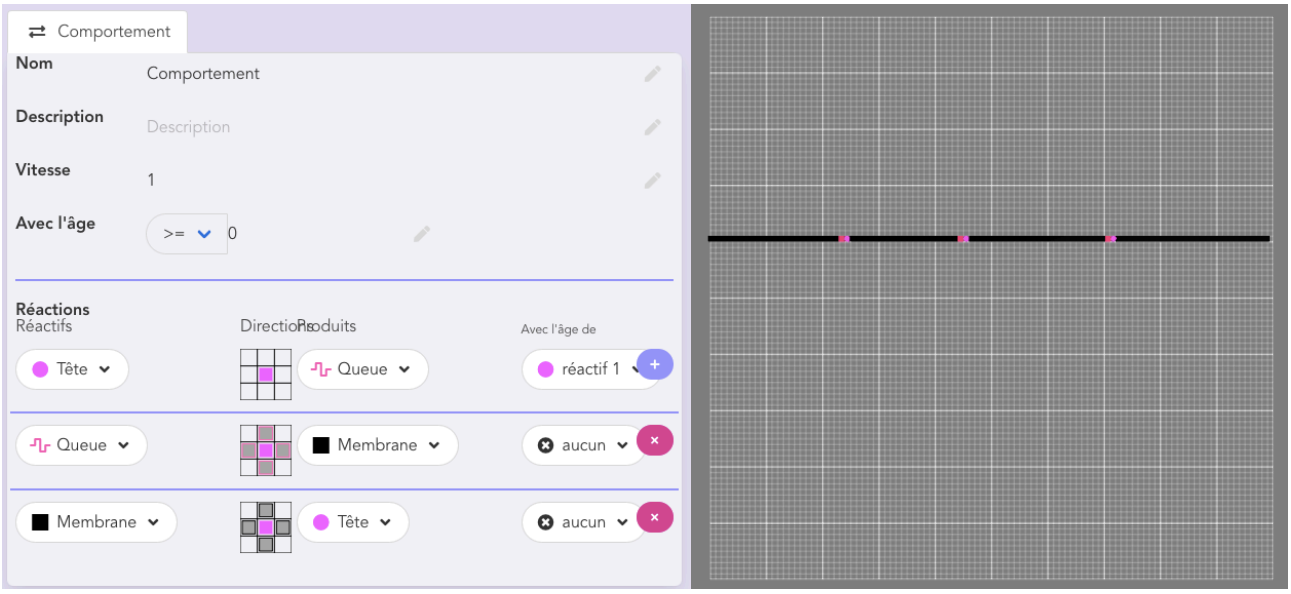
Créer une membrane sur laquelle des récepteurs se déplacent :

- Créer un agent *Membrane*, avec une vitesse de déplacement nulle.
- Créer un agent *Récepteur*, avec une vitesse de déplacement nulle.
- Créer un comportement *Déplacement sur la membrane.*
- Modifier la vitesse du comportement pour modifier la vitesse de déplacement des récepteurs sur la membrane, et la direction pour le sens de déplacement des récepteurs.



Modéliser la transduction d'un signal :

- Créer un agent *Membrane*, avec une vitesse de déplacement nulle
- Créer un agent *Tête*, avec une vitesse de déplacement nulle
- Créer un agent *Queue*, avec une vitesse de déplacement nulle
- Créer un comportement *Transduction du signal*.
- Modifier la vitesse du comportement pour modifier la vitesse de déplacement du signal sur la membrane.



Cas concret : infection tissulaire

Ici, on souhaite modéliser une infection dans un tissu quelconque. Cette infection produit une inflammation, et peut être éliminée par des macrophages contenus dans des ganglions.

1. Les agents

On a cinq agents dans la modélisation. On a l'agent *macrophage*, une cellule des défenses immunitaires. L'agent *infection* représente une infection quelconque. Ce modèle se passe au sein de *tissus*, des tissus vivants, qui font ici office de barrière mécanique. Les *ganglions* contiennent les macrophages, et sont entourés d'une *paroi ganglion* qui ici aussi fait office de barrière.

Nom de l'agent	Couleur	Déplacement
Macrophage		1
Infection		0
Tissus		0
Ganglion		0
Paroi ganglion		0

2. Les comportements

On a un seul comportement ici, le comportement *Stop infection*. Lorsqu'un macrophage rencontre un élément de l'infection, de n'importe quel côté, il a 10% de chance de l'éliminer.

Nom du comportement	Vitesse	Réactifs	Trajectoire	Produits
Stop infection	0.1	Macrophage		Macrophage
		Infection		

3. Les champs

Il y a deux champs dans ce modèle. Le premier champ *infection* est produit par l'infection en grande quantité et va attirer les macrophages pour réaliser le comportement décrit. Ce champ ne traverse pas les tissus ni la paroi des ganglions, ainsi il est forcé d'emprunter les passages ouverts des tissus pour s'épandre. Ainsi, les ganglions peuvent suivre ce même chemin en remontant les gradients et atteindre l'infection, sans rester coincé dans les tissus, pour éliminer l'infection. Une fois l'infection éradiquée, les macrophages doivent retourner dans les ganglions. Il y a pour cela un champ *Gg*, produit par les ganglions. Ce champ ne traverse ni les parois des ganglions, ni les tissus. Il est produit en plus faible quantité et a moins d'influence sur les macrophages. Ainsi, en présence des deux champs, les macrophages sont plus attirés par l'inflammation, ce qui va permettre leur sortie des ganglions pour éliminer l'infection. Une fois l'infection disparue, le champ inflammation disparait à son tour. Les macrophages ne sont donc plus influencés par ce champ, le champ Gg reprend le dessus et attire de nouveau les macrophages dans les ganglions.

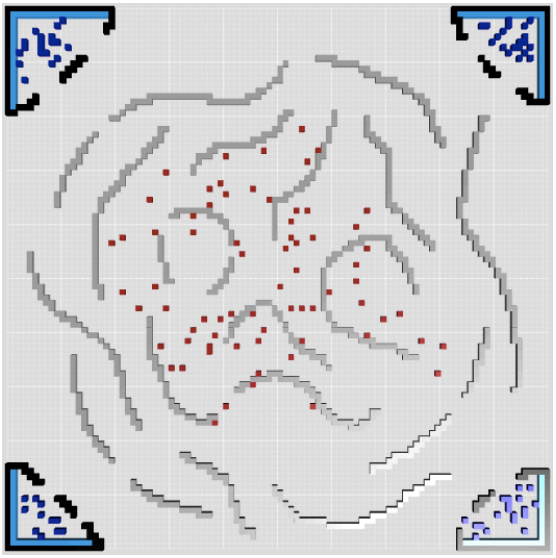
Nom du champ	Couleur	Demi-vie	Déplacement	Produit par	Effet
Inflam-mation		10	0.8	Infection	L'*infection* produit fortement un champ i*nflammation*, qui va attirer les *macrophages*, qui pourront éliminer détruire l'*infection*.
Gg		10	0.8	Ganglion	Ce champ est produit en petite quantité par les *ganglions*, et attire peu les macrophages. Lorsqu'il n'y a plus d'*infection*, et que le champ *inflammation* se dissipe, les macrophages sont attirés par cet autre champ et retournent dans les ganglions.

On adapte aussi en fonction des agents, qui vont produire, être influencés et/ou perméables aux champs.

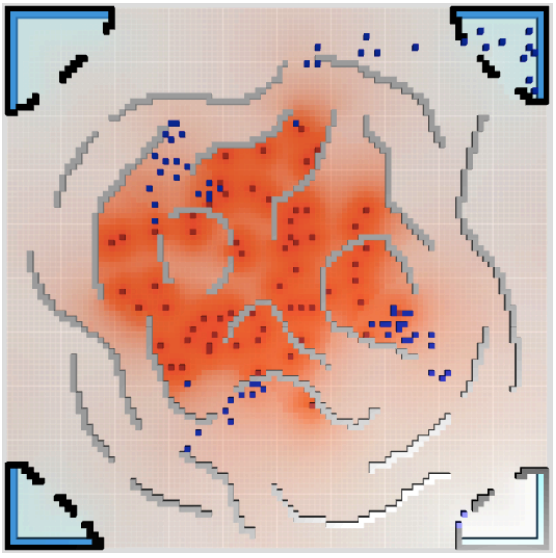
Nom de l'agent	Inflammation			Gg		
	Productions	Influences	Perméabilités	Productions	Influences	Perméabilités
Macrophage	0	1	1	0	0,25	1
Infection	1	0	1	0	0	1
Tissus	0	0	0	0	0	0
Ganglion	0	0	0	0,5	0	1
Paroi ganglion	0	0	0	0	0	0

4. Evolution du modèle et intérêt

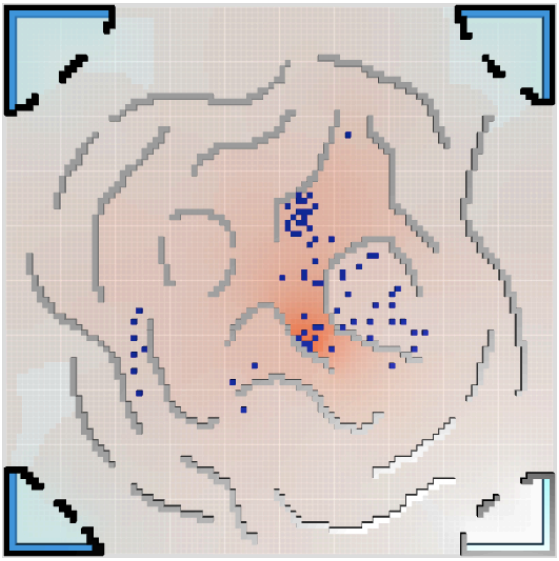
On observe que lorsque l'infection est peu étendue, elle recrute moins de macrophages. Ce petit nombre de macrophages parvient tout de même à l'éliminer. Lorsque l'infection est plus importante, ce sont presque tous les macrophages qui sont recrutés. A chaque fois que l'infection est éradiquée et qu'il n'y a plus d'inflammation, les macrophages retournent en quasi-totalité dans les ganglions.



Arrivée de l'infection
step = 0



Propagation de l'inflammation et recrutement des macrophages
step = 350



Elimination de l'infection et diminution de l'inflammation
step = 730

Retour des macrophages dans les ganglions
step = 1500

Cas concret : écosystème marin

Ici, on souhaite modéliser une population de poissons vivant dans l'océan. On souhaite que cette population varie, avec des poissons qui se reproduisent et d'autres qui meurent. On souhaite éviter que la population s'effondre et disparaisse, mais on ne veut pas non plus que celle-ci explose. On va donc mettre en place le modèle suivant.

1. Les agents

On a sept agents dans la modélisation. On a d'abord un agent *sol*, qui représente le sol du fond marin. Cet agent n'entre pas en compte dans les comportements. L'eau est délimitée par l'agent *Surface eau*, qui indique la séparation entre la mer et l'air. Ces deux agents ne bougent pas (aucun déplacement) et "posent" le décor de la simulation. L'agent *CO2* se dissout de la surface de l'eau. Ensuite, sur le sol, on retrouve des *racines*, les racines des algues. Ces racines vont faire pousser des *algues* qui elles mêmes feront pousser des *fruits*, grâce au CO2. Les fruits sont consommés par les *poissons* (du genre poisson-chèvre), ce qui permet leur reproduction. Les fruits correspondent aux vésicules de l'algue du genre *Codium*. On met donc en place les agents avec des paramètres suivants :

Nom de l'agent	Couleur	Demi-vie	Déplacement
Racine		0	0
Algue		0	0
Sol		0	0
Fruit		0	0
CO2		300	1
Surface Eau		0	0
Poisson		150	0.7

2. Les comportements

On retrouve ici sept comportements. Le premier est la *Naissance de l'algue* : lorsque le CO2 (qui se déplace et a une demi-vie) rencontre une racine, le CO2 est absorbé par celle-ci et elle produit une algue (au dessus ou sur le côte de la racine). Ensuite, on a la *Croissance de l'algue* : lorsque le CO2 rencontre une algue, celle-ci produit un nouvel agent *algue*, elle grandit (au dessus ou sur le côte de l'algue). Le CO2 est produit grâce au comportement *Dissolution du CO2* : avec une chance de 1%, un agent *Surface eau* produira en plus un agent *CO2* vers le bas uniquement, donc dans l'eau uniquement et pas dans l'air. Le comportement *Mélange CO2* permet au CO2 de se déplacer vers le bas avec 20% de chance (pour ne pas que tout le CO2 chute d'un coup vers le bas). On met ce comportement plutôt qu'un déplacement de l'agent directement pour que les agents ne se déplacent pas de façon aléatoire et trop rapidement dans le modèle, ce qui pourrait nuire à la simulation. Le comportement *Apparition fruit* transforme 0.1% des algues en fruit. Le comportement *Reproduction poisson* permet au poisson de se reproduire lorsqu'il rencontre un fruit : le poisson absorbe le fruit, le consomme, et donne un poisson de plus. Enfin, le comportement *Algue Tombe* sert lorsqu'une algue n'est pas rattachée à une racine : l'agent *Algue* est sédimenté jusqu'à une racine, vers le bas uniquement.

Nom du comportement	Vitesse	Réactifs	Trajectoire	Produits
Naissance Algue	1	Racine		Racine
		CO2		Algue
Croissance algue	1	Algue		Algue
		CO2		Algue
Dissolution CO2	0.01	Surface Eau		Surface Eau
				CO2
Mélange CO2	0.2	CO2		
				CO2
Apparition fruit	0.001	Algue		Fruit
Reproduction poisson	1	Poisson		Poisson
		Fruit		Poisson
Algue tombe	1	Algue		
				
				Algue
				

3. Les champs

Il y a trois champs dans la simulation. Un champ *Odeur* est produit par les fruits pour attirer les poissons et que ceux-ci mangent les fruits. Il se déplace à une vitesse de 0.8 et ont une demi-vie de 10, ce qui permet d'éviter que le champ se déplace trop vite et qu'il soit produit en trop grande quantité. Un champ *Trajectoire* produit par les poissons permet de voir leur trajectoire et de repérer leur déplacement. Il a une demi-vie de 1 pas pour ne pas qu'il encombre trop le modèle et aucune vitesse de déplacement pour rester fidèle au déplacement des poissons. Enfin, le champ *Algue* est produit par les algues et sert à mieux les visualiser.

Nom du champ	Couleur	Demi-vie	Déplacement	Produit par	Effet
Odeur		10	0.8	Fruit	Une odeur est produite par le fruit des algues, et va attirer les poissons pour qu'ils consomment le fruit et se reproduisent (comportement *Reproduction poisson*)
Trajectoire		1	0	Poisson	Ce champ montre la trajectoire des poissons au cours de la simulation, pour observer leur déplacement
Algue		2	0.01	Algue	Ce champ est surtout décoratif, pour mieux visualiser l'agent *Algue*.

On adapte aussi en fonction des agents, qui vont produire, être influencés et/ou perméables aux champs.

Nom de l'agent	Odeur			Trajectoire			Algue		
	Production	Influence	Perméabilité	Production	Influence	Perméabilité	Production	Influence	Perméabilité
Racine	0	0	0	0	0	1	0	0	1
Algue	0	0	0	0	0	1	1	0	1
Sol	0	0	0	0	0	1	0	0	1
Fruit	1	0	1	0	0	1	0	0	1
CO2	0	0	1	0	0	1	0	0	1
Surface Eau	0	0	0	0	0	1	0	0	1
Poisson	0	0.5	1	1	0	1	0	0	1

4. Evolution du modèle et intérêt

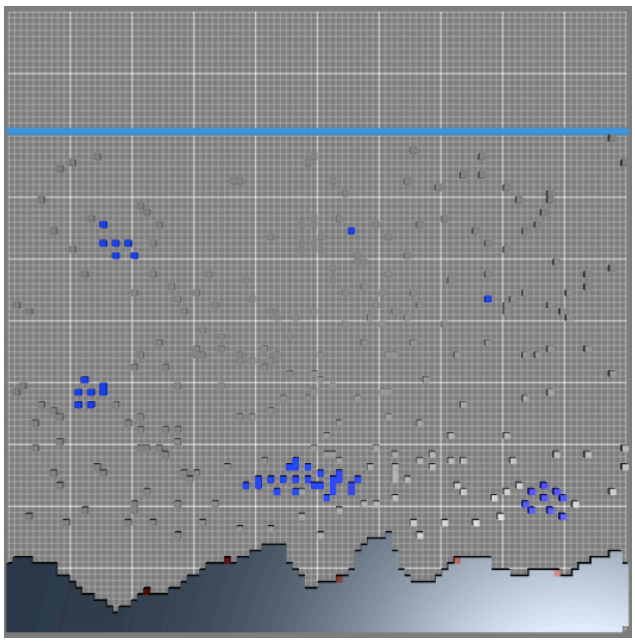
Au début du modèle, on a environ 235 agents CO2, 50 poissons et 5 racines. Il n'y a aucune algue ni aucun fruit. Le CO2 est produit à partir de la surface de l'eau et sa quantité augmente. Lorsqu'il rencontre les racines, celles-ci donnent une algue. Lorsque l'algue est en contact avec le CO2, celle-ci grandit et pousse. Ces algues donnent des fruits qui vont produire une odeur. Les poissons, qui se déplacent librement sous la surface de l'eau, sont attirés par l'odeur vers les fruits. Ils vont alors consommer ces fruits et se multiplier.

On observe d'abord une grande variation en nombre des agents. le nombre de poissons diminue fortement tandis que celui des fruits augmente. Lorsque le nombre de fruits attend environ 1à fruits, au pas 450, il y a plus de fruits que de poissons. Au pas 1000 environ, les courbes s'inversent : il y plus de poissons que de fruits. Lorsqu'on atteint environ 6à poissons et moins d'une dizaine de fruits, le modèle se stabilise. Le nombre de poissons oscille entre 40 et 60 et le nombre de fruits entre 0 et 10. La population de poissons ne diminue pas et n'augmente plus.

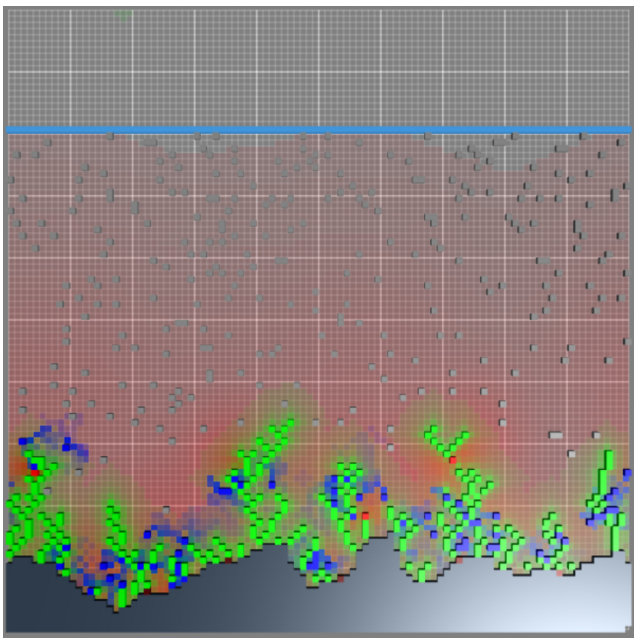
En fait, le nombre de poissons dépend du nombre de fruits. Les poissons ayant une demi-vie de 150, leur nombre diminue de moitié tous les 150 pas. Lorsqu'ils consomment des fruits, ils peuvent se reproduire et leur nombre augmenter. S'il n'y avait pas de fruit, la population n'augmenterait pas et finirait par mourir en intégralité. Ils peuvent donc se multiplier tant qu'il y a des fruits, mais puisqu'ils les consomme, le nombre de fruits n'augmente jamais beaucoup. Plus il y a de fruits, plus il y a de poissons, mais lorsqu'il n'y a plus de fruits, les poissons ne se multiplient pas. Si le nombre de fruits augmente trop rapidement, la croissance de la population de poissons ne s'arrêterait pas.

En fait, l'agent *Fruit* et les comportements associés sont limitants et permettent la présence d'une boucle rétro-active négative qui limite et contrôle la population de poissons. Ainsi, la population ne meurt pas en totalité, mais ne grandit pas de façon exponentielle évitant ainsi de bloquer le modèle.

Modèle au pas 0

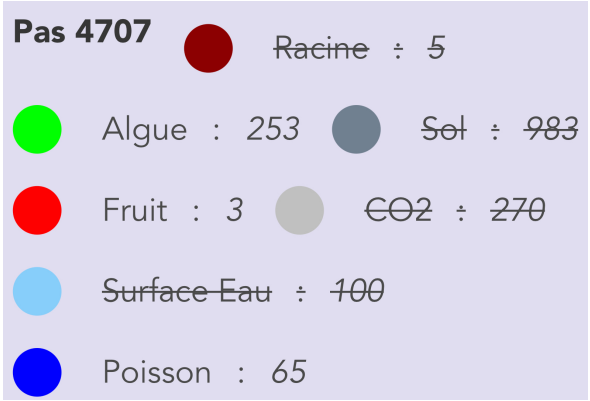


Modèle au pas 4709

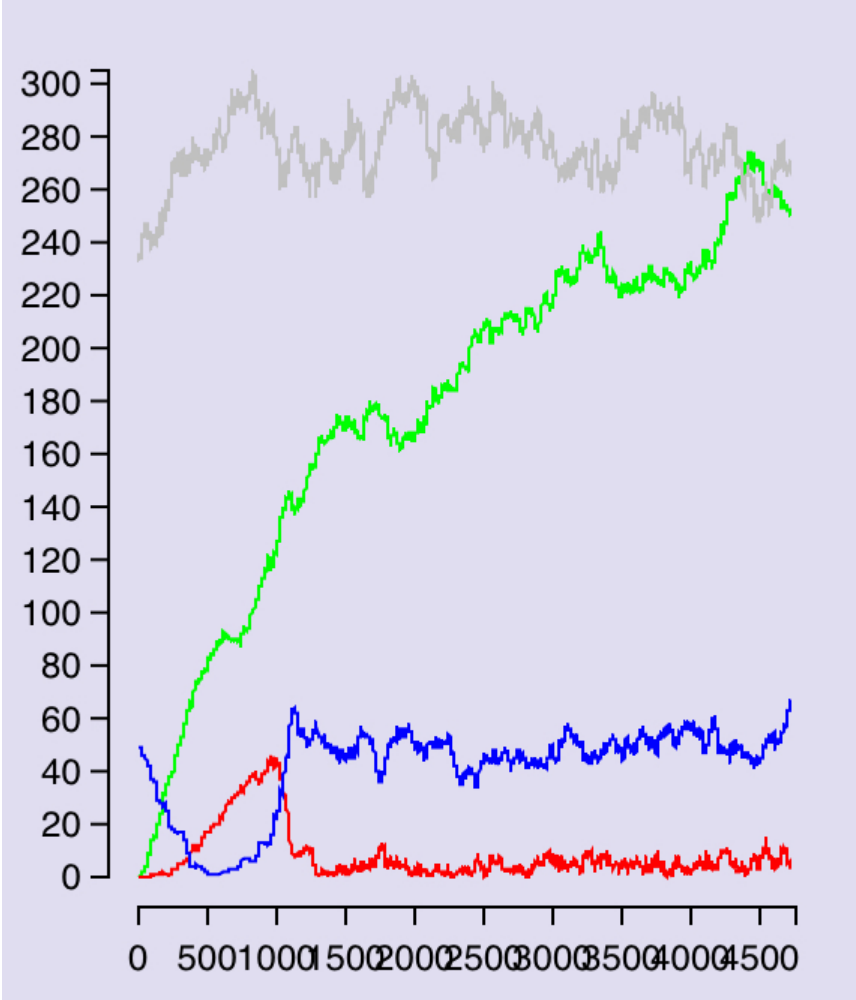


Nombre d'agents au début de la simulation

Nombre d'agents à la fin



Evolution du modèle



Bibliographie

- Ballet, P. et al. (2017). Modelling and Simulating Complex Systems in Biology: introducing NetBioDyn. In: *Multi-Agent-Based Simulations Applied to Biological and Environmental Systems*. IGI Global.
- Georges, B. (2022). Quand l'IA fabrique des images. *Les Echos*, (23800), p.14.
- Tallent, A. (2024). *" L'envers des mots ": Prompt*. The Conversation Media Group.
- Craiyon. Modèle d'IA générative d'images. Consulté sur https://www.craiyon.com/
- *Codium*. (2023, 18 septembre). Dans *Wikipédia*.
- *Upeneus taeniopterus*. (2025, 27 janvier). Dans *Wikipédia*.