ISSN en cours

Revue Internationale de

systēmique

Vol. 1, N° 2, 1987

afcet

Dunod

AFSCET

Revue Internationale de systēmique

Revue Internationale de Sytémique

volume 01, numéro 2, pages 181 - 207, 1987

Dynamac : un logiciel interactif pour l'étude des systèmes dynamiques

Bertrand Rousseau

Numérisation Afscet, décembre 2015.

Références

G. PINSON, A. DEMAILLY et D. FAVRE, *La Pensée, Approche holographique,* P.U.L., Lyon, 1985.

T.G.R. BOWER, *Human Development*, Freeman, San Francisco, 1979.

A. KORZYBSKI, *Science and Sanity,* International non-aristotelian Library, Lakeville, Connecticut, 1933.

J.C. TABARY, *Du Logique au Vivant et du Vivant au Logique : une éternelle spirale dorée,* Actes des Gésys, 1983.

A.E. VAN VOGT, *Le Monde des A*, traduction de Boris Vian, Gallimard, Paris, 1953.

REVUE INTERNATIONALE DE SYSTEMIQUE
Vol. 1, N° 2, pp. 181 à 207

DYNAMAC : UN LOGICIEL INTERACTIF POUR L'ETUDE DES SYSTEMES DYNAMIQUES

Bertrand ROUSSEAU

Université Claude-Bernard [1] — IMAG [2]

Résumé
La simulation des systèmes dynamiques impose l'utilisation de logiciels spécialisés. Néanmoins, il semble que la plupart des logiciels existant n'ont pas suivi l'évolution des potentialités de l'informatique en matière d'interaction et d'aide à l'utilisateur. Après avoir présenté les principes directeurs pour la conception d'un logiciel de simulation interactif, nous décrivons DYNAMAC, un programme reposant sur ces principes.

Abstract
The simulation of dynamic systems implies the use of specialized software. Nevertheless, it seems that most of existing softwares did not gain much from current computer trends related to interaction and user assistance. After a presentation of basic principles for the design of interactive simulation softwares, this paper describes DYNAMAC, a program which lies on those principles.

Depuis l'introduction des micro-ordinateurs en milieu scientifique, un nombre croissant de logiciels de simulation de modèles à base d'équations différentielles ont fait leur apparition sur ce type de matériels. On peut citer SCES, ACSL, ISIM, DYNAMO, TUTSIM, STELLA, PHASER.

Mais bien souvent, il s'agit d'adaptations de logiciels initialement développés dans un cadre informatique plus classique, à une époque où les contraintes matérielles et techniques étaient fortes.

1. Laboratoire de Biométrie, 69622 Villeurbanne Cédex, France.
2. Laboratoire ARTEMIS, B.P. 68, 384902 Saint-Martin d'Hères Cédex, France.

Revue internationale de systémique. Vol. 1/87/02/181-207/27/$ 4.70/

Les progrès quantitatifs accomplis ces dernières années en matière technologique et leur mise en œuvre sur des matériels à grande diffusion (mémoire et écrans bitmap par exemple) ont permis non seulement d'entreprendre la réalisation de programmes plus ambitieux et plus rapides, mais aussi d'envisager de nouveaux types d'applications apportant un progrès qualitatif dans la façon d'utiliser l'outil informatique.

Parmi ces derniers, un concept très en vogue est celui d'interaction homme/machine, ensemble des mécanismes permettant à l'utilisateur de dialoguer avec un programme, celui-ci étant perçu par l'utilisateur par l'intermédiaire d'un modèle conceptuel. Il existe en effet une relation très forte entre la forme externe que revêt un programme (ses entrées/sorties) et la façon dont l'utilisateur intègre ce programme dans son schéma Action/Réflexion. Pour que cette intégration soit la plus efficace possible, il est nécessaire de limiter le nombre des intermédiaires entre les modèles mentaux de l'utilisateur et les entrées/sorties du programme (Kay 84).

Dans cette optique, l'apparition des techniques de l'interaction graphique basées entre autres sur les concepts de fenêtres, d'icones et de dispositifs de pointage ouvre un large champ d'investigations dans le but d'améliorer l'interaction homme/machine en proposant à l'utilisateur des modèles conceptuels plus proches de sa façon de travailler.

On peut alors se demander si les concepteurs de logiciels de simulation ont su tirer profit de ce nouveau degré de liberté mis à leur disposition pour améliorer qualitativement la tâche des utilisateurs. Bien que la réponse à cette question ne soit pas simple, on peut néanmoins affirmer que la plupart de ces programmes reposent toujours sur les mêmes concepts. Il est d'ailleurs révélateur de constater qu'un bon nombre d'entre eux portent encore les stigmates de l'époque où ils fonctionnaient en mode de traitement par lot (ou batch), par exemple sous la forme de cartes de contrôle dans le texte décrivant le modèle.

La présente contribution essaie de dégager les principes fondamentaux sur lesquels devrait s'appuyer une nouvelle génération de logiciels de simulation. On décrit ensuite Dynamac, un logiciel de simulation fonctionnant sur MacIntosh, qui constitue une première tentative dans cette direction.

1. Vers un logiciel de simulation interactif

L'inclusion de fenêtres et de menus déroulants dans un programme ne suffit pas à rendre celui-ci interactif. Il s'agit plutôt de briques qu'il faut assembler dans le but d'approcher au mieux le mode de travail des utilisateurs. En corrolaire, l'interface-utilisateur d'un logiciel doit être spécifiée en même temps que ses fonctionalités et non pas après comme c'est souvent le cas dans les logiciels de simulation.

En utilisant un logiciel de simulation, un utilisateur souhaite

– construire un modèle
– explorer les comportements possibles de ce modèle
– expérimenter sur le modèle en se posant des questions du type «que se passe-t-il si je change le modèle dans tel ou tel sens ?».

Cette démarche n'est pas l'inéaire mais est constituée d'allers et retours entre ces différentes étapes. Un logiciel de simulation doit donc être construit en favorisant ce mode de fonctionnement par essais/erreurs, tout en permettant de conserver les résultats des essais précédents à des fins de comparaison.

En outre, il existe plusieurs niveaux d'utilisation d'un logiciel de simulation.

On peut distinguer :

– un niveau pédagogique, où l'étudiant examine des modèles à 2 variables d'état en s'appuyant sur leur portrait de phase et en s'exerçant aux techniques de l'analyse qualitative. A ce niveau, la facilité d'apprentissage et d'utilisation du logiciel prend toute son importance.

– un niveau où l'on étudie de petits modèles (dimension $\leqslant 5$) comme cela est souvent le cas dans des disciplines telles que la dynamique des populations ou la cinétique chimique.

– un niveau portant sur l'étude de modèles beaucoup plus importants.

C'est le cas de domaines tels que l'économie ou l'écologie par exemple.

Le logiciel doit pouvoir se plier à ces différents types d'utilisations qui entraînent chacun l'utilisation d'outils différents.

Pour atteindre ces buts, on peut retenir les ingrédients suivants :

Un langage simple de description de modèles

Le logiciel doit offrir un langage simple de description de modèles. A cet effet deux conceptions s'opposent. La solution la plus employée consiste à décrire le modèle dans son formalisme différentiel, sous la forme d'un texte, à l'aide d'un langage de description. La seconde solution vise à représenter le modèle dans un formalisme non différentiel à l'aide d'une représentation graphique à base de graphes par exemple. C'est le cas du logiciel STELLA où cette étape du processus de modélisation est brillamment traitée dans le cadre de la Dynamique des Systèmes.

Cette solution possède l'avantage de fournir à l'utilisateur un formalisme plus proche de sa façon de penser et contribue donc à le libérer

de l'hermétisme de la formulation mathématique et de la lourdeur d'un langage de programmation. Malheureusement, il est difficile de trouver un formalisme général et la plupart des formalismes existants sont soit réducteurs à certaines classes de modèles comme dans la représentation boîtes/flèches du programme COSMOS (modèle à compartiments : Hamrouni 1979) ou le formalisme pseudo-chimique (Pavé 1980), soit imposent à l'utilisateur un modèle de pensée qui ne lui est pas forcément adapté (Dynamique des Systèmes de Forrester. Par exemple : Forrester 1971), soit sont trop complexes (systèmes écologiques : Odum 1983). Cette raison, jointe à d'autres qui seront exposées au paragraphe 3, nous ont conduit à adopter la solution classique.

$$Q1' = r1.\,Q1 - k1.\,Q1.\,Q2$$

$$Q2' = r2.\,Q2 - k2.\,Q1.\,Q2$$

Figure 1 a

$$Q1 + S \xrightarrow{k1} (1 + r1)\,Q1$$

$$Q2 + S \xrightarrow{k2} (1 + r2)\,Q2$$

Figure 1 b

Figure 1 : Description d'un modèle de compétition entre deux espèces à l'aide de plusieurs formalismes. Les variables d'état Q1 et Q2 représentent les abondances de chacune des deux espèces. Le modèle suppose que les abondances ne dépendent que de la croissance naturelle et de l'interaction de compétition :

Figure 1 a : formalisme différentiel

Figure 1 b : formalisme pseudo-chimique

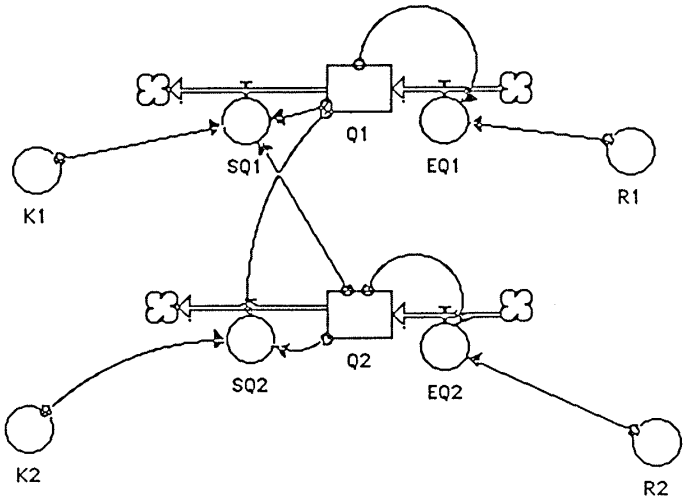

Figure 1 c

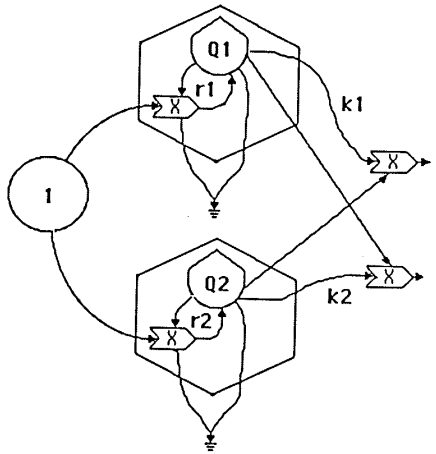

Figure 1 d

Figure 1 c : formalisme de Forrester (obtenu avec STELLA)

Figure 1 d : formalisme de Odum

La représentation graphique de toutes les simulations

La compréhension du comportement d'un modèle passe par des représentations graphiques. Chaque état du système pouvant être considéré comme un point dans un espace multidimensionnel, on peut visualiser l'évolution de cet état sous la forme de courbes tracées dans un repère formé par 2 variables du modèle. Chaque point de la courbe représente donc une vue partielle de l'état du système à un certain temps.

Le plus fréquemment, ce comportement est difficile à percevoir pour plusieurs raisons :

a) Un modèle peut adopter des comportements très variés en fonction de l'état initial qu'on lui impose lors d'une simulation.

b) La plupart des modèles comportent un nombre important de variables d'état et il est donc difficile de connaître leur dynamique à l'aide de quelques graphiques seulement.

c) Le comportement d'un modèle peut évoluer lorsque l'on fait varier plus ou moins les valeurs de ses paramètres.

L'appréhension de la dynamique d'un modèle passe donc par la multiplication des simulations et des représentations sous des angles variés des mêmes résultats de simulation.

La plupart des logiciels utilisent les chroniques, graphiques décrivant l'évolution de plusieurs variables du système en fonction du temps. Bien que ce type de visualisation soit indispensable, il ne permet de représenter qu'une solution du modèle. Le portrait de phase, par contre, est basé sur la représentation simultanée des résultats de plusieurs simulations, pour deux variables d'état du modèle. Ces deux types de vues graphiques possibles d'un modèle sont complémentaires et il est essentiel qu'un programme de simulation autorise la création d'un nombre quelconque de celles-ci, tout en permettant leur visualisation simultanée à l'écran.

Tous les résultats de simulation doivent être des objets manipulables

Il est fondamental que les solutions calculées soit mises en mémoire et manipulables par l'utilisateur. Ceci permet par exemple de choisir une solution dessinée dans une des vues graphiques du modèle et de la faire redessiner dans une autre vue. On peut donc facilement obtenir plusieurs points de vues différents pour une même solution. Un étudiant peut, par exemple, dessiner un portrait de phase pour un modèle prédateur-proie et étudier les différents types de solutions dans des vues de type chroniques. Ceci permet également d'effectuer

des opérations sur toute simulation déjà effectuée : effacer une trajectoire ou poursuivre une simulation à partir du temps où on l'avait arrêtée entre autres exemples.

La distinction entre le modèle et ses vues

Il est important de bien dissocier le modèle et les paramètres qui permettent de définir une de ses vues. Chaque vue d'un modèle peut posséder des caractéristiques complètement différentes de celles de ses voisines : variables représentées, échelles, conditions initiales, méthode d'intégration...). Ces paramètres ne doivent donc pas être fixés dans le texte décrivant le modèle, mais sont propres aux vues.

Grâce à cette distinction on peut, par exemple, modifier le modèle et répercuter les modifications dans certaines de ses vues (les solutions sont alors recalculées en tenant compte des modifications). De même, on peut modifier les caractéristiques d'une vue et redessiner les solutions associées à la vue.

La localisation spatiale approximative

Une notion très importante concerne la localisation approximative par l'utilisateur d'un certain nombre de paramètres relatifs à l'espace. Considérons par exemple le choix des conditions initiales avant le lancement d'une simulation. La plupart des logiciels de simulation imposent à l'utilisateur de fixer les conditions initiales de façon numérique et précise. Or, dans le cas des portraits de phase, on peut envisager de choisir les conditions initiales à l'aide de la souris, en plaçant le pointeur sur un point de la vue situé dans une zone de conditions initiales intéressante. Cette technique, malgré sa futilité apparente, apporte plus qu'un confort d'utilisation supplémentaire ; elle permet de s'approcher d'un mode de travail de type papier/crayon en supprimant un certain nombre d'étapes qui alourdissent le cycle réflexion/action de l'utilisateur.

Le logiciel Dynamac constitue une première tentative dans cette voie. Ce logiciel a été entièrement développé sur MacIntosh et s'appuie largement sur les modèles d'interaction qui lui sont propres (Averill 1984), à l'aide des fonctions contenues dans sa ROM.

2. Présentation du logiciel Dynamac

2.1. Le langage de description de modèles

Dans un premier temps, l'utilisateur doit décrire son modèle à

l'aide d'un texte, introduit au clavier à l'aide d'un éditeur associé à une fenêtre. Ce texte obéit à une syntaxe bien définie.

Schématiquement, un modèle est composé de trois parties, la dernière étant optionnelle.

– On déclare les différentes variables intervenant dans le modèle : leur type et leur nom. De manière assez classique, on distingue les variables d'état, les variables auxiliaires, les constantes, les paramètres, les variables d'entrées, les variables tabulées et les variables booléennes. A chaque variable on peut associer une échelle utilisée lors de la création des graphiques. Il est possible de fixer, pour chaque variable d'état, une condition initiale qui sera utilisée par défaut lors des simulations.

– On décrit les équations du modèle en utilisant les variables déclarées, les opérateurs arithmétiques de base ainsi qu'un certain nombre de fonctions mathématiques. Il est également possible d'utiliser des constructions du type si... sinon... alors.

– On déclare les éventuelles variables de sortie (variables n'intervenant pas dans la dynamique du modèle) et les équations associées.

2.2. Les vues d'un modèle et le tracé des solutions

L'étape suivante consiste en la création d'un certain nombre de vues graphiques du modèle. La représentation d'une même solution dans des vues différentes permet d'obtenir plusieurs perspectives permettant d'examiner les mêmes objets sous différents angles.

Dynamac distingue trois types de vue :

Les portraits de phase

Peu utilisés dans les logiciels de simulation plus classiques, il s'agit au contraire dans Dynamac d'un mode de représentation privilégiée. Il s'agit de la représentation graphique des solutions du modèle dans un repère formé par deux variables d'état de celui-ci. Les avantages de ce type de représentation sont multiples, surtout dans le cas de petits modèle :

– il permet une cartographie des comportements possibles d'un modèle sur un même graphique

– il autorise le choix de conditions initiales à l'aide de la souris

– il est adapté à la mise en œuvre d'un certain nombre d'outils d'aide à l'analyse qualitative qui seront explicités au paragraphe 2.4.


```
modele rouen;
etat
     poploc = 2000 [ 0 , 5000 ] ,
     resec = 400 [ 0 , 3000 ] ,
     image = 1  [0.5 , 1.5 ] ;
tabul
     relint(8), impot(5) ;
aux
     im , sesat ,  em , emin , emex , intelp ;
param
     tipl = 0.06 , tir = 0.05 , temin = 0.01 , emm = 0.005 , maxintelp = 280 ,
     tarres = 0.03 , tim = 0.01 , limite = 2500 , seuil = 1 ;

debut
     relint(0) = 0.8 ;
     relint(0.1) = 0.9 ;
     relint(0.2) = 1 ;
     relint(0.25) = 1.2 ;
     relint(0.3) = 1.2 ;
     relint(0.35) = 1 ;
     relint(0.4) =  0.9 ;
     relint(0.45) = 0.8 ;

     impot( 0.6  ) = 0 ;
     impot( 0.8  ) = 0.5 ;
     impot( 1 ) = 1 ;
     impot( 1.2  ) = 1.5 ;
     impot( 1.4  ) = 2 ;

     sesat = resec / poploc ;
     intelp = tipl * poploc + tir * resec ;
     im = tim * poploc * relint( sesat ) * ( 2.5 * image - 1.5 ) ;
     emin = si [ relint( sesat ) ≥ seuil ]  alors 0 sinon temin * poploc ;
     emex = emm * poploc ;
     em = emex + emin ;

     poploc' = im - em ;
     resec' = tarres * image * ( 1 - ( resec / limite ) ) * resec ;
     image' = ( 1 - ( intelp / maxintelp ) ) * ( 1 - ( image / 1.2 ) ) * image ;
fin

sortie obsrel, obsimpot , obsimage , obspop , obintel ;
debut
     obsrel = relint( sesat ) ;
     obsimpot = ( 2.5 * image - 1.5 ) ;
     obspop = im - em ;
     obintel = 1 - ( intelp / maxintelp ) ;
fin.
```

Figure 2. Description dans le langage Dynamac d'un modèle de l'impact de la présence de résidents secondaires sur la population locale d'une petite ville. Le langage autorise la déclaration de variables tabulées et l'écriture de teste conditionnels.

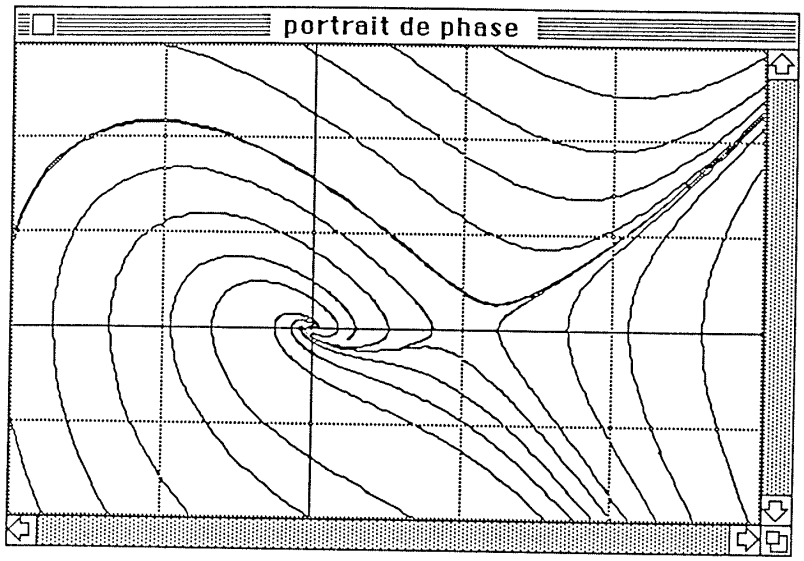

Figure 3. Exemple de portrait de phase. Chacune des courbes (ou trajectoire) correspond à une simulation à partir de conditions initiales réparties sur tout le plan.

Les chroniques

Type de graphique le plus utilisé en simulation, il s'agit de la représentation de l'évolution d'un certain nombre de variables du modèle en fonction du temps. Contrairement aux portraits de phase, ces graphiques ne permettent de représenter qu'une seule solution, mais pour un nombre de variables plus important.

Les graphes quelconques

Il est souvent utile de pouvoir disposer d'autres types de représentation que ceux décrits précédemment. Les graphes quelconques permettent de représenter une ou plusieurs solutions dans un repère formé par deux variables quelconques du modèle, on peut par exemple dessiner l'évolution d'une variable d'état en fonction de sa dérivée,

ou bien encore dans le cas de modèles à plus de 2 dimensions représenter une projection plane des solutions à l'aide de deux variables de sortie définies comme des combinaisons linéaires des variables d'état.

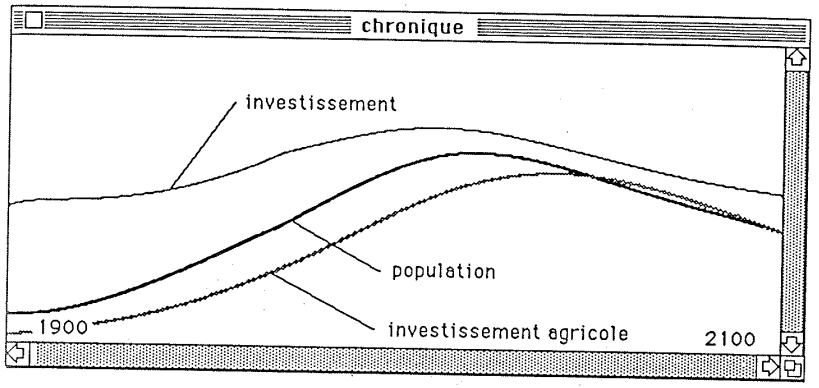

Figure 4. Exemple de chronique correspondant à une solution du modèle mondial de Forrester (Forrester 1971). Chaque courbe représente l'évolution d'une variable du modèle en fonction du temps.

L'utilisateur peut définir un nombre quelconque de ces vues, en choisissant lui même leur taille et leur emplacement à l'intérieur d'une page au format A4 choisie parmi plusieurs. Cette opération se fait entièrement à l'aide de la souris par l'intermédiaire d'une zone de dialogue (figure 6).

Lors d'une impression ou de la sauvegarde sur disque du travail effectué, ce sont ces pages qui sont manipulées. La commande d'impression imprime une page complète avec toutes les vues lui appartenant ; la commande d'enregistrement provoque la création d'un fichier au format MacPaint, contenant une image complète de la page.

Le logiciel va ensuite associer chacune de ces surfaces à une fenêtre, qui vient alors se superposer aux fenêtres déjà existantes sur l'écran.

Chaque fenêtre constitue un environnement possédant ses caractéristiques propres : variables représentées, échelles, conditions initiales

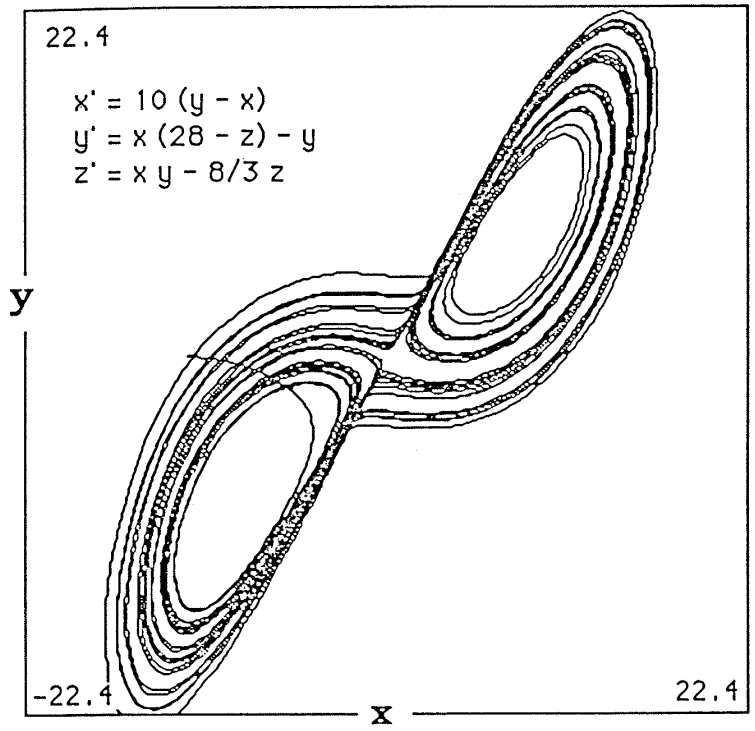

Figure 5a

Figure 5. Représentation d'un portrait de phase (5a) et d'un graphe quelconque (5b) pour l'attracteur étrange de Lorentz. Les graphes quelconques permettent de dessiner des vues en perspective.

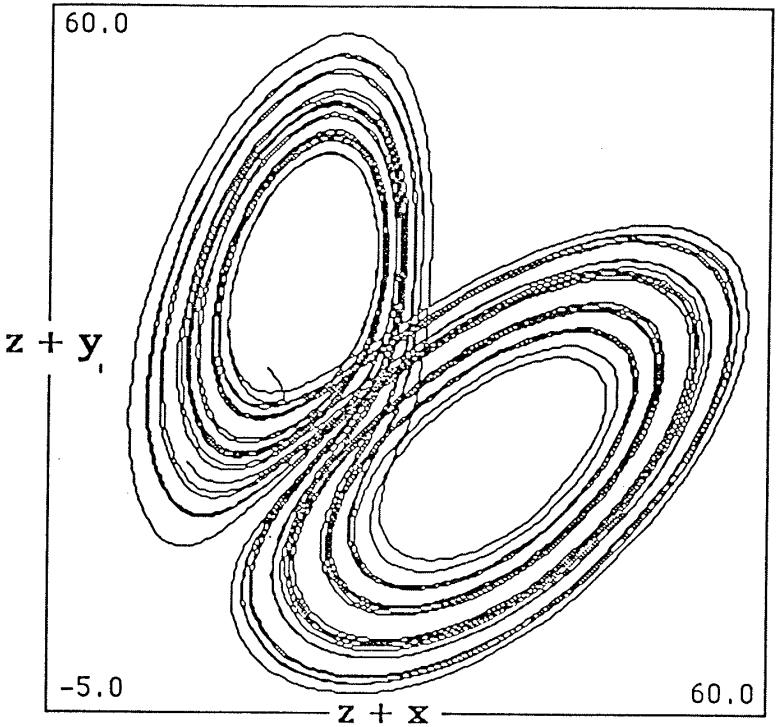

Figure 5b

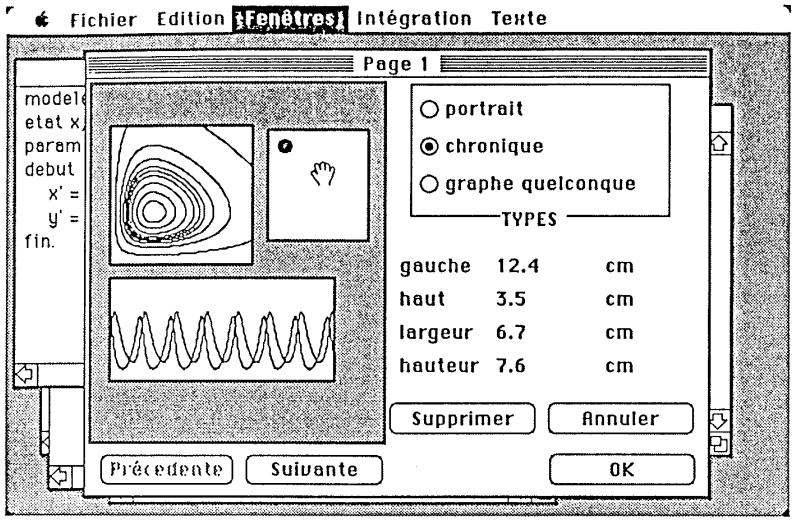

Figure 6. La zone de dialogue de création des vues. Les vues sont créées et modifiées à l'aide de la souris.

par défaut, méthode et pas d'intégration, valeurs des paramètres du modèle sont locaux à une fenêtre et peuvent donc être modifiés indépendamment du texte et des autres fenêtres.

Lors de la création d'une vue, ces caractéristiques prennent une valeur par défaut qui est soit imposée dans le texte du modèle, soit dans le cas contraire, fixée par le programme.

Diverses zones de dialogues permettent de modifier les valeurs de ces caractéristiques dans une vue donnée, même si des simulations ont déjà été effectuées. Dans ce dernier cas, il est alors possible de faire redessiner les solutions en tenant compte des modifications apportées.

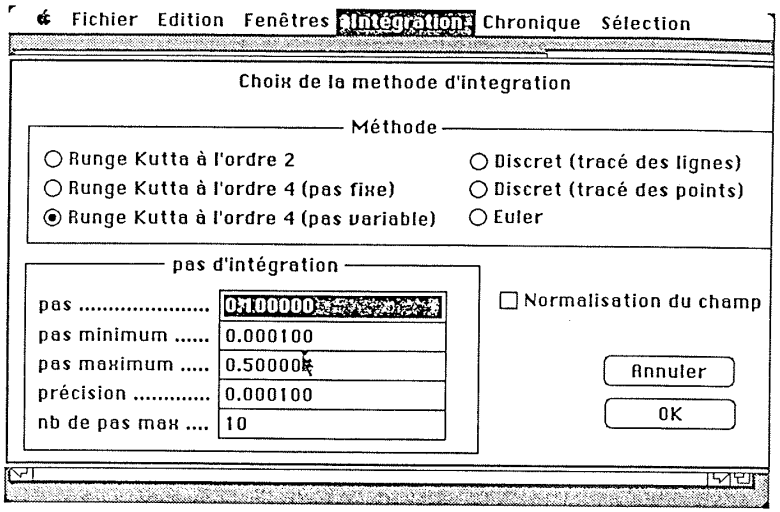

Figure a

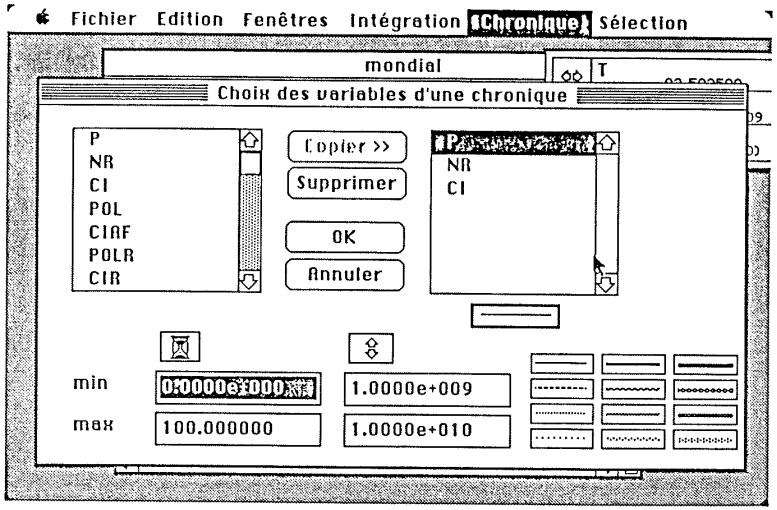

Figure b

Figure 7. Deux des zones de dialogue utilisées pour la modification des vues. La première (7a) correspond au choix d'une méthode d'intégration, la seconde (7b) permet de choisir des variables et les échelles pour une vue graphique de type chronique.

La façon dont on lance une simulation dépend du type de la vue sur laquelle on travaille. Dans tous les types de vue, on peut lancer l'intégration à partir d'une zone de dialogue dans laquelle on introduit les valeurs numériques des conditions initiales à l'aide du clavier (figure 8).

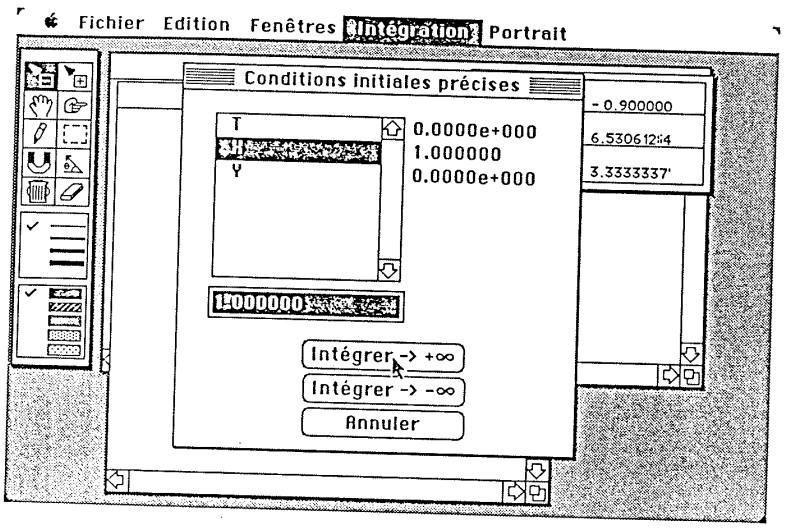

Figure 8. Zone de dialogue permettant de fixer les conditions initiales dans le but de lancer une simulation. On peut procéder ainsi sur les trois types de vue.

En revanche, dans une fenêtre de type portrait il est possible de choisir l'état initial à l'aide de la souris. Les portraits de phase donnent accès à une palette, fenêtre qui permet de choisir divers outils correspondant chacun à une opération que l'on peut effectuer sur ce type de vue. Parmi ceux-ci figurent des outils d'intégration. Après avoir sélectionné un de ces outils, l'utilisateur peut choisir un état initial en plaçant cet outil en un point quelconque de la vue. Une fenêtre permet de visualiser la valeur des coordonnées de ce point, et il suffit d'un clic sur la souris pour lancer la simulation à partir de cet état.

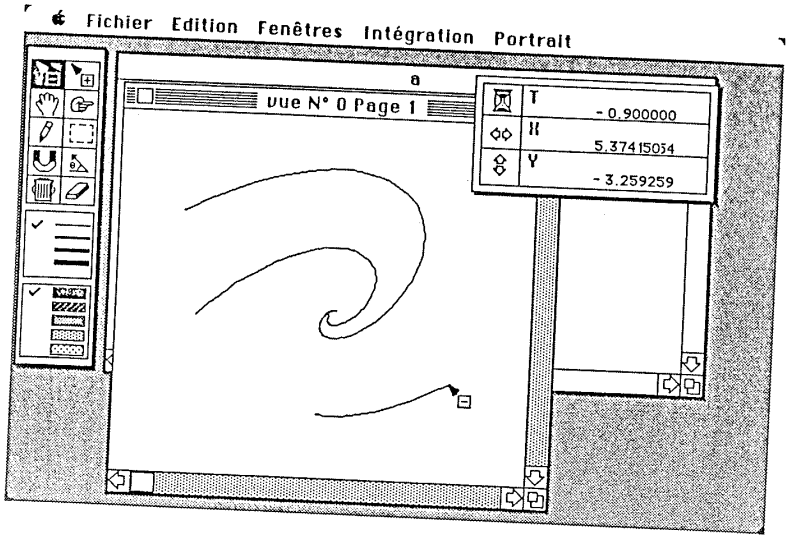

Figure 9. Choix d'un état initial à l'aide de la souris. La trajectoire correspondant à l'état choisi se dessine au fur et à mesure des calculs, tandis que les valeurs des variables sont affichées dans une fenêtre spéciale.

2.3. Modèle, vues et solutions

L'analyse exploratoire du comportement d'un modèle nécessite, pour être efficace, que le modèle, ses vues et les diverses simulations effectuées soient manipulables en tant qu'objets. En particulier, l'utilisateur doit pouvoir modifier leurs caractéristiques à l'aide d'un minimum d'opérations.

A cet effet, ces objets sont liés par des relations hiérarchiques d'appartenance et possèdent chacun des caractérisituqes propres que l'utilisateur peut modifier à tout moment.

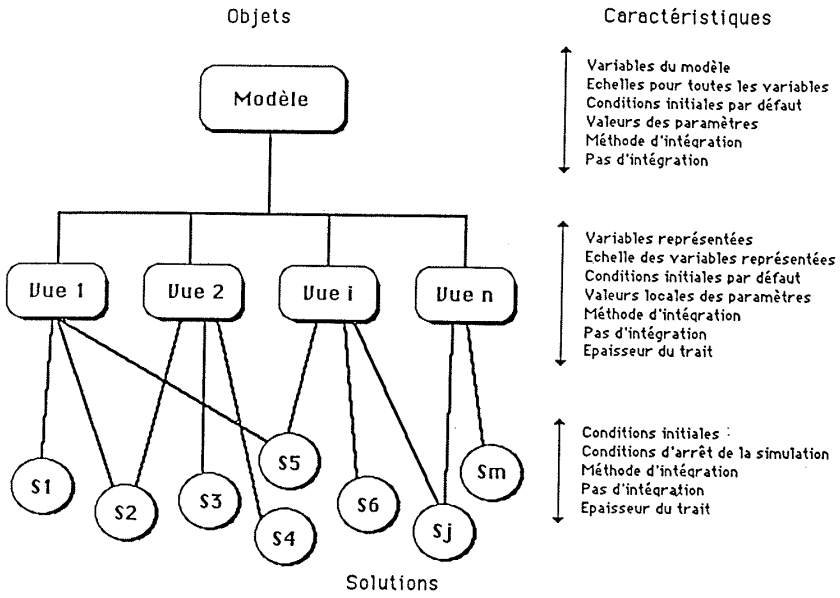

Figure 10. Relations d'appartenance entre les objets utilisés en simulation. Un modèle peut posséder plusieurs vues, chacune des vues contient un certain nombre de solutions, mais une même solution peut être partagée entre plusieurs vues. Par défaut, chaque objet hérite des caractérisitiques de l'objet qui le contient, mais l'utilisateur peut à tout moment modifier les caractéristiques d'un objet et répercuter ou non ces modifications sur les objets qu'il contient.

Lors du lancement d'une simulation par exemple, il y a création d'un objet solution dont les caractéristiques méhode, conditions initiales, valeurs des paramètres, épaisseur du trait prennent les valeurs des caractéristiques associées à la vue dans laquelle on lance la simulation. Par la suite, il est toujours possible de faire redessiner cette solution arprès avoir modifié les valeurs de ses caractéristiques. De même,

lors de la création d'une nouvelle vue, les valeurs des caractéristiques de la vue sont celles du modèle, que l'on peut également modifier par la suite.

Ce modèle de représentation des principaux concepts de la simulation donne une plus grande liberté à l'utilisateur car il se rapproche davantage de son mode de travail constitué d'essais et d'erreurs.

On peut envisager par exemple de sélectionner une des solutions tracées dans un portrait puis de la faire redessiner dans une vue différente, possédant ses caractéristiques propres.

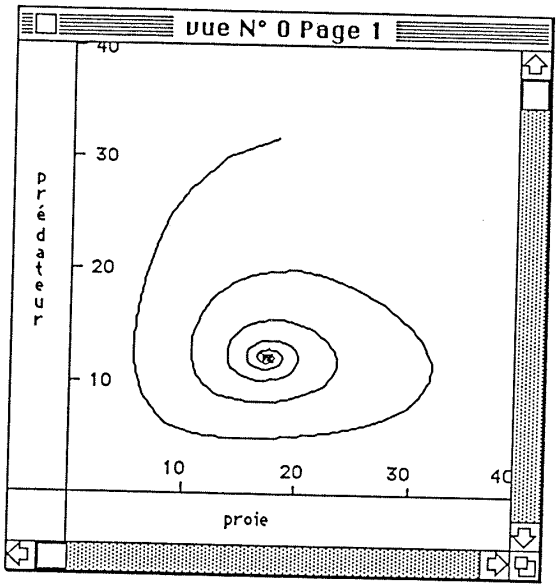

Figure 11a

Figure 11. Solution d'un modèle prédateur-proie dessinée dans deux vues différentes. La figure 11a montre sa représentation sous la forme d'un portrait de phase, tandis que la figure 11 b présente la chronique associée. Partant d'un état initial, les effectifs des deux populations oscillent, **mais** ces oscillations s'amortissent et on atteint alors un point d'équilibre où les effectifs ne varient plus.

Autre exemple : il est possible de commencer l'étude dans une vue puis de changer d'échelle sans pour autant perdre le bénéfice des simulations déjà effectuées.

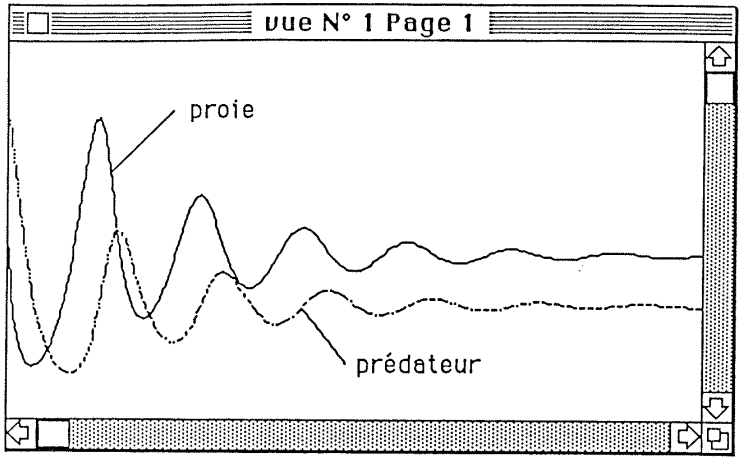

Figure 11b

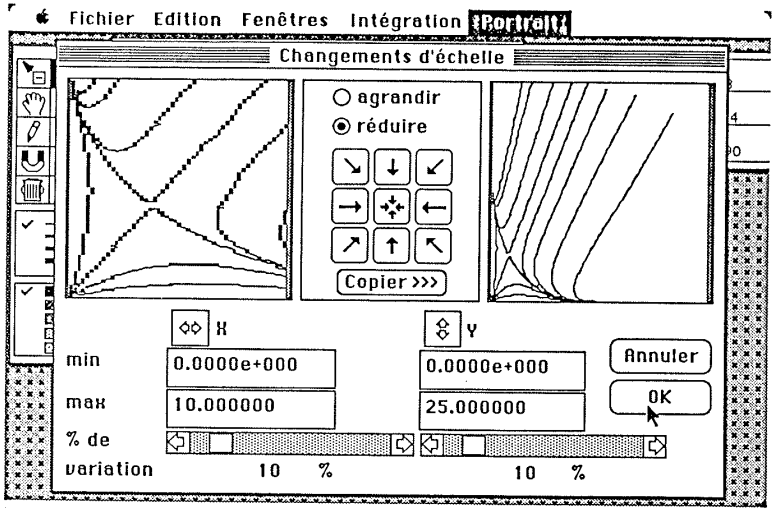

Figure 12. Une zone de dialogue autorise des changements d'échelles effectués entièrement à l'aide de la souris. Pour chaque changement l'utilisateur peut voir l'effet produit sur les solutions déjà dessinées.

2.4. Les outils d'analyse qualitative

Lors de l'étude de petits modèles, on peut utiliser des outils d'aide à l'analyse qualitative des comportements du modèle. Dynamac en contient quatre, utilisables dans un portrait de phase avec des modèles à deux ou trois variables d'état.

Le dessin du champ des vecteurs vitesse

La représentation graphique des vecteurs tangents aux trajectoires en tous les nœuds d'un réseau régulier permet d'obtenir une première idée du comportement qualitatif du modèle.

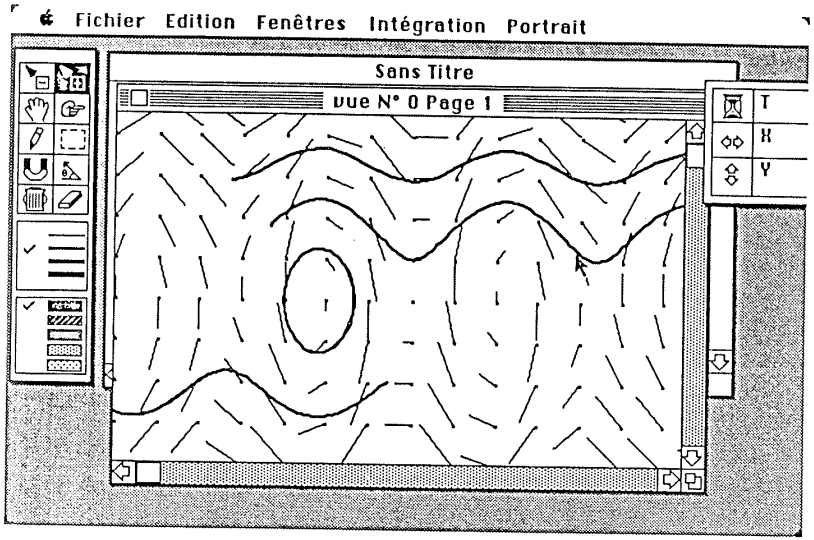

Figure 13. Tracé du champ des vecteurs vitesse en tous les nœuds d'un réseau régulier. La taille de ce réseau peut être choisie à l'aide d'une zone de dialogue.

Le tracé interactif du vecteur champ

Mené à l'aide de la souris, ce tracé permet d'avoir une idée plus fine du champ en un point précis, ou autour d'une zone, ce qui peut donner des indications qualitatives sur le comportement du modèle à l'intérieur de cette zone.

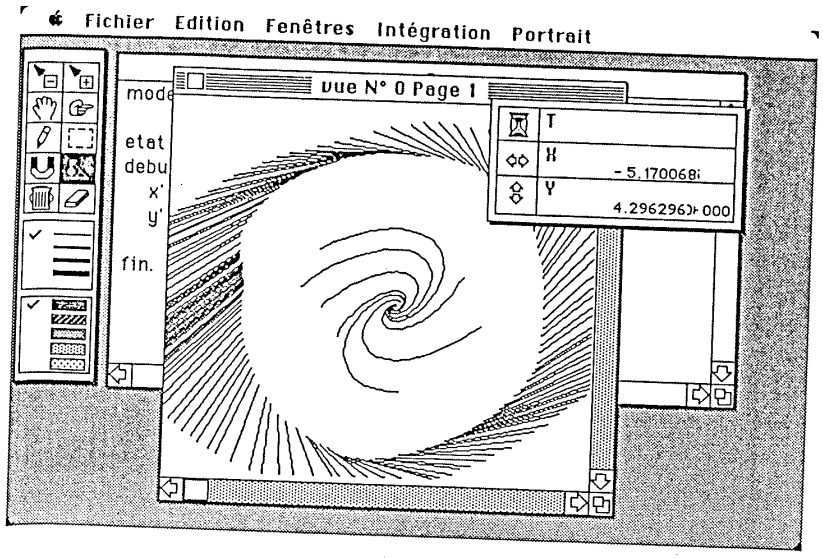

Figure 14. Tracé interactif du champ autour d'une zone.

Le tracé des isoclines

Les isoclines sont les lieux du portrait de phase où les vecteurs vitesse ont même pente. Le tracé de plusieurs de ces courbes permet de déterminer l'emplacement des points d'équilibre et de découper le portrait de phase en zones de comportement homogène.

La recherche et la détermination des points d'équilibre

Dans le cas des systèmes à deux dimensions, Dynamac permet de trouver la position des points fixes, leur nature et la stabilité des points d'équilibre.

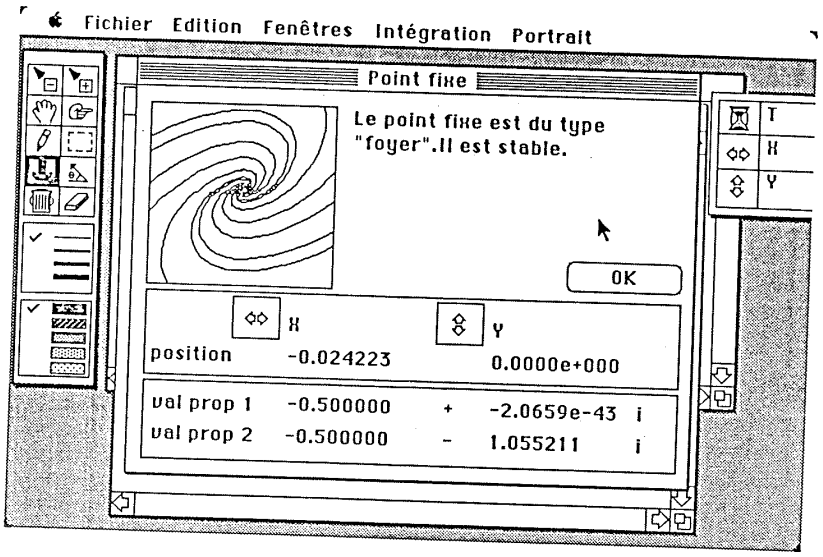

Figure 15. Fenêtre décrivant les caractéristiques d'un des deux points d'équilibre détectés par le programme dans la vue du modèle représentée sur la figure 4.

3. Directions de recherche futures

La fonction d'analyse exploratoire d'un logiciel de simulation peut être encore enrichie par l'introduction de nouvelles fonctionnalités et la création de modèles conceptuels plus puissants. Dans cet esprit, on peut retenir quatre voies de recherches.

La création et la maintenance de contraintes entre les différentes vues d'un modèle

Dans la plupart des cas, les différentes vues graphiques d'un modèle ne sont pas indépendantes dans la mesure où l'utilisateur souhaite qu'une action sur une vue soit immédiatement répercutée sur d'autres vues. Par exemple : une simulation lancée à partir d'une vue peut être visualisée dans d'autres vues simultanément, un changement d'échelle peut concerner plusieurs vues, etc.

La prise en compte de ce type de contraintes permettrait d'éviter les tâches répétitives conduites successivement sur plusieurs vues différentes, un peu à la façon dont un tableur propage les modifications à l'aide de formules de calculs.

La gestion des versions successives d'un modèle

La construction d'un modèle relève d'un processus d'essais/erreurs et de raffinements successifs. Il est donc important de pouvoir mémoriser et gérer l'arbre généalogique des diverses versions d'un modèle étudié, de façon à pouvoir les comparer et éventuellement, les reprendre. Ce point amène une extension du modèle conceptuel qui avait été envisagé jusqu'à présent. En effet, la possibilité de travailler simultanément sur plusieurs versions d'un modèle, entraîne l'apparition d'un objet supplémentaire : le projet, celui-ci étant constitué de plusieurs modèles qui peuvent éventuellement partager les mêmes vues.

L'inclusion d'outils numériques plus évolués

La simulation d'un modèle s'inscrit dans une démarche expérimentale qui ne donne qu'un aperçu partiel des comportements possibles d'un modèle. L'intégration d'outils d'analyse numérique complémentaires permettrait de conduire une exploration plus systématique de ces comportements. On peut citer l'analyse de sensibilité qui consiste en l'étude de l'impact des variations des paramètres sur l'évolution de chacune des variables du modèle, la recherche des points de bifurcation : valeurs des paramètres pour lesquels le modèle change qualitativement son comportement, entre autres exemples. Cette intégration doit être accompagnée d'une étude sur la façon d'interagir avec ces méthodes. Cipière (1978) propose ainsi un programme interactif de recherche des solutions périodiques d'un système différentiel où l'algorithme numérique s'efface au profit d'un travail de type papier/crayon.

La description du modèle dans un formalisme non-mathématique

Le logiciel Dynamac a été réalisé dans le cadre d'un projet de plus grande envergure : le projet EDORA (Equations Différentielles Ordinaires et Récurrentes Appliquées). Regroupant l'INRIA, l'INRA, ainsi que plusieurs laboratoires universitaires et du CNRS, son objectif est de fournir aux biologistes-modélisateurs un logiciel incluant la majorité des outils nécessaires à la modélisation et capable de les guider à toutes les étapes de la construction d'un modèle. Le noyau central de ce logiciel est constitué d'un moteur d'inférence travaillant sur une base de connaissances centrée-objet.

Au sein de ce projet le logiciel Dynamac vise l'étude des mécanismes d'interaction utilisables en simulation.

Parmi les axes de recherches de ce projet figure un problème déjà soulevé au début de cet article : le problème de la description d'un modèle dans un formalisme non-mathématique. La résolution de ce problème semble impliquer non seulement la réalisation d'interfaces interactives, mais également l'intervention d'un système de gestion de bases de connaissances.

En effet, la construction d'un modèle entraine la mise en œuvre d'autres aides que celles apportée par les interfaces interactives.

Cette aide peut être fournie à deux niveaux :

– La recherche dans une base de modèles, d'un ou plusieurs modèles ou sous-modèles répondant à certaines conditions et adaptés à un domaine particulier,

– La construction progressive du modèle à l'aide des hypothèses sur sa structure et son fonctionnement.

Dans les deux cas, cette approche nécessite l'existence :

– d'un ou plusieurs formalismes non-mathématiques permettant à l'utilisateur de décrire le modèle en termes de structure et de fonctionnement.

– d'une structure de représentation informatique des modèles permettant l'enregistrement de leurs caractéristiques,

– d'algorithmes et d'heuristiques permettant de passer progressivement du formalisme de description au formalisme mathématique, par combinaison de sous-structures de base correspondant à des situations élémentaires.

Le passage du formalisme de description au formalisme mathématique peut être effectué de façon progressive, les étapes étant guidées par le logiciel. On peut ainsi envisager une stratégie telle que :

– définir les variables
– spécifier l'existence d'interactions entre ces variables
– définir plus précisément la nature de ces interactions.

Cette approche soulève de gros problèmes qui ne peuvent être résolus que par l'utilisation combinée

– de techniques de représentation et d'exploitation de la connaissance

– de méthodes de calcul symbolique (assemblage de primitives, simplification, re-paramétrages, analyse dimensionnelle)

– des outils de l'interaction graphique permettant d'envisager la réalisation d'un éditeur de modèles.

Conclusion

Le degré d'interactivité d'un programme se mesure à l'aide qu'il apporte dans son utilisation. Dans le domaine du calcul scientifique, et celui de la modélisation en particulier, cette aide ne doit pas se limiter à rendre le programme plus facile d'accès, par l'emploi d'un ensemble de techniques graphiques. Il convient également de faciliter l'accès aux outils de calcul intégrés à un programme en créant des ponts entre les modèles mentaux de l'utilisateur et les outils mathématiques plus ou moins complexes dont il a besoin, par l'intermédiaire de modèles conceptuels adaptés. Parce qu'il repose sur une étude des pratiques de l'utilisateur, qu'il induit de nouvelles façons de travailler, et qu'il permet de proposer des outils nouveaux que sont les algorithmes interactifs, ce type de travail ne relève pas seulement du domaine de la programmation, mais constitue une contribution méthodologique.

Références

AVERILL, «User interface Guidelines», *Inside MacIntosh.* Apple Computer, 1984.

CIPIERE P. «Un programme de détermination des solutions périodiques d'une équation différentielle par la méthode de Schmitt», *RAIRO Automatique/Systems Analysis and Control,* Vol. 12, no 12, pp. 155-170, 1978.

FORRESTER J.W., «World Dynamics», *Wright-Allen Press,* Cambridge, 1971.

HAMROUNI M.K., «Etude et développement d'un système informatique d'aide à l'élaboration de modèles en biologie», Thèse 3e cycle, Paris VI, 1979.

KAY A., «Les logiciels», *Pour la Science*, no 85, pp. 14-22, novembre 1984.

McDONALD J.A., PEDERSEN J., «Computing environments for data analysis» :
I : Introduction
II : Hardware
SIAM J., *Sci. Computer*, Vol. 6, no 4, pp. 1004-1021, october 1985.

ODUM H.T., «System Ecology», Wiley, New-York, 1983.

PAVE A., «Contribution à la théorie et à la pratique des modèles mathématiques pour l'analyse dynamique des systèmes biologistes», Thèse Doct. ès Sciences, Université Claude Bernard, Lyon 1, 1980.

PAVE A., RECHENMANN F., «Computer aided modelling in biology : an artificial intelligence approach», in *Artificial Intelligence in Simulation : State of the Art*, Vansteenkiste and al, Eds, SCS publications, Vol. 18, no 1, 1986.

ROUSSEAU B., RECHENMANN F., «Le projet EDORA. Vers un poste de travail informatique pour l'aide à la modélisation des systèmes dynamiques en biologie». XIII Colloque International d'Econométrie Appliquée. *Aux Frontières de l'Econométrie : Expériences en Biologie et en Econométrie*, Sofia-Antipolis, 13-14 mars 1986.