N° d'ordre : 17/2008-M/CH

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université des Sciences et de la Technologie Houari Boumediene

Faculté de CHIMIE

MEMOIRE

Présenté pour l'obtention du diplôme de MAGISTER

En : CHIMIE

Spécialité : Physico-Chimie Théorique et Chimie Informatique

Par : Naziha BENAMROUCHE

Sujet

Implémentation d'outils statistiques QSAR dans la plateforme Virtual Screening Manager for Grids (VSM-G) : Application aux ligands des récepteurs de la cholécystokinine

Soutenu le 15 / 12 / 2008, devant le jury composé de:

B. MAOUCHE	Professeur	USTHB	Président
S. KELLOU-TAIRI	Maître de conférences	USTHB	Directrice de thèse
B. MAIGRET	Directeur de recherches au CNRS	Nancy	Examinateur
C. CHELGHOUM	Professeur	USTHB	Examinateur
M. NAIT ACHOUR	Professeur	USTHB	Examinateur

A mes parents

Remerciements

Les travaux présentés dans ce mémoire de magister ont été menés au laboratoire de Physico-chimie Théorique et Chimie Informatique de l'USTHB, sous la direction scientifique de Madame Safia TAIRI-KELLOU, Maître de conférences à l'USTHB et au sein de l'équipe Orpailleur du Laboratoire Lorrain de Recherche en Informatique et ses Applications (LORIA) de Nancy, dirigée par le Docteur Amédéo NAPOLI (DR2 au CNRS). Ces travaux s'insèrent dans le cadre de l'accord programme TASSILI CMEP (Comité Mixte d'Evaluation et de prospection de la coopération interuniversitaire Franco-Algérienne). Nous tenons à remercier les autorités du CMEP pour avoir financé les stages de recherche au sein du LORIA de Nancy. Je remercie également le LORIA pour avoir financé quatre mois de stage afin de mener à bien ces travaux.

Que Monsieur Amédéo NAPOLI, qui m'a accueillie dans son équipe, trouve ici l'expression de mes profonds remerciements.

J'exprime ma sincère gratitude à Monsieur le Professeur Boubekeur MAOUCHE pour m'avoir accueillie dans le laboratoire. Je le remercie pour le soutien qu'il m'a témoigné et pour l'honneur qu'il me fait en acceptant de juger mes travaux et de présider le jury.

J'adresse mes remerciements les plus sincères à Madame Safia TAIRI-KELLOU pour m'avoir initiée à la recherche dans un cadre studieux, pour m'avoir proposé un sujet de recherche très intéressant et pour m'avoir donné l'opportunité de découvrir la modélisation. La confiance, la disponibilité, l'esprit critique, les divers conseils ainsi que la rigueur, sont les qualités qu'elle a su mettre en évidence pour la réalisation de ces travaux.

Je tiens à remercier tout particulièrement le Docteur Bernard MAIGRET, Directeur de recherche au CNRS, pour m'avoir accueillie au sein de son équipe et d'avoir mis à ma disposition tous les moyens nécessaires à l'avancement et

l'aboutissement à ces travaux. Je lui exprime ma profonde reconnaissance pour l'aide précieuse, les conseils, la confiance et l'intérêt qu'il a accordé en jugeant le présent travail malgré son emploi du temps chargé et ses nombreuses tâches.

Je tiens également à remercier tout chaleureusement Monsieur le professeur Gilles MOREAU, directeur de recherche dans la société pharmaceutique Sanofi- Aventis pour sa disponibilité, ses précieuses explications et ses conseils méthodologiques. Son déplacement au LOARI m'a été d'une grande aide, qu'il trouve ici l'expression de ma profonde gratitude.

J'adresse mes remerciements les plus respectueux à Monsieur Chaabane CHELGHOUM, Professeur à la faculté de chimie de l'USTHB pour l'honneur qu'il me fait en acceptant de juger mes travaux.

Je remercie profondément Monsieur Madjid NAIT ACHOUR, Professeur à la faculté de chimie de l'USTHB, d'avoir accepté de faire partie du jury. Je suis particulièrement ravie par cet honneur.

J'exprime mes remerciements chaleureux à l'ensemble de mes enseignants, plus particulièrement ceux de la post- graduation de physico-chimie théorique et chimie informatique, ainsi que le personnel de la faculté de chimie de l'USTHB.

Je tiens à remercier l'ensemble de mes collègues du laboratoire de Physico-Chimie Théorique et Chimie Informatique, plus particulièrement Samira Feddal, Amel Toumi-Maouche, Souhila Terrachet-Bouaziz, Djamila Ikhlef et Djouadi Abdelmalek pour leur sincère sympathie et leur soutien.

Ce travail étant le fruit d'une collaboration, je ne saurais oublier d'ajouter à mes remerciements ma gratitude la plus sincère à tous les membres du LORIA pour l'immense soutien, scientifique et humain, qu'ils m'ont apporté durant mes stages. Que Birama Ndiaye et Laurent Pierron trouvent ici l'expression de mes remerciements les plus profonds pour leur précieuse aide et leur disponibilité. Je n'oublie pas de remercier aussi Alexandre, Matthieu,

Vincent, Léo, Florent, Florian, et plus particulièrement Yesmine Asses pour ses nombreuses aides, sa sincère sympathie et son amitié.

Je voudrais associer à ces remerciements tous mes amis : Sadjia, Sabrina, Sonia, Samira, Rima, et Mohamed pour le soutien qu'ils m'ont témoigné et leur disponibilité dans les durs moments. Je n'oublie pas mes anciens collègues : Ahlem, Nassima, Amina, Sabrina, Raouf et Chakib pour leur soutient constant et leur sympathie. Les listes sont longues, mais que chaque personne de mon entourage qui a su m'aider ou m'apporter quelque chose, trouve ici l'expression de ma sincère amitié.

Je fini par remercier continuellement ma famille, plus particulièrement mes chers parents, ma sœur et mon frère pour leur perpétuel soutien, leurs précieux aides et conseils et leurs nombreux encouragements. Sans vous je ne serais jamais arrivée à réaliser ce travail et avancer dans ma vie. Je n'oublie pas de saluer et de remercier mes tantes et mon oncle pour leur présence et leur assistance.

SOMMAIRE

ABREVIATIONS

1D: Unidimensionnel

2D: Bidimensionnel

3D: Tridimensionnel

ACP: Analyse en composantes principales

AD : analyse discriminante

ADM : Analyse Discriminante Multifactorielle

API: Application Programming Interface

AR: Activité Relative

AWT: Abstract Window Toolkit

CADD: Computer Assisted Drug Design

CCK: Cholécystokinine

CCK_1: Récepteur périphérique de la cholécystokinine

CCK_2: Récepteur central de la cholécystokinine

CCK-8: Octapeptide de la cholécystokinine

CCK-8S: Octapeptide de la cholécystokinine Sulfatée

CNRS : Centre National de Recherche Scientifique

DOS : Disk Operatig System

EE : Entreprise Edition

FITTED: Flexibility Induced Through Targeted Evolution

GOLD: Genetic Optimization for Ligand Docking

GUI : Graphical User Interface

IC_{50} : Concentration nécessaire à inhiber 50% de l'activité biologique d'un composé.

IDE: Integrated Development Environment

JDK : Java Development Kit

JRE : Java Runtime Environment

JVM : Java Virtual Machine

J2ME : Java 2 Micro Edition

LORIA : Laboratoire Lorrain de Recherche en Informatique et ses Applications

LXR : Liver X Receptors

MC: Monté Carlo

ME : Micro Edition

MM: Molecular Mechanic (Mécanique Moléculaire)

MSSH: Molecular Simulation using Spherical Harmonics

NAMD: NAnoscale Molecular Dynamics

OMEGA: Optimized Molecular Ensemble Generation Application

PC: Personal Computer

PDB : Protein Data Bank

PSA : Polar Surface Area

QSAR: Quantitative Structure-Activity Relationships

RCPG(s): Récepteurs Couplés aux Proteines G(stimulatrices)

RM : Regression multilinéaire

SBDD : Structure based Drug Design

SDF : Structure Data File

SE : Standard Edition

SMS: Sting Millenium Suite

SVM : Support Vector Machine

VMD: Visual Molecular Dynamics

VSM-G: Virtual Screening Manager for Grids

INTRODUCTION GENERALE

La conception et la prédiction de nouvelles molécules à visées thérapeutiques est un domaine d'actualité très prisé par les firmes pharmaceutiques et les établissements académiques. Cependant, la recherche dans ce domaine, très exigeant, est très onéreuse et incite les chimistes, les biologistes, les informaticiens, les mathématiciens et les pharmacologues à unir leurs efforts autour de la même problématique, en y apportant chacun un éclairage différent.

Le développement d'un médicament passe par plusieurs étapes et utilise différentes techniques dont le criblage à haut débit existant sous deux types qui diffèrent par les outils utilisés : le criblage expérimental et le criblage virtuel.

Le criblage réel ou expérimental consiste en la sélection parmi une diversité moléculaire étendue, les meilleures molécules qui possèdent les propriétés biologiques les plus intéressantes. Cette sélection est effectuée au moyen d'un test *in vitro*. Ces expériences sont à présent robotisées et régies par des logiciels adéquats. Cependant, cette technique de criblage réel reste très onéreuse, autant par le nombre de molécules à tester que par le coût du test qui est estimé à un dollar par molécule.

Le criblage moléculaire *in silico,* appelé aussi le screening virtuel, de banques de composés chimiques de taille importante est une alternative aux multiples essais coûteux qu'utilise la technique du criblage réel ou expérimental, et ce en utilisant des moyens informatiques qui sont de moins en moins chers et de plus en plus performants. Ce type de criblage utilise les méthodes de la modélisation moléculaire qui se développent de manière spectaculaire. Cette technique de criblage virtuel vise à collecter, dans une chimiothèque, des molécules complémentaires d'un site actif de la cible biologique, susceptibles de l'activer ou de l'inhiber.

Cette technique dont il est question dans nos travaux contribue, p*ar une évaluation rationnelle de l'activité biologique des molécules,* à épargner le coût que nécessiterait une expérimentation "aveugle". En effet, grâce aux progrès de la biologie moléculaire et de l'informatique, les chercheurs parviennent à simuler l'action de substances thérapeutiques,

d'où la naissance des techniques du «Drug Design». Dans la recherche des candidats-médicaments, le drug design utilisant les méthodes de la modélisation moléculaire comprend deux types d'approche : celle du « ligand-based » et celle du « receptor-based ». L'approche « ligand-based » utilise des techniques de calcul basées sur la connaissance du ligand, comme le QSAR (Quantitative Structure Activity Relationships) qui consiste à disposer d'une liste de molécules présentant une activité sur une cible donnée et ensuite à élaborer des modèles statistiques permettant de prédire l'activité d'une nouvelle molécule ou de proposer une structure de molécule plus active. L'autre approche : « receptor-based », basée sur la connaissance de la cible biologique, utilise les techniques du criblage virtuel à haut.

Dans le but d'améliorer l'efficacité de la recherche du candidat médicament, le développement de plateformes fonctionnant sur grilles de calcul et pouvant effectuer des criblages virtuels sur plusieurs cibles biologiques s'avère nécessaire et est en plein essor.

Les travaux entrepris et présentés dans ce mémoire, entrent dans le cadre de la contribution au développement de la plateforme logicielle de screening virtuel VSM-G (Virtual Screening Manager for Grids)[(1)]. Comme son nom l'indique, VSM-G traite les calculs liés au screening virtuel sur des bases de données de taille importante. Cependant, cette plateforme n'utilise que la technique du SBDD (Structure based Drug Design), c'est à dire le criblage qui est basé sur le docking moléculaire. Notre contribution au développement de la plateforme VSM-G consiste en l'introduction des outils statistiques QSAR (Quantitative Structure Activity Relationships), en amont du criblage. Notons que ces outils ont été antérieurement utilisés par notre équipe dans le cadre de la recherche d'anti-inflammatoires non stéroïdiens[(2,3)].
L'implémentation de ces outils statistiques fera de VSM-G un outil complet en termes de méthodes du drug design.

A la suite de cette introduction générale qui vise à présenter la problématique et les objectifs de nos travaux, notre manuscrit comporte trois chapitres suivis d'une conclusion générale et de trois annexes.

Dans le premier chapitre, nous présentons brièvement les méthodes de la modélisation moléculaire et, en particulier, celles du Drug Design utilisées dans ce travail. La plateforme VSM-G y sera également présentée.

Le deuxième chapitre concerne la présentation de l'outil de programmation utilisé, à savoir le JAVA qui est le langage d'écriture de la plateforme VSM-G. Nous y présenterons aussi la maquette que nous avons développée en implémentant les outils QSAR. Enfin, une validation du prototype sera également effectuée.

Le troisième chapitre sera consacré à l'application du prototype sur quelques bases de données de taille importante dont celle contenant des antagonistes des récepteurs périphériques de la cholécystokinine (CCK). La CCK qui est une hormone neuropeptidique, impliquée dans un grand nombre d'actions physiologiques et physiopathologiques[4-5] sera aussi décrite dans ce chapitre.

Nous terminons par une conclusion générale qui porte sur les principaux apports de nos travaux. Nous discuterons aussi des perspectives que nous envisageons pour cette plateforme de screening virtuel à haut débit.

Nous reportons, en annexe 1, le User Guide qui est un manuel qui aide et explique l'utilisation de toutes les interfaces graphiques créées pour la chaîne de programmes implémentés.

L'annexe 2 comprend les structures des molécules de la base de données comportant les ligands des récepteurs LXR (Liver X Receptors) , qui a servi à valider notre maquette.

Dans l'annexe 3, nous reportons le tableau comportant les structures des antagonistes des récepteurs périphériques de la CCK et les valeurs des activités biologiques.

REFERENCES BIBLIOGRAPHIQUES

1- Beautrait, A., Leroux, V., Chavent, M., Ghemtio, L., Devignes, MD., Smail Tabbone, M., Cai, W., Shao, X., Moreau, G., Bladon, P., Yao, J., et Maigret, B., *Multiple-step virtual screening using VSM-G: overview and validation of fast geometrical matching enrichment*, J Mol Model, 14, 135-48, **2008**

2- Bouaziz Terrachet, S., Mémoire de Magister, Conception d'inhibiteurs sélectifs de l'activité enzymatique de la cyclo-oxygénase-2 : COX-2, Alger, USTHB, N° d'ordre : 14/**2006**-M/CH.

3- S. Taïri–Kellou, S. Bouaziz–Terrachet, B. Maouche and G. Moreau, *2D–QSAR Autocorrelation Study on Selective COX–2 Inhibitors,* Internet Electron. J. Mol. Des. **2008**, *7*, 161-185

4- Taïri-Kellou, S., Thèse de Doctorat d'état, Conception Assistée par Ordinateur des Complexes ligand/récepteur de la Cholécystokinine, Alger, USTHB, N° d'ordre : 04/**2002**-E/CH.

5- Asses, Y., Mémoire de Magister, Modélisation et Etude Structurale de Ligands du Récepteur périphérique de la Cholécystokinine : Screening Virtuel, Alger, USTHB, N° d'ordre : 13/**2006**-M/CH.

CHAPITRE I

LE DRUG DESIGN
LES OUTILS QSAR
ET
LA PLATEFORME
VSM-G

I. INTRODUCTION

La modélisation, au sens large du terme, consiste à construire un modèle artificiel, le plus souvent mathématique, d'un phénomène réel pour l'utiliser ensuite à des fins prédictives. Un modèle essaie d'illustrer la réalité de la manière la plus précise possible mais en l'état actuel de la technologie, c'est à dire évolutive, il reste perfectible.

La modélisation moléculaire est l'élaboration des modèles dans le cadre de la chimie c'est-à-dire des systèmes d'intérêt chimique. Cette méthode utilisant l'outil informatique et les nombreuses possibilités de calcul théorique est appelée également conception assistée par ordinateur. De ce fait, elle est considérée aujourd'hui comme une nouvelle technique de recherche et de compréhension des phénomènes physico-chimiques des molécules. Son progrès, notamment en biologie moléculaire, en informatique et en pharmacochimie, permet de concevoir des composés actifs et bénéfiques pour l'homme. En effet, les chercheurs tentent de développer les méthodes de la modélisation moléculaire comprenant celles du drug design qui ont pour objectif la conception de médicament.

II. METHODES DU DRUG DESIGN

Les méthodes du Drug Design connaissent aujourd'hui un développement remarquable. En effet, les premières études s'effectuaient, par faute de moyens, en estimant les corrélations statistiques entre les descripteurs structuraux et les activités biologiques des systèmes étudiés. L'évolution rapide des moyens informatiques a permis l'utilisation de processeurs à grande vitesse et a conduit à ce qui est actuellement appelé « conception de médicament assistée par ordinateur » ou encore « Computer Assisted Drug Design » (CADD). Ainsi, le drug design peut être défini comme étant une approche visant à rechercher des molécules thérapeutiques par conception *in silico* sur la base d'informations expérimentales accessibles.

Dans le vaste domaine du drug design deux types d'approches principales sont actuellement connues : l'approche dite « receptor-based » et celle dite « ligand-based ». La première approche repose sur l'étude du docking moléculaire et nécessite la connaissance de la structure spatiale de la cible biologique et également celle du site de liaison. Le docking consiste à positionner, de manière optimale, le ligand dans le site de liaison du récepteur et ensuite étudier les interactions cruciales entre le ligand et le récepteur. La deuxième approche repose

sur l'étude des relations entre l'activité d'une molécule et ses propriétés structurales. Elle utilise des méthodes statistiques sur les données structurales et biologiques d'une série de ligands; ce qui conduit à l'élaboration de modèles statistiques entre l'activité et la structure. Cette technique est appelée communément le QSAR (Quantitative Structure-Activity Relationships).

II.1 LE QSAR (Quantitative Structure-Activity Relationships)

Le QSAR représente une des plus anciennes méthodes dans le drug design puisque ses débuts remontent à la fin du XIXe siècle. En effet, Ehrlich introduit le QSAR comme une étude réalisée sur la base unique d´un ensemble de ligands reconnus affins à une protéine[(1)].

Le QSAR est le résultat d'une analyse statistique qui permet de déterminer des liens quantitatifs voire qualitatifs entre les structures physico-chimiques et les activités biologiques des molécules. Les modèles statistiques établis permettent de prévoir l'activité d'une nouvelle molécule non encore testée. Ils permettent également de prévoir la structure d'une molécule susceptible d'avoir une meilleure activité.

Pour effectuer une étude de relation entre la structure et l'activité, on doit disposer d'un ensemble de molécules représentées par un ensemble de propriétés moléculaires (biologiques, structurales, chimiques, physiques, électroniques…etc). Ainsi, le QSAR recherche la relation entre la structure d'une molécule et son activité ; cette relation traduit le modèle mathématique, ayant la formulation générale suivante :

Activité = f (Structure moléculaire)

Les études QSAR peuvent être effectuées dans plusieurs dimensions, variant entre une et six dimensions [(2-7)]. Elles utilisent des outils statistiques différents qui peuvent être : l'analyse en composantes principales, la régression multilinéaire et l'analyse discriminante, pour ne citer que ceux utilisés dans le cadre de ce mémoire. En général, pour effectuer une étude de relation entre la structure et l'activité des molécules, il est nécessaire de générer préalablement des propriétés structurales et/ou physico-chimiques appelées aussi des descripteurs. Les descripteurs considérés dans nos travaux sont les vecteurs d'autocorrélation déterminés par l'application de la méthode d'autocorrélation [(8)] que nous présentons ci-dessous.

A) L'autocorrélation et génération de vecteurs d'autocorrélation

Dans le cadre de cette méthode, une molécule est représentée par une structure bidimensionnelle ou tridimensionnelle, et est définie par un certain nombre de propriétés moléculaires. Cette technique consiste à calculer des vecteurs 2D issus de la transformation d'une matrice de connectivité permettant des études de reconnaissance de forme. Les vecteurs d'autocorrélation sont utilisés comme descripteurs moléculaires en appliquant la notion mathématique d'une fonction d'autocorrélation à la structure topologique des molécules. Chaque composante du vecteur d'autocorrélation est calculée par la formule suivante:

$$A(d) = \sum_{ij} PiPj \qquad (1)$$

Où A est le coefficient d'autocorrélation attribué aux atomes *i* et *j*, *Pi* est une propriété atomique, et *d* la distance topologique entre les deux atomes. Le vecteur d'autocorrélation est une entité mathématique qui montre comment est distribuée une propriété physique sur la structure moléculaire.

Pour illustrer le concept de vecteurs d'autocorrélation, considérons la structure 2D (figure 1) représentée par les atomes désignés par a, b, c et d.

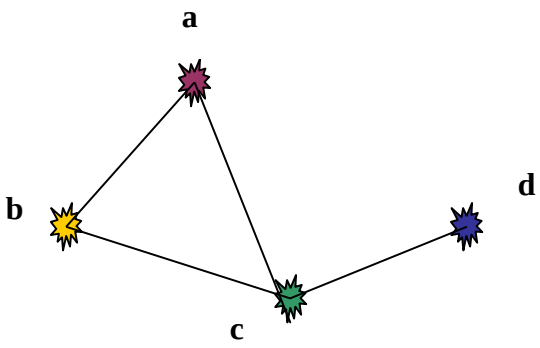


Figure 1 : Structure 2D d'un assemblage d'atomes

On écrit toutes les paires d'atomes : aa, bb, cc, dd, ab, ac,…, bb, bc, bd,….dans la colonne de la matrice correspondant aux trajets les plus courts entre les atomes.

0	1	2	3	4	5	6
aa	ab	ac				
bb	ac	bd				
cc	bc					

dd cd

Ensuite, on fait intervenir la propriété f d'un atome *f(atome)*, et on remplace chaque propriété par le produit f(b).f(c), ce qui donnera le tableau suivant :

0	1	2	3	4
f(a).f(a)	f(a).f(b)	f(a).f(d)		
f(b).f(b)	f(a).f(c)	f(b).f(d)		
f(c).f(c)	f(b).f(c)			
f(d).f(d)	f(c).f(d)			

Puis on somme chaque colonne, ce qui donne :

C0 C1 C2

qui sont les vecteurs d'autocorrélation de la propriété f sur le graphe moléculaire.

On voit que selon la taille de la molécule, on aura pour ces vecteurs d'autocorrélation un plus ou moins grand nombre de composantes. Cette diversité pose des problèmes (corrélations plus ou moins fortes entre les rangs ou les atomes) dans les techniques, telle que l'Analyse en composantes principales (ACP), qui doivent travailler sur des tableaux rectangulaires sans « données manquantes » (ou sans perte d'informations).

Pour pallier à cet inconvénient, au lieu de ne considérer que les trajets les plus courts entre les paires d'atomes, on considère tous les trajets possibles avec « retour sur ses pas », en repassant plusieurs fois sur un même atome, etc. Par exemple, on peut sur la figure 1, imaginer le trajet aa de longueur 0, mais aussi le trajet aba de longueur 2, ou abca de longueur 3, ou abcdca de longueur 5, et bien d'autres encore. La paire aa figurera donc dans les colonnes 0, 2, 3, 5 (l'information apportée par la composante est partiellement partagée).

Ainsi, même les plus petites molécules fournissent un nombre de composantes théoriquement illimité. En pratique, on se limitera aux 10 ou 20 premiers (le programme est limité à 20), et le calcul ne deviendra pas difficile pour autant. En effet, désignons par M la matrice de connectivité du graphe moléculaire, qui a des termes égaux à 1 ou 0 selon qu'il y a ou qu'il n'y a pas de liaison (indépendamment de la nature chimique simple, double ou triple) entre atomes

$$M = \begin{pmatrix} & a & b & c & d \\ a & 0 & 1 & & \\ b & 1 & & & \\ c & & & & \\ d & & & & \end{pmatrix} \qquad (2)$$

M^n étant la puissance énième de cette matrice, les termes de M^n sont des nombres entiers, M^n (i, j) = x signifie qu'entre les atomes i et j il y a x trajets, de toutes sortes mais différents, de longueur n. par suite, il est donc facile de faire figurer le produit f(i).f(j) x fois dans la colonne n.

La génération de ces vecteurs d'autocorrélation va permettre de faire une analyse en composantes principales, afin de calculer les nouvelles composantes des molécules portées par les facteurs principaux dont les premiers apportent le maximum d'informations.

B) Méthodes statistiques

1- L'analyse en composantes principales

L'analyse en composantes principales (ACP) [(9)], introduite en 1901 par K. Pearson et développée par H. Hotelling en 1933, est une méthode très puissante pour explorer la structure des données. Chaque donnée étant représentée dans un espace à p dimensions, l'ensemble des données forme un "*nuage de n points*" dans R^p. Le principe de l'ACP est d'obtenir une représentation approchée du nuage dans un sous-espace de dimension faible k, par projection sur des axes bien choisis.

Un ensemble dans R^p étant choisi (en général normalisé par l'utilisation de variables centrées réduites), les k axes principaux sont ceux qui maximisent l'"*inertie*" du nuage projeté, c'est-à-dire la moyenne pondérée des carrés des distances des points projetés à leur centre de gravité. Les composantes principales sont les n vecteurs ayant pour coordonnées celles des projections orthogonales des n éléments du nuage sur les k axes principaux.

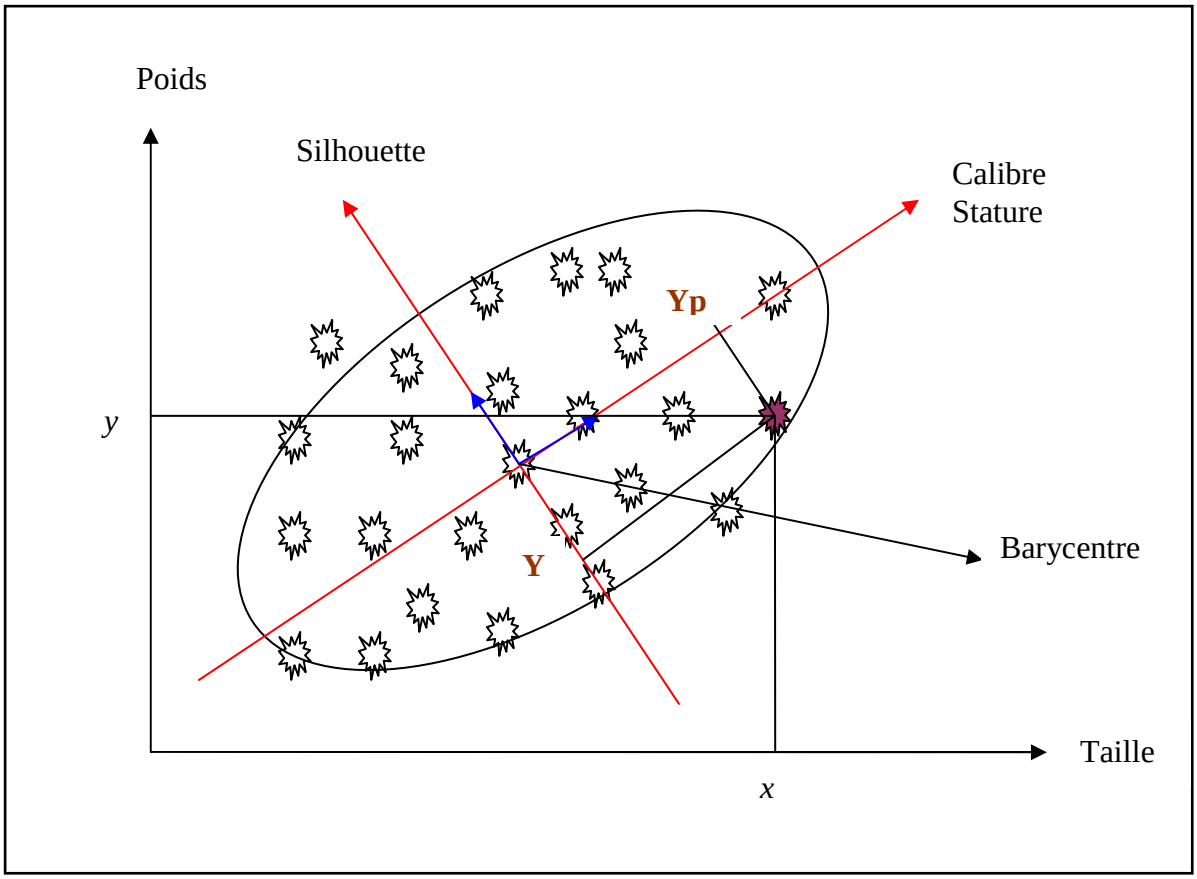


Figure 2 : Représentation des deux axes d'une ACP

En d'autres termes, l'ACP consiste à translater les données de manière à ce que le barycentre du nuage de points soit l'origine du repère (centrage) (figure 2). Ensuite, toutes les composantes qui décrivent le nuage sont analysées et toutes celles qui sont nulles ou presque nulles (réduction) sont éliminées. La diagonalisation de la matrice d'autocorrélation qui devient carrée conduit aux valeurs et aux vecteurs propres.

Dans notre cas, l'ACP consiste à rechercher les ressemblances au sein d'un ensemble de N molécules décrites par ***K*** variables. Ces dernières sont représentées par des descripteurs moléculaires qui sont souvent inter-corrélés. L'ACP consiste donc à utiliser des combinaisons de plusieurs descripteurs théoriques jusqu'à obtenir les combinaisons principales d'un nombre raisonnable ***n*** de descripteurs qui renferment le plus d'informations sur la structure moléculaire. Les ***N*** molécules seront représentées dans un espace non plus à ***K*** mais à ***n*** (<<***K***) dimensions. Le nouveau repère du système est représenté par ***n*** axes principaux ou composantes principales, qui ne sont plus inter-corrélés et où les molécules sont décrites avec les descripteurs les mieux corrélés à l'activité expérimentale. Ainsi, deux molécules proches l'une de l'autre sur le graphe 3D auront des propriétés qui varient dans le même sens, et il sera

alors possible d'interpréter la formation du nuage de points comme un groupe de molécules possédant des propriétés communes.

Pour donner une approche mathématique à la technique de l'ACP, on doit disposer d'une matrice rectangulaire *M* à *N* lignes (représentant le nombre de molécules) et *K* colonnes (représentant le nombre de variables d'origine)

$$X = \{ A(d)_{ij} ; 1 \leq i \leq N ; 1 \leq j \leq K \} \quad (3)$$

Cette matrice peut être assimilée à un nuage de N points définis par K variables. Pour que toutes les propriétés aient une importance égale et afin que la distance entre les atomes ne dépende plus des unités de mesures, on définit la matrice diagonale suivante :

$$M_{1/\sigma^2} = \begin{bmatrix} 1/\sigma_1^2 & 0 & \cdots & 0 \\ 0 & 1/\sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1/\sigma_k^2 \end{bmatrix} \quad (4)$$

Où σ_j est l'écart type du vecteur d'autocorrélation *j* donnée par la relation suivante :

$$\sigma_j = \sqrt{\frac{\sum_{i=1}^{N} (A(d)_{ij} - \overline{A}(d)_j)^2}{N}} \quad (5)$$

La variance (inertie) qui résulte du sous espace engendré par les *k'* premiers vecteurs propres de la matrice d'inertie est la somme de leurs valeurs propres λ_i et aussi le carré de l'écart type σ

$$I_{K'} = \sigma^2 = \sum_{i=1}^{K'} \lambda_i \quad (6)$$

Chaque valeur propre mesure la part de variance expliquée par l'axe factoriel correspondant.
La matrice de variance-covariance entre les variables des données centrées réduites est la suivante :

$$C_{xx} = \frac{1}{N}\left[(X - I_N g^t)M_{1/\sigma}\right]^t\left[(X - I_N g^t)M_{1/\sigma}\right] \quad (7)$$

Où I_N est la matrice identité

Les axes principaux sont les vecteurs propres *a* de la matrice de $C_{xx}M_{1/\sigma}$ qui maximisent l'inertie du nuage projeté, c'est-à-dire la moyenne pondérée des carrés des distances des points projetés à leur centre de gravité.

On associe à chaque axe principal *a* un facteur principal défini par

$$F = M_{1/\sigma}a \quad (8)$$

a est déterminé par les valeurs propres C_{xx} $M_{1/\sigma}$ correspondante et *F* est déterminé par les vecteurs propres de $M_{1/\sigma}C_{xx}$

Les composantes principales sont les ***n*** vecteurs ayant pour coordonnées celles des projections orthogonales des ***N*** éléments du nuage de points sur les ***k*** axes principaux. Ce sont les variables C_i définies par les facteurs principaux :

$$C_i = XF_i \quad (9)$$

C_i est le vecteur renfermant les coordonnées des projections des individus sur l'axe défini par a_i. Ce sont donc des combinaisons linéaires des vecteurs d'autocorrélations ayant des variances maximales.

2- La régression multilinéaire

La **régression** est une méthode de prévision mathématique très utilisée dans divers domaines.

La **régression linéaire multiple** [(10)] est une généralisation à *p* variables explicatives, de la régression linéaire simple. Elle consiste à établir une relation linéaire entre une variable quantitative à expliquer (endogène) ***Y,*** et un ensemble de variables également quantitatives (explicatives ou exogènes), $X_1, X_2, ...X_p$. Quand on dispose de ***n*** observables et d'une variable X_i, on dit que c'est une régression linéaire mais quand on dispose de ***n*** observables et de ***p*** variables : $X_i^1, X_i^2 ....X_i^p$, on dit que c'est une régression multilinéaire et dans ce cas, ***Yi*** est donnée en fonction des variables $X_i^1, X_i^2 ....X_i^p$ par la formule :

$$Y_i^{cal} = a_1 X_i^1 + a_2 X_i^2 + ... + a_p X_i^p + E \quad (10)$$

Avec

X : la matrice constituant les variables explicatives

Y : le vecteur constituant les variables endogènes

n : le nombre d'observations

p : le nombre de variables explicatives

X_i^j : la variable de la variable X^j sur l'observation i

Y_i^{cal} : la valeur de la variable Y^{cal} sur l'observation i

a_i : les coefficients de régressions des variables X_i

E : vecteur aléatoire résiduel, il représente l'erreur aléatoire sur Y_i ; $E = Y_i^{cal} - Y_i^{obs}$

Les cœfficients $a_1,\ldots..a_p$ sont estimés grâce à la méthode des moindres carrés et ce dans le but de calculer la valeur de Y_i^{cal}, qui est dans notre cas la valeur de l'activité (calculée ou prédite), à partir de la relation (10).

❖ Critères statistiques

- Le coefficient de corrélation multiple R est donné par la relation suivante :

$$R = \sqrt{1 - \frac{\sum_i^n (Y_i^{cal} - Y_i^{obs})^2}{\sum_i^n (Y_i^{cal} - \sum_i^n \frac{Y_i^{obs}}{n})^2}} \qquad (11)$$

Où :

Y_i^{cal} : représente la valeur de l'activité calculée ou prédite

Y_i^{obs} : représente la valeur de l'activité mesurée ou expérimentale

n : représente le nombre de molécules soumises à la recherche de la régression (c'est le nombre d'observations)

Il important de noter que ce coefficient devient significatif au-delà de $\frac{\sqrt{3}}{2} (\approx 0.86)$

- Le calcul de la valeur du R^2 permet de donner une signification à la régression

La valeur du R^2 est tirée à partir de relation suivante :

$$F = \frac{R^2}{1 - R^2} - \frac{n - K_2}{K_2 - K_1} \qquad (12)$$

K_1 étant le nombre de paramètres dans l'équation de référence

K_2 étant le nombre de variables indépendantes dans l'équation de régression considérée
F étant la variable de *Fischer Snedecor* qui représente la variance de la régression, ou le taux de confiance. Cette valeur doit être supérieure à la valeur de l'indice de *Fischer Snedecor* $F_{(tabulé)}$ qui est tirée des tables statistiques. Déterminer $F_{(tabulé)}$ peut se faire en utilisant la table statistique donnant la distribution de la variance de la régression.

$$F_{(tabulé)} = F_{\gamma'}^{\gamma} = F_{n-p-1}^{p-1} \qquad (13)$$

p : est le nombre de variables indépendantes

n : le nombre de molécules soumises à la recherche de régressions

L'incertitude basée sur la moyenne arithmétique des valeurs observées et calculées pour une même série de composés est définie par la déviation standard ou encore appelée écart type noté ***s***. Plus la valeur de ***s*** est petite, meilleure est la corrélation obtenue. La valeur de ***s*** est calculée à partir de la relation suivante :

$$S = \sqrt{\frac{\sum_{i}^{n} (Y_i^{cal} - Y_i^{obs})^2}{n - K - 1}} \qquad (14)$$

K étant le nombre de variables indépendantes

- *Validation croisée*

Valider la technique de la régression multilinéaire exige, pour de gros systèmes, du temps et une mémoire m²

achine considérable. Pour remédier à cela, *Cramer* essaya, en 1988, d'ajuster cette méthode en introduisant la notion de la validation croisée (cross-validation)[(11)]. Cette validation croisée consiste à enlever aléatoirement des observations à chaque étape de la régression afin d'obtenir un nouveau modèle. Ce modèle engendre une nouvelle équation de corrélation permettant la prédiction d'une nouvelle activité biologique. Ainsi, cette validation croisée permet d'évaluer le pouvoir prédictif du modèle de la régression et donc de le valider.

La valeur du coefficient de la validation croisée R_{cv}^2 est donnée par la relation suivante :

$$R^2_{cv} = 1 - \frac{\sum_1^n (Y_i^{obs} - Y_i^{cal})^2}{\sum_1^n (Y_i^{obs} - \overline{Y}_i^{obs})^2} \qquad (15)$$

Où :

Y_{obs} est l'activité observée

Y_{cal} est l'activité calculée

$\overline{Y_{obs}}$ est la moyenne de la somme des activités observées

Il est important de noter que plus la valeur de R^2_{cv} est proche de R^2, meilleure est la prédiction du modèle obtenu, et pour une valeur proche de zéro le modèle ne possède pas de pouvoir prédictif.

3- L'analyse discriminante

L'analyse discriminante (AD) [(12)] est une technique statistique qui vise à décrire, expliquer et prédire l'appartenance d'un ensemble d'observations (individus, exemples, ...), à des groupes prédéfinis (classes, modalités de la variable à prédire, ...), à partir d'une série de variables prédictives (descripteurs, variables exogènes, ...).

Contrairement à la méthode de régression multilinéaire décrite précédemment, l'AD permet de déterminer des relations qualitatives en combinant les variables non corrélées provenant de l'ACP.

Ces relations apparaissent sous forme d'appartenance ou de non appartenance à un sous-ensemble de molécules classées sur la base de leur activité biologique, soit active soit inactive. Ce crible d'activité permet de classer les molécules selon un seuil d'activité en transformant les paramètres quantitatifs mesurés en variables qualitatives.

Cette technique, initiée par Fisher en 1936, part d'une matrice **X** de données observées (individus x variables) dont les éléments sont identifiés dans une (et une seule) des *k* classes possibles. L'idée de Fisher était de créer une méthode pour choisir entre les combinaisons linéaires des variables, celle qui maximise l'homogénéité de chaque classe.

Dans le but de répartir les molécules, selon leur valeur d'activité, dans deux classes distinctes, on doit disposer d'un lot d'apprentissage, servant à la construction du modèle, et d'un lot d'essai. Les molécules appartenant au lot d'apprentissage doivent avoir des activités connues, leurs structures doivent être tirées de différentes familles chimiques et elles doivent

présenter une diversité biologique. Quant aux molécules appartenant au lot d'essai et dont les activités sont connues, elles vont servir pour tester le modèle obtenu.

L'analyse discriminante, d'un point de vue statistique, passe par le calcul du centre de gravité d'un ensemble de *N* points, chaque point représentant une structure. Ce calcul est donné par la relation suivante :

$$g^t = \frac{1}{M}\left(\sum_{i=1}^{N} P_i f_{i1}, \sum_{i=1}^{N} P_i f_{i2}, ..., \sum_{i=1}^{N} P_i f_{ik'}\right)^t \qquad (16)$$

Avec :

p_i : poids statistique qui vaut 1 pour donner la même importance à toutes les molécules

$f_{ik'}$: sont les composantes des facteurs principaux

M : le poids moléculaire total des N molécules

Le centre de gravité d'une classe Q est donné par la relation suivante :

$$g^t = \frac{1}{P_q}\left(\sum_{i=1}^{N} P_i f_{i1}, \sum_{i=1}^{N} P_i f_{i2}, ..., \sum_{i=1}^{N} P_i f_{ik'}\right)^t \qquad (17)$$

p_q étant le poids moléculaire total de la classe Q

Le classement de nouveaux points se fait sur la base de la proximité relative à l'une des classes de la partition donnée. La distance d'un point i à une classe Q ayant un centre de gravité q est définie par la relation suivante :

$$d^2(i,Q) = (f_i - q)^t M_{1/\sigma} (f_i - q) \qquad (18)$$

Où $(f_i - q)^t$ représente la transposé de $(f_i - q)$ et f_i sont les facteurs principaux.
On peut illustrer l'analyse discriminante comme suit :

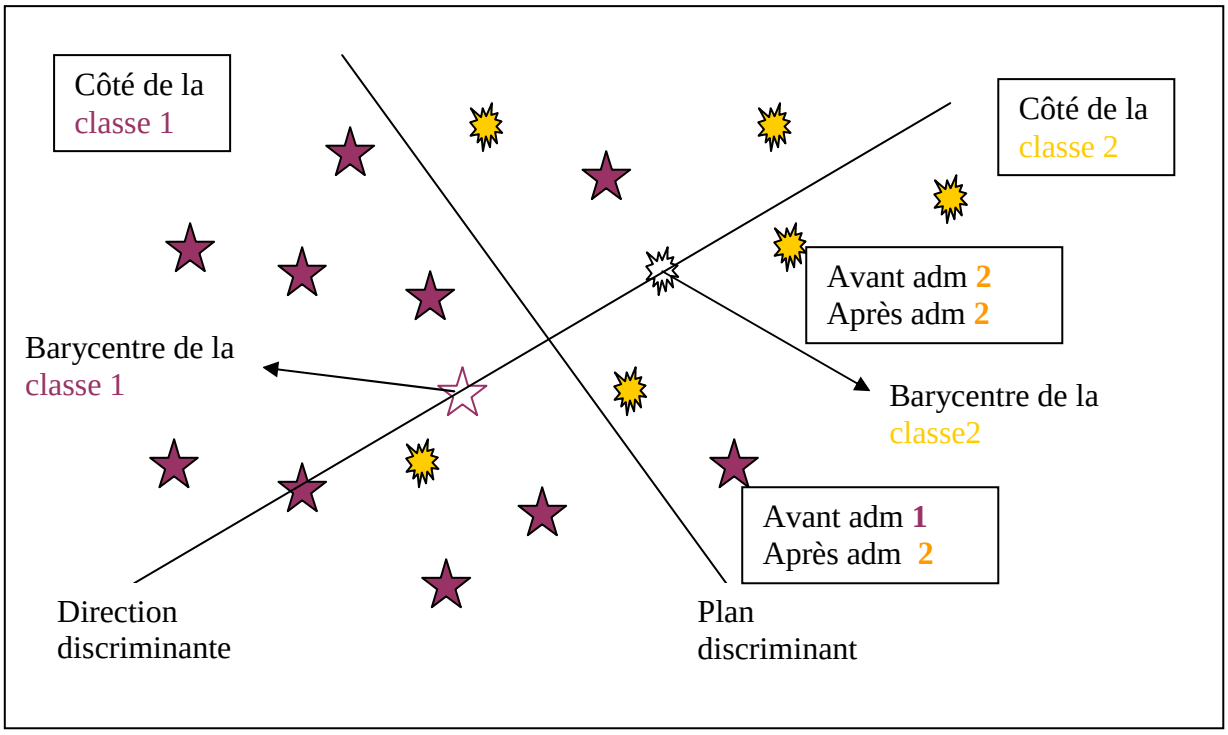


Figure 3 : Illustration graphique de l'analyse discriminante

Avec :

Avant adm : c'est le groupe 1 ou 2 défini par l'activité réelle.

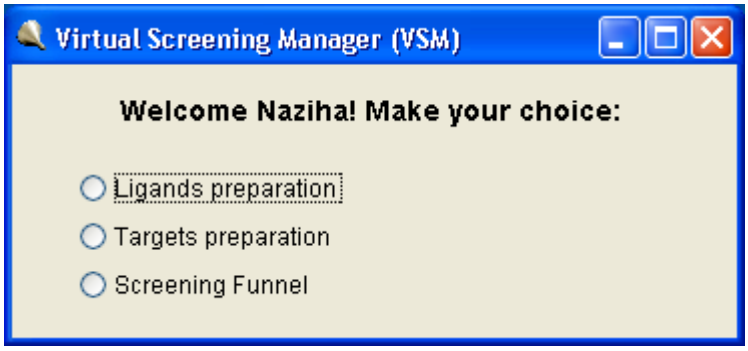
Après adm : c'est le groupe auquel la molécule se trouve rattachée par l'application de l'analyse discriminante.

III. PRESENTATION DE LA PLATEFORME VSM-G

VSM-G (Virtual Screening Manager for Grids)[13], est une plateforme logistique écrite en langage Java, développée par l'équipe de recherches du LORIA (Laboratoire Lorrain de Recherche en Informatique et ses Applications) de Nancy. Cette plateforme qui combine des outils de Drug Design, basés sur la cible biologique, a été créée pour cribler des bases de molécules de taille importante.

VSM-G a été conçu afin d'essayer de remédier aux problèmes du docking de plusieurs millions de molécules sur des centaines de cibles biologiques potentielles. Cette plateforme utilise un processus de filtrage et est dotée de plusieurs interfaces graphiques simple d'utilisation.

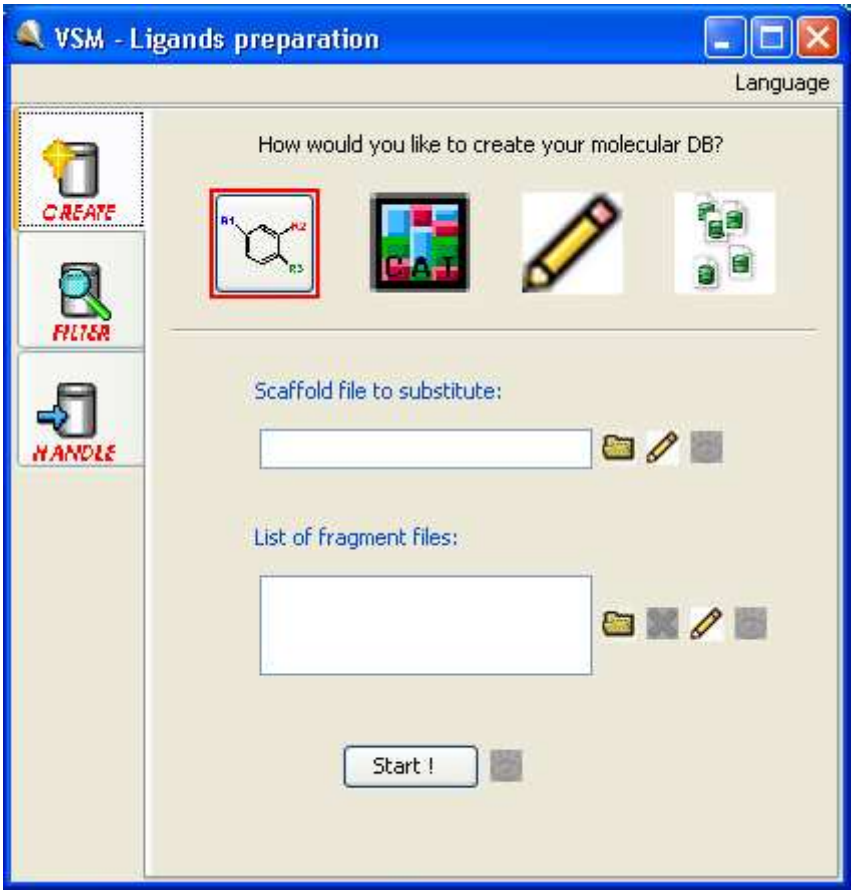
Dès le lancement de VSM-G, la fenêtre suivante s'ouvre :



- ✓ Préparer la chimiothèque (représentée par les ligands)
- ✓ Préparer les cibles biologiques (représentées par un ensemble de structures)
- ✓ Cribler virtuellement les ligands en utilisant une approche en entonnoir

1- Préparation des ligands

Un clic sur le bouton « Ligand preparation » conduit à l'ouverture de l'interface suivante :



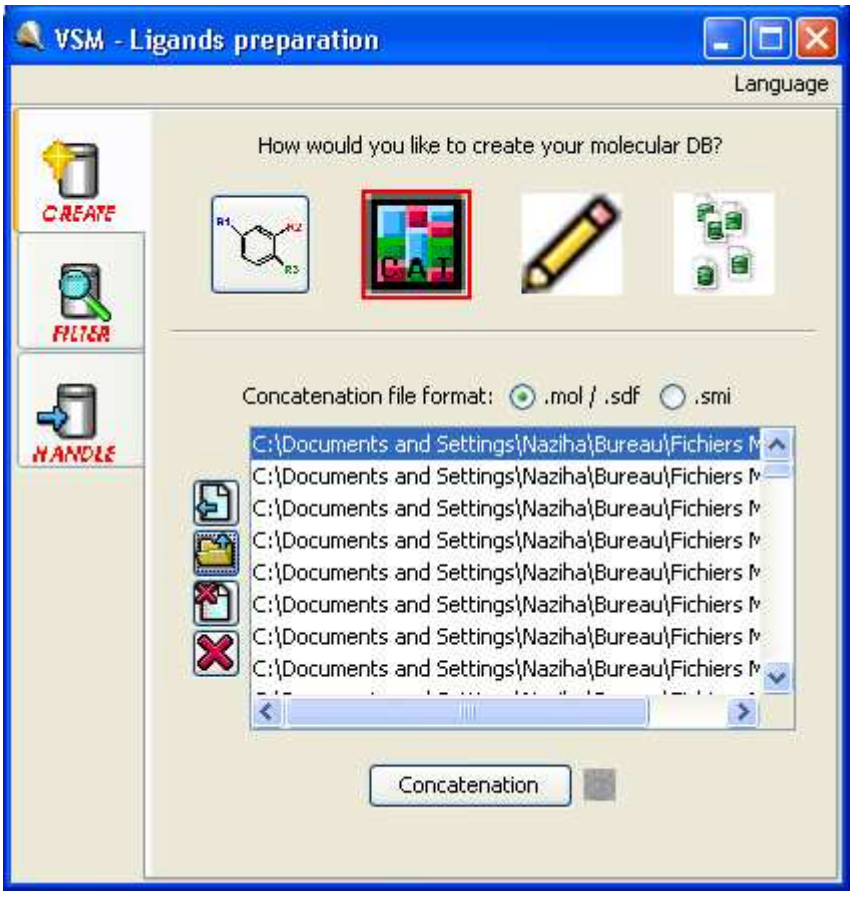
Elle comporte trois onglets permettant une gestion complète des ligands.

A) Création de bases de données moléculaires (Create)

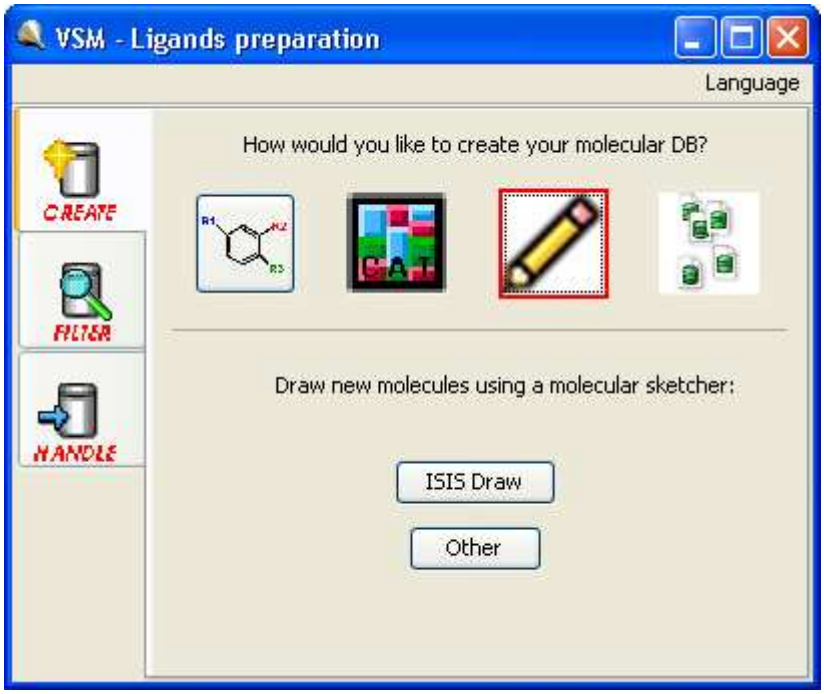
Ce module permet de créer la base de données si celle-ci n'est pas créée de manière externe au programme. Cette création peut se faire de 4 façons différentes :

α - Par chimie combinatoire virtuelle (voir fenêtre précédente) en greffant des substituants sur des squelettes de molécules

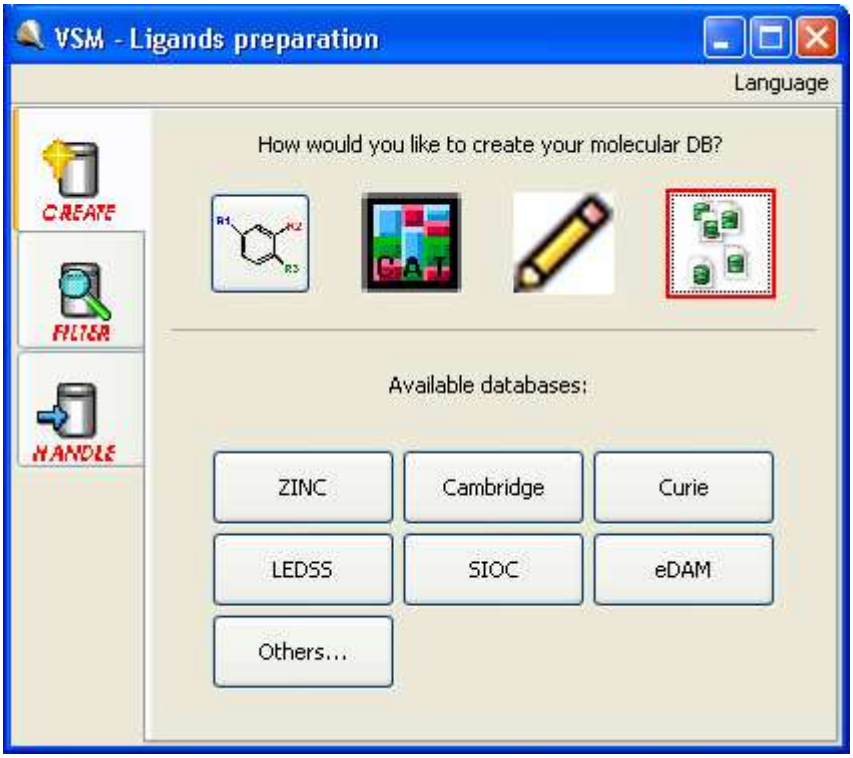
β- Par concaténation de fichiers moléculaires ; ce qui permet de fusionner plusieurs sous bases.



γ- Par saisie de molécules grâce au programme Isis Draw



δ- Par téléchargement à partir de bases de données disponibles sur le web (Zinc Data Base, Curie, Cambridge, …..) ou commerciales. Il faut noter que quelques bases de données sont proposées par VSM-G.



B) Filtrage de bases de données (Filter)

Cette option permet d'extraire des molécules spécifiques à partir de bases de données préalablement établies. Ce filtrage peut se faire au moyen d'une :

– recherche par sous-structure

– exclusion des composés toxiques

– recherche par groupement pharmacophore

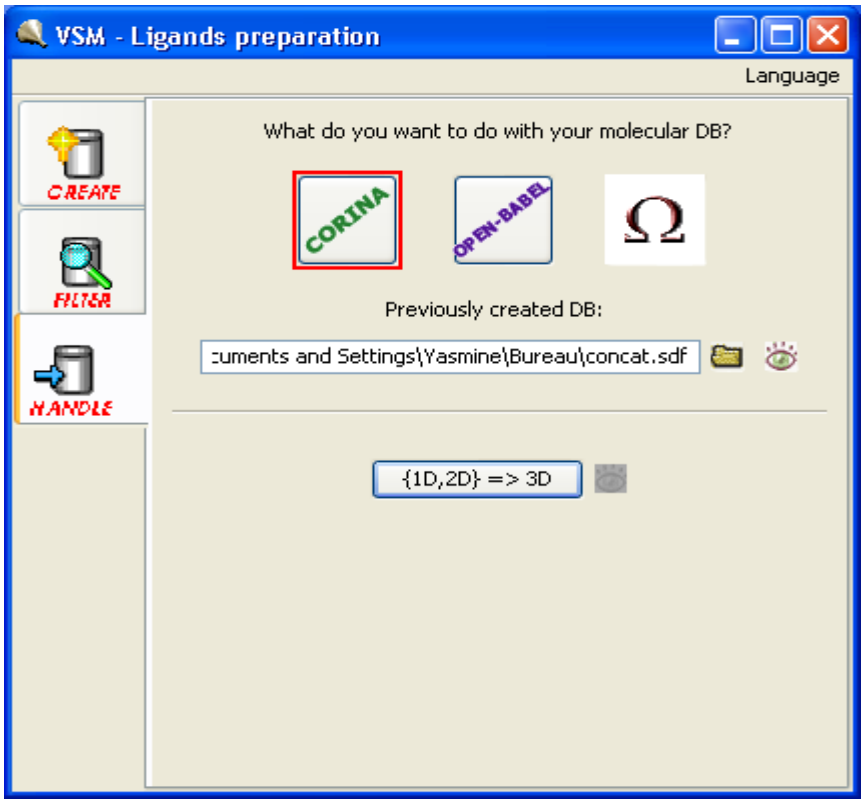
C) Manipulation de la base de données (Handle)

Cette option permet d'uniformiser les formats de fichiers de molécules provenant de chimiothèques distinctes. Ceci peut se faire au moyen de programmes implémentés :

– CORINA[14] permettant les transformations : 1D/2D au 3D

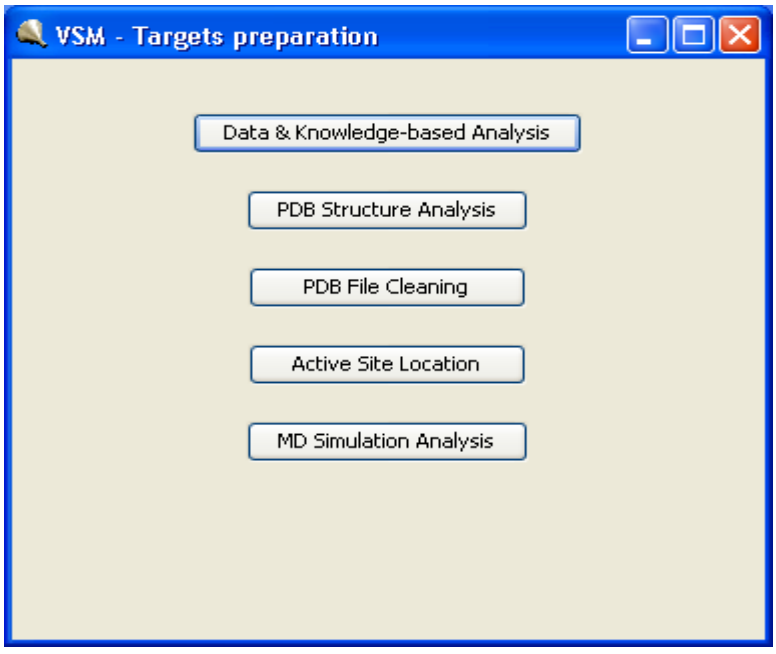
– Open-Babel[15] permettant les conversions de formats de fichiers

Le programme OMEGA[16] permet d'obtenir un échantillonnage conformationnel de chacune des molécules contenues dans la base de données



2- Préparation de la cible (Target preparation)

Dans le cas où la cible biologique est connue VSM-G peut proposer différentes opérations à effectuer sur cette dernière. Un clic sur le bouton « Target preparation » conduit à l'ouverture de la fenêtre ci-dessous :



En cliquant sur bouton « Data & Knowledge-based Analysis » le logiciel nous reconduit vers une page web donnant un exemple de base de données regroupant différentes caractéristiques chimiques, biologiques, structurales….etc, expliquant l'approche « Knowladge based » et « Data mining ».

En cliquant sur bouton « PDB Structure Analysis » VSM-G nous redirige vers un site web contenant un outil logistique d'accès gratuit et appelé Sting Millenium Suite (SMS)[17]. Cet outil est dédié à la visualisation et à l'analyse des structures 3D des protéines permettant principalement de :

- mettre à la bonne norme les fichiers PDB
- corriger la protonation de la protéine
- analyser les points de contact des acides aminés d'une structure PDB
- supprimer des molécules d'eau selon leur positionnement relatif à une interface protéine /ligand ou protéine

En cliquant sur les boutons « Active site location » et « MD Simulation Analysis », le programme VMD (Visual Molecular Dynamics)[18], permettant de faire des visualisations et des calculs de dynamique, se lance. Le premier bouton sert à faire une localisation de sites actifs sur la cible, et le deuxième permet de faire des analyses de simulation de dynamique moléculaire sur le système ligand / récepteur.

3- Entonnoir de criblage (Screening Funnel)

En cliquant sur le dernier bouton « Screening Funnel) de la fenêtre de lancement de VSM-G, l'interface suivante apparaît :

Ce module, utilisant le procédé de "l'entonnoir " comme illustré, répond essentiellement au système de criblage virtuel à haut débit par filtres successifs.

Les méthodes de calculs, présentées sous forme de logiciels, sont classées de la plus rapide mais moins efficace à la plus précise. Comme le montre la fenêtre ci-dessus, les trois étapes qui doivent être appliquées successivement sont :

- ✓ le calcul de complémentarité de surfaces : cavité/ligand, au moyen du programme de filtrage MSSH[19].
- ✓ le docking moléculaire flexible au moyen du logiciel « FITTED »[20] remplaçant GOLD[21] (qui permet de classer les molécules selon une fonction de score estimant l'énergie d'interaction ligand/récepteur). Le programme GOLD a été remplacé par FITTED car ce dernier est un produit commercialisé, mais toutefois les deux programmes présentent des similarités au niveau des fonctionnalités.

Chacune de ces étapes agit comme un filtre moléculaire excluant à chaque niveau un pourcentage de structures moléculaires.

Selon la taille de la chimiothèque, VSM-G présente l'option d'exécuter les calculs relatifs à MSSH, sur un simple PC, ou sur une grille de calcul pouvant atteindre des milliers de noeuds.

REFERENCES BIBLIOGRAPHIQUES

1- Patrick, G. L., Chimie Pharmaceutique. Bruxelles, de boeck, **2003**

2- E. Victor, Kuz'min., G. Anatoly, G. Artemenko Pavel, Polischuk., N. Eugene, Muratov I. Alexander, Hromov. Anatoly V., Liahovskiy Sergey A., Andronati., Svetlana Yu., Makan., Hierarchic system of QSAR models (1D – 4D) on the base of simplex representation of molecular structure; J. Mol. Model. 11: 457–467, **2005.**

3- N. Baurin, J. C. Mozziconacci, E. Arnoult, P. Chavatte, C. Marot, L. Morin-Allory; 2D QSAR Consensus Prediction for High-Throughput Virtual Screening. An Application to COX-2 Inhibition Modeling and Screening of the NCI Database; J. Chem. Inf. Comput. Sci. 44, 276-285, **2004.**

4- M. Sree, Vadlamudi, M. Vithal, Kulkarni, 3D-QSAR of Protein Tyrosine PhosphatasE 1B Inhibitors by Genetic Function Approximation; Internet Electronic Journal of MolecularDesign ; 2, **2003.**

5- J. Polanski, A. Bak, R. Gieleciak, T. Magdziarz ; Self-organizing Neural Networks for Modeling Robust 3D and 4D QSAR: Application to Dihydrofolate Reductase Inhibitors ;Molecules 9, 1148–1159, **2004.**

6- A. Vedani, M. Dobler, 5D-QSAR: The Key for Simulating Induced Fit? ; *J. Med. Chem.* 45, 2139-2149, **2002.**

7- A. Vedani, M. Dobler and L. Lill, Combining protein modeling and 6DQSAR-Simulating the binding of structurally diverse ligands to the estrogen receptor. J.Med. Chem. 48, 3700–3703, **2005.**

8- http://fr.wikipedia.org/wiki/Autocorr%C3%A9lation

9- http://fr.wikipedia.org/wiki/Analyse_en_composantes_principales

10- R. D. Cramer, Cross validation, bootstrapping, and Partial Least Squares compared with multiple regression in conventional QSAR studies, Quantitative Structure-Activity Relationships, 7, 18-25; **1988**.

11- R. D. Cramer, Cross validation, bootstrapping, and Partial Least Squares compared with multiple regression in conventional QSAR studies, Quantitative Structure-Activity Relationships, 7, 18-25; **1988**.

12- http://fr.wikipedia.org/wiki/Analyse_discriminante

13- Beautrait, A., Leroux, V., Chavent, M., Ghemtio, L., Devignes, MD., Smail Tabbone, M., Cai, W., Shao, X., Moreau, G., Bladon, P., Yao, J., et Maigret, B., Multiple-step virtual screening using VSM-G: overview and validation of fast geometrical matching enrichment, J Mol Model, 14, 135-48, **2008**

14- http://www.mol-net.de/software/corina

15- http://openbabel.sourceforge.net

16- http://www.eyesopen.com/products/applications/omega.html

17- http://sms.cbi.cnptia.embrapa.br

18- http://www.ks.uiuc.edu/Research/vmd/

19- http://www.ustc.edu.cn

20- http://moitessier-group.mcgill.ca/research.htm#fitted

21- http://www.ccdc.cam.ac.uk/

CHAPITRE II

LE LANGAGE JAVA
ET
LA MAQUETTE DÉVELOPPÉE

I. INTRODUCTION :

Il existe plusieurs langages de programmation qui permettent, de nos jours, de faire du graphisme ou de créer des interfaces [(1)] graphiques pour différents utilisateurs. Pour pouvoir réaliser de jolies interfaces graphiques, faciles à l'emploi et efficaces, il faut axer son choix sur un langage puissant, qui offre une multitude de fonctionnalités.

Etant donné que la plateforme de screening VSM-G [(2)] est écrite en langage Java [(3)], nous avons été contraint de nous initier à ce dernier, car il correspond au langage d'écriture de la plateforme, et il fallait assimiler l'apparence des interfaces graphiques à implémenter, à celles que comporte VSM-G. Le choix s'est également porté sur ce langage pour la facilité à la maîtrise et à la manipulation qu'il présente.

II. LE LANGAGE JAVA

Le langage Java [(3)] est un langage de programmation [(4)] informatique orienté *objet* [(5)], créé par *James Gosling* [(6)] et *Patrick Naughton* [(7)], employés de *Sun Microsystems* [(8)], avec le soutien de *Bill Joy* [(9)] (cofondateur de *Sun Microsystems* en 1982), présenté officiellement le 23 Mai 1995 par *SunWorld*.

Le langage Java a la particularité principale que les logiciels écrits avec ce dernier sont très facilement portables [(10)] sur plusieurs systèmes d'exploitation [(11)] tels que : *UNIX* [(12)], *Microsoft Windows* [(13)], *Mac OS* [(14)] ou *Linux* [(15)], avec peu ou pas de modification. C'est la plateforme qui garantit la portabilité [(10)] des applications développées en Java.

Le langage reprend en majeure partie la syntaxe du langage C++ [(16)], fortement utilisé par les informaticiens. Néanmoins, ce dernier a été épuré des concepts difficiles du C++, tel que l'héritage multiple [(17)], remplacé par l'implémentation des interfaces. Les concepteurs ont privilégié *l'approche orientée objet*, de sorte qu'en Java tout est objet, à l'exception des types primitifs (nombres entiers, nombres à virgule flottante, etc…)

Java a donné naissance à un système d'exploitation (en *JavaOS*) [(18)] à un environnement de développement (*JDK*) [(19)], des machines virtuelles (*MEJVM, JRE*) [(20)] applicatives multiplateformes (*JVM*) [(21)] une bibliothèque Java (*J2ME*) [(22)], avec interface graphique (*awt / swing*) [(23)], des applications java (*logiciel9s, servlet, applet*). La portabilité du code java réside dans le support de plusieurs processus de compilation.

II.1 Caractéristiques du langage Java

A) Un langage orienté objet

Une programmation orientée *objet* [5] ou une programmation [4] *objet* est un style fondamental de programmation informatique, qui consiste en la définition et l'assemblage de briques logicielles (entités informatiques) appelées *objets*, un objet représente un concept, une idée ou toute entité du monde physique, comme une voiture, une personne ou encore une page d'un livre. En d'autres termes, un langage orienté objet se réfère à une méthode moderne de programmation et de conception de ce dernier. Ou encore, un langage orienté *objet* est un langage qui repose sur la représentation informatique des éléments du monde réel sur lesquels nous portons notre intérêt, sous forme d'*objet*. Sa principale caractéristique est de rassembler en une seule entité logique (*Classe*) [24] un ensemble de données (*Propriétés*) et les fonctions qui les traitent (*Méthodes*) [25]. Le code devient ainsi plus facilement réutilisable, car il est intrinsèquement modulaire.

D'autres mécanismes tels que l'*héritage* [26] permettent d'exploiter toutes les caractéristiques d'une *Classe* [24] précédemment écrite, dans ses propres programmes sans même avoir à en connaître le fonctionnement interne, on n'en voit que l'*interface* [1] ou les principales commandes.

Ce style fondamental de programmation vise à rendre les grands projets logiciels plus faciles à gérer, à améliorer la qualité des logiciels et à réduire le nombre d'échecs de projet.

B) Un langage portable, indépendant de la plateforme

Cette indépendance signifie que les codes écrits en langage Java fonctionnent d'une manière analogue, sur toutes les différentes architectures matérielles (ou tous les compilateurs). N'importe quel code, peut donc être très bien développé, dans un compilateur (qui traduit en langage machine) donné, et être (fait) tourné sur d'autres. Cette conséquence est due aux compilateurs Java, qui ne compilent le code qu'à moitié afin d'obtenir un *bytecode* [27] *(pseudo-code)* Java qui est un langage machine propre à la plateforme Java. Le code est ensuite interprété par la machine virtuelle Java (JVM) [20], qui interprète et exécute ce dernier. En outre, Java possède des bibliothèques standards, qui sont fournies pour que les programmeurs puissent accéder à la machine hôte, qui fournit les éléments de graphisme, de la programmation réseau…etc. L'indépendance de Java vis-à-vis de la plateforme peut très bien

être perçu dans les applications côté serveurs, tel que les services web. Effectivement, divers sites web (notamment ceux conçus pour les banques), qui sont écrits en langage Java et qui témoignent une grande fiabilité et une sécurité remarquable.

C) Un langage robuste et sûr

Java est un langage fortement typé [(28)], il élimine bien les problèmes d'incohérence de type à la compilation, la suppression de la manipulation des pointeurs [(29)] (une variable qui contient une adresse mémoire, ou encore, une variable qui contient l'adresse d'une autre variable d'un type donné, comme en langage C), permet de réduire considérablement les erreurs. Un glaneur de mémoire (un ramasse miettes [(30)], récupérateur de mémoire et en anglais garbage collector [(30)] qui est un sous système informatique de gestion automatique de la mémoire, responsable du recyclage de la mémoire préalablement allouée puis inutilisée), permet de décharger le programmateur d'une gestion fastidieuse de la mémoire.

Destiné pour des applications réseaux, la sécurité dans Java est un aspect primordial. Le fait de ne pas manipuler les pointeurs [(29)] et d'accéder à des zones mémoire sensibles, diminue fortement l'introduction de virus informatiques [(31)]. Ceci dit, Java ne supprime pas tous les problèmes de sécurité, mais les réduit fortement. Un ensemble de classes de sécurité a été conçu pour contrôler au mieux l'exécution du code Java tels que : Classe Loader et Security Manager [(32)].

D) Un langage dynamique et multithread

Java est un langage dynamique [(33)], qui s'adapte à l'évolution du système sur lequel il s'exécute. Les classes (une classe déclare des propriétés communes à un ensemble d'objets, ou encore elle déclare des attributs représentant l'état des objets et des méthodes (une méthode est une fonction faisant partie de l'interface d'un objet)), sont chargées au fur et à mesure des besoins, et à travers le réseau s'il le faut. Les mises à jour des applications peuvent se faire classe par classe, sans avoir à recompiler le tout en un exécutable final. De nos jours, les applications possèdent un haut degré de parallélisme, par exemple il faut pouvoir écouter une musique tout en visualisant une animation graphique, et c'est justement la notion de multithreading, que Java permet d'une manière simple.

II.2 Les Frameworks et API

On énumère trois plateformes Java : Micro Edition (ME), Standard Edition (SE) et Entreprise Edition (EE). En plus de cela, la société *SUN* offre un grand nombre de Framework et d'API.

Un Framework [(34)] est un espace de travail modulaire. C'est un ensemble de bibliothèques, d'outils et de conventions permettant le développement d'applications. Il fournit suffisamment de briques logicielles et impose suffisamment de rigueur pour pouvoir produire une application aboutie et facile à maintenir. Ces composants sont organisés pour être utilisés en interaction les uns avec les autres. Autrement dit, Un framework fournit un ensemble de fonctions facilitant la création de tout ou d'une partie d'un système logiciel, ainsi qu'un guide architectural en partitionnant le domaine visé en modules. Un framework est habituellement implémenté à l'aide d'un langage à objets, bien que cela ne soit pas strictement nécessaire : un framework objet fournit ainsi un guide architectural en partitionnant le domaine visé en classe, et en définissant les responsabilités de chacune ainsi que les collaborations entre classes.

Une API [(35)] (Application Programming Interface) ou une interface de programmation, est un ensemble de fonctions, procédures ou classes mises à disposition des programmes informatiques par une bibliothèque logicielle, un système d'exploitation ou un service. La connaissance des API est indispensable à la capacité de fonctionner entre les composants logiciels.

On distingue quatre grands frameworks :

- J2SE [(36)] : Ce framework est destiné aux applications pour poste de travail. Mais en théorie on parle désormais de Java SE.
- J2EE [(37)] : Ce framework est spécialisé dans les applications serveurs. Il contient pour ce faire un grand nombre d'API et d'extensions, mais en théorie on parle désormais de Java EE.
- J2ME [(38)] : Ce framework est spécialisé dans les applications mobiles.
- JavaCard [(39)] : Ce framework est spécialisé dans les applications liées aux cartes à puces et autres SmartCards.

La programmation peut se faire pour des exemples simples avec le compilateur *javac* [(40)], mais pour avoir plus de confort il est préférable d'utiliser un environnement de développement intégré [(41)] (en anglais : Integrated Development Environment), certains sont gratuits, on en cite : CodeWirrior [(42)], Idea [(43)], JBuilder [(44)], NetBeans [(45)], jDeveloper [(46)], Xcode [(47)], JCreator [(48)], Eclipse [(49)] et BlueJ [(50)], que nous avons utilisés pour développer les codes des interfaces graphiques créées lors de l'insertion de la chaine de programme de l'Autocorrélation dans VSM-G.

II.3 La programmation en langage Java

L'élaboration d'un programme en langage Java passe par les trois étapes suivantes :

1- *Ecrire* le programme Java et l'enregistrer
2- *Compiler* le programme, pour le traduire en langage machine (*bytecode*) compréhensible par la plateforme Java
3- *Exécuter* le programme

L'environnement de développement intégré *IDE* (*BlueJ* dans notre cas) peut réaliser ces trois étapes simultanément et rapidement, dans le même endroit. Si on veut, par exemple, créer un code qui affiche « Bonjour Monde » à l'écran, on écrit le code qui représente *la classe* BonjourMonde (en langage Java, classe = programme), permettant cela. Le code est écrit comme suit :

```
public class BonjourMonde {
        public static void main(String[] args) {
            System.out.println("Bonjour Monde");
                }
    }
```

On compile cette classe, et à l'exécution le message « Bonjour Monde » apparaît à l'écran de l'ordinateur.

Mais comment fonctionne la classe BonjourMonde ?

La classe BonjourMonde ne possède qu'une méthode main(), qui est le point d'entrée d'une *application* (programme) Java. On peut dire que main est une méthode parce qu'il y a des parenthèses après le mot main. Les méthodes peuvent *appeler* (*call*), c'est-à-dire utiliser

d'autres méthodes. Par exemple, notre méthode main() *appelle la méthode* println() pour afficher le texte "Bonjour Monde" à l'écran.

Toutes les méthodes commencent par une *ligne de déclaration* appelée *signature de la méthode* :

```
public static void main(String[]
```

Cette signature nous donne les informations suivantes :

- ✓ Qui a accès à la méthode – *public* : Le mot-clé public signifie que la méthode main() peut être utilisée par n'importe quelle autre classe Java ou par Java lui-même.
- ✓ Comment utiliser la méthode - *static* : Le mot-clé static signifie qu'il n'est pas nécessaire de créer une *instance* (une copie) de l'objet BonjourMonde en mémoire pour pouvoir utiliser cette méthode.
- ✓ La méthode *retourne*-t-elle des données ? Le mot-clé *void* signifie que la méthode main() ne retourne aucune donnée au programme appelant, (qui en l'occurrence est BlueJ). Mais si on prenait l'exemple d'une méthode effectuant des calculs, celle-ci pourrait retourner un résultat à son appelant.
- ✓ Le nom de la méthode est main.
- ✓ La liste des arguments – des données qui peuvent être fournies à la méthode – *String[] args* : Dans la méthode main(), String[] args signifie que la méthode peut recevoir un *tableau* (*array*) de chaînes de caractères (String) qui représentent du texte. Les valeurs qui sont passées à la méthode sont appelées *arguments*.

Un programme peut être constitué de plusieurs classes, mais l'une d'entre elles possède la méthode main(). Les classes Java ont en général plusieurs méthodes. Par exemple, une classe *Jeu* pourrait avoir les méthodes démarrerJeu(), arrêterJeu(), lireScore() et ainsi de suite.

Le corps de notre méthode main() ne contient qu'une ligne :

```
System.out.println("Bonjour Monde");
```

Chaque instruction ou appel de méthode doit se terminer par un point-virgule **;** . La méthode println() sait comment afficher des données sur la *console* système (fenêtre de

commandes). Les noms de méthodes Java sont toujours suivis par des parenthèses. S'il n'y a rien entre les parenthèses, cela signifie que la méthode ne prend pas d'arguments. L'expression System.out signifie que la variable *out* est définie à l'intérieur de la *classe System*, fournie avec Java. L'expression out.println() indique que la *variable* out représente un objet et que ce "quelque chose nommé out" possède une méthode nommée println(). Le point entre la classe et le nom de la méthode signifie que la méthode existe à l'intérieur de la classe.

Les programmes Java sont constitués de *classes* qui représentent des objets du monde réel. C'est la meilleure façon de le faire pour employer le style dit *orienté objet*. Cela signifie qu'un programmeur commence par décider quels objets doivent être inclus dans le programme, et quelles classes Java vont les représenter. Ce n'est qu'une fois cette étape menée à bien qu'il commence à écrire du code Java.

Les classes Java peuvent posséder des *méthodes* et des *attributs*. Les méthodes définissent les actions qu'une classe peut effectuer. Les attributs décrivent la classe.

Pour la création des interfaces graphiques en java, deux grandes *librairies* sont utilisées : *AWT* [51-52] (Abstract Windowing Toolkit) et *SWING* [53-54]. La librairie AWT fournit une API indépendante du système d'exploitation, pour mettre en œuvre des composants graphiques. La librairie SWING offre la possibilité de créer des interfaces graphiques identiques quel que soit le système d'exploitation.

AWT fut la première librairie à être apparue, c'est un ensemble de classes permettant la création de bouton (Button), des champs de texte (TextField), des labels (Label), etc. La librairie SWING (plus évoluée) est apparue peu après, elle comporte également l'ensemble des classes permettant la création d'une interface, dont les composants sont précédés de la lettre J, tel que : JButton [55], JTextField [56], JLabel [57], etc.

Pour la création de nos interfaces graphiques, nous avons utilisé les deux librairies, mais nous avons travaillé beaucoup plus avec SWING, car tout est plus rapide et plus simple d'utilisation. Par exemple, pour réaliser l'interface représentée ci-dessous, nous avons écrit le code suivant :


```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.*;
import javax.swing.UIManager;
```

Java est fourni avec de nombreuses classes utiles organisées en *paquetages* (*packages*) [(58)]. Certains paquetages contiennent des classes responsables du dessin, d'autres des classes permettant de travailler avec Internet et ainsi de suite.

Pour l'exemple, plusieurs déclarations d'*import* [(58)] *ont* été faites, pour éviter de re-écrire le nom complet d'une classe, à chaque fois qu'elle est utilisée (on effectue une seule fois la déclaration *import* avant la ligne de déclaration de la classe). Avec un *import*, on peut utiliser le nom court d'une classe, comme JFrame ou JButton.

S'il y a besoin de plusieurs classes d'un même paquetage, ce ne sera pas la peine d'écrire la liste dans la déclaration import. Il faut utiliser simplement le *caractère joker* (*wildcard*). Dans l'exemple suivant, l'étoile (astérisque) rend toutes les classes du paquetage javax.swing *visibles* dans le programme :

```
import javax.swing.*;
```

Pour déclarer les composants de la fenêtre (bouton et panneau), on procède comme suit :

```
JPanel panneauBoutons;
JButton boutonQSAR;
```

Les panneaux contiennent tous les composants d'une fenêtre (boutons, champs textuels, boites déroulantes, etc.)

Pour affecter des valeurs initiales aux variables membres d'une classe, Il faut déclarer un *constructeur*. Les constructeurs ne sont appelés qu'une fois, au cours de la construction d'un objet en mémoire. Ils doivent avoir le même nom que la classe elle-même et ils ne retournent pas de valeur ; il n'est même pas nécessaire d'utiliser le mot-clé *void* dans la signature d'un constructeur.

Pour déclarer un bouton et lui donner un nom, il faut taper l'instruction suivante :

```
boutonQsar = new JButton("QSAR");
```

Le panneau qui comprend ce bouton, est crée et déclaré comme suit :

```
panneauBoutons = new JPanel();
panneauBoutons.setLayout(new BoxLayout(panneauBoutons, BoxLayout.Y_AXIS));
panneauBouton.add(boutonQsar);
```

Une fois le bouton créé, il faut lui attribuer une place dans l'interface. Ceci se fait au moyen de gestionnaires de placement ou de disposition *Layout Manager* [(59)]. Ces *layout* permettent d'arranger les composants sur l'interface sans avoir à affecter des positions précises aux contrôles graphiques. Les gestionnaires de disposition assurent une bonne visualisation à l'écran, quelles que soient les dimensions de la fenêtre. Swing propose les gestionnaires de disposition suivants :

- ✓ GridLayout (présentation des composants en ligne et en colonne dans une grille)
- ✓ BoxLayout (présentation des composants soit horizontalement (en ligne) suivant l'axe X, ou verticalement (en colone) suivant l'axe Y)

- ✓ FlowLayout (présentation en file : ligne par ligne)
- ✓ BorderLayout (présentation avec bordures : la fenêtre est divisée en cinq zones South (sud), North (nord), East (est), West (ouest) et Center (centre))
- ✓ CardLayout (présentation en pile qui ressemble à un classeur d'onglets)
- ✓ GridBagLayout (présentation en grille composite permettant d'avoir des cellules de tailles différentes)

L'ajout du panneau contenant le bouton à la fenêtre (contenant tous les composants), se fait par cette instruction :

```
contenuFenetre.add("Center",panneauBoutons);
```

Pour placer ce panneau dans la zone « centre » de la fenêtre, il faut utiliser le gestionnaire de disposition *BorderLayout*,

Les actions susceptibles de parvenir lors d'un clic sur un bouton, sont générées par les gestionnaires d'événement (*ActionListener)* [60] ou écouteurs.

```
boutonQsar.addActionListener (new ListenerQsar( )) ;
-----------
    private class ListenerQsar implements ActionListener {
        public void actionPerformed(ActionEvent evt) {
                    // Code de l'action désirée
                        }
                }
```

Le bouton doit être à l'écoute au moyen d'une classe Java appelée *Listener*, pour produire l'événement ou l'action demandée. Quand un événement d'action (action event) se produit, Java appelle la méthode *actionPerformed (ActionEvent evt)* [60] de la classe Listener, et lui fournit des informations sur l'événement dans l'argument *evt*. La classe *ListenerQsar* et la méthode *actionPerformed*, doivent s'écrire en dehors du constructeur.

Pour finir le code il faut définir la fenêtre *frame* [61], en lui donnant un titre, des dimensions et en la rendant visible :

```
JFrame frame = new JFrame("Programs");
frame.setSize (80, 100);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setVisible(true);
```

Et bien évidemment, il faut ajouter en plus la fonction *main* pour ensuite exécuter le programme Java, à fin de visualiser l'interface graphique.

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.*;

public class Programs {
 //Déclarer toutes les composantes de la fenêtre.
JPanel contenuFenetre;
JPanel panneauBoutons;
JButton boutonMSSH;
JButton boutonGOLD;
JButton boutonQSAR;

//On crée les composants en mémoire
public Programs() {
contenuFenetre = new JPanel();

//Création des boutons et ajout des ActionListeners.
boutonMSSH= new JButton("MSSH");
boutonGOLG = new JButton("GOLD");
boutonQSAR = new JButton("QSAR");
boutonQSAR.addActionListener(new EcouteurQSAR());
```

```
 //Présentation : création du panneau et du quadrillage qui contient
 //tous les boutons.
panneauBoutons = new JPanel();
GridLayout GridLay = new GridLayout (3, 1, 10, 10);
panneauBoutons.setLayout(GridLay);

//Ajout des controles au panneauBoutons.
panneauBoutons.add(boutonMSSH);
panneauBoutons.add(boutonGOLD);
panneauBoutons.add(boutonQSAR);

//Ajout du panneauBoutons à la zone centrale de la fenêtre.
contenuFenetre.add("Center", panneauBoutons);
contenuFenetre.add(panneauBoutons);

//Présentation du contenuFenêtre
GridLayout Grid = new GridLayout (1, 1);
contenuFenetre.setLayout(new BoxLayout(Grid);

//Créer et afficher la fenêtre.
JFrame frame = new JFrame("Programs");
frame.setSize (80, 100);
frame.setContentPane(contenuFenetre);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.pack();
frame.setVisible(true);

} // fin du constructeur

//Classe interne : écouteur QSAR qui permet de lancer la fenêtre QSAR
private class EcouteurQSAR implements ActionListener {
          public void actionPerformed(ActionEvent evt) {
                // écrire le code de l'action
             }
       }

public static void main(String[] args) {
   Programs prog = new Programs ();
  }
```

III. LA MAQUETTE COMPORTANT LES OUTILS QSAR

La maquette que nous avons développé au cours des travaux entrepris dans le cadre de ce magister, se résume en un ensemble d'interfaces graphiques conçues pour un usage facile et agréable des outils statistiques QSAR, et ce à travers VSM-G.

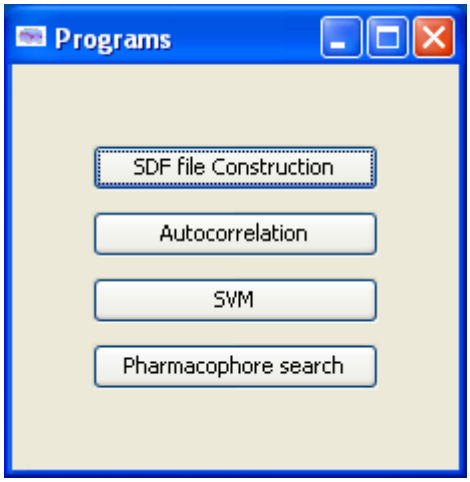
Une interface graphique est un environnement, qui permet le couplage entre classes, objets et modules. L'interface qui est présentée à l'utilisateur est appelée *interface utilisateur.* Elle donne accès aux fonctions du programme par le biais d'un clavier, d'une souris ou d'un écran tactile tout en les représentant d'une manière graphique (on parle alors du couplage : Homme- Machine). En d'autres termes, une interface graphique est une zone réservée à l'utilisateur, sur laquelle il peut agir avec différents périphériques d'entrée comme le clavier ou la souris, de telle sorte qu'il ait en retour sur un écran, une image, une animation ou même des vidéos.

Le but de nos travaux est celui d'élaborer des interfaces graphiques qui seront faciles à l'emploi telle que l'interaction entre l'homme et l'ordinateur qui se fait au moyen d'un clavier, et sans visualisation élaborée dans un terminal ou dans une fenêtre de terminal en mode texte, comme par exemple dans le cas du DOS [(62)] (sous lequel les exécutables de la suite de programmes de l'Autocorrelation apparaissent), qui signifie *Disk Operatig System*, ou tout simplement *système d'exploitation* développé par *Microsoft*. On parle alors de GUI (Graphical *User Interface*), qui signifie *Interface Utilisateur Graphique*. Dans le cadre de ces travaux, entrepris en collaboration avec l'équipe Orpailleur du LORIA (Laboratoire Lorrain de la Recherche en Informatique et ses Applications) de Nancy, et sous la direction de monsieur Bernard Maigret directeur de recherche au CNRS (Centre National de Recherche Scientifique), nous devions respecter trois critères pour l'élaboration des interfaces graphiques, sous lesquelles apparaîtraient les exécutables de l'autocorrelation, ces dernières devaient être faciles à utiliser, conviviales et agréables.

La réalisation de la partie statique (boutons [(63)], dialogues [(64)], champs textes [(65)] et autres), et de la partie dynamique (réaction aux événements), des interfaces graphiques que comporte la maquette qui devait comporter les techniques QSAR, ont été écrites en Java. Cette réalisation s'est faite en respectant le cheminement suivant :

- ✓ Créer une interface servant à la préparation du fichier SDF [66-67], si ce dernier n'est pas préparé.
- ✓ Implémenter les quatre exécutables de la chaine de programmes de l'Autocorrélation à savoir : Autocorrelation2 [68], Acplpk2 [68], Regmul [69] et Anadis [70].
- ✓ Faire appel à OriginPro 7.5 [71], un outil pour l'illustration graphique des résultats obtenus au cours des applications effectuées
- ✓ Ecrire une aide et un guide à l'utilisateur

L'ensemble de ces interfaces graphiques et leur fonctionnement sont présentés ci-dessous :

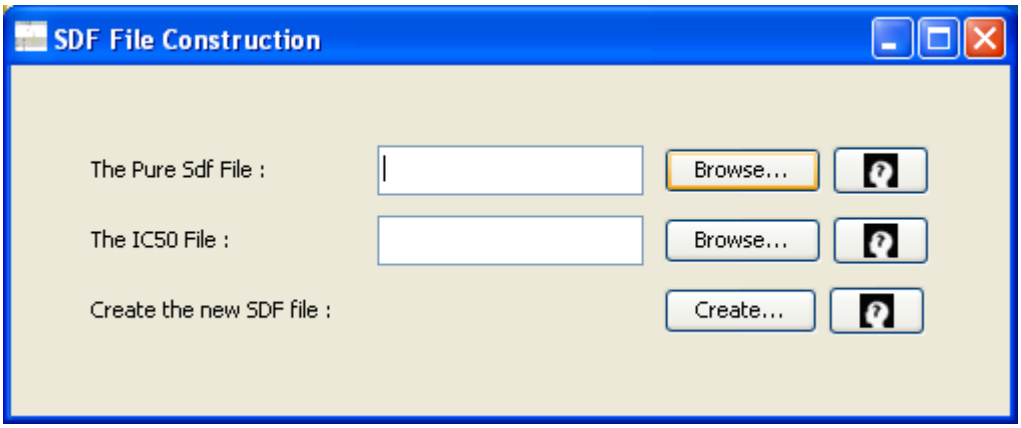


Cette interface intitulée « Programs », englobe quatre boutons. Le troisième et le quatrième bouton sont, pour le moment, inactifs. Le troisième bouton sert à faire du Support Vector Machine (SVM) [72], ou encore dit : séparateurs à vaste marge, qui sont un ensemble de techniques, d'apprentissage supervisé, utilisées pour la résolution des problèmes de discrimination et de régression. Le quatrième bouton « Pharmacophore Search » [73], représente une technique pour le screening virtuel, basée sur la recherche de pharmacophores dans une base de donnée.

Pour que la chaîne de programmes de l'Autocorrelation puisse fonctionner, il faut avoir le fichier d'entrée. Ce fichier est de type sdf (Structure data File), il contient les structures des molécules et leurs valeurs d'activités, la partie structure étant dans le format V2000 [74]. Ces structures peuvent être dessinées avec n'importe quel logiciel conçu pour cela, dans notre cas ISIS DRAW [75] : ce programme permet de créer un fichier pour chaque molécule dessinée (sans valeur d'activité), un traitement informatique est nécessaire pour les mettre bout à bout et

obtenir un fichier sdf « pur » qui contient toutes les structures. Si les structures et leurs activités à étudier sont déjà dans une base de type ISIS-BASE [76], le plus simple est de les extraire sous la forme d'un fichier *sdf*.

Le premier bouton « SDF File Construction », est celui qui ouvre l'interface qui permet de créer le fichier des structures moléculaires constituant la base de données, au format SDF, si ce dernier n'existe pas. Elle se présente comme suit:



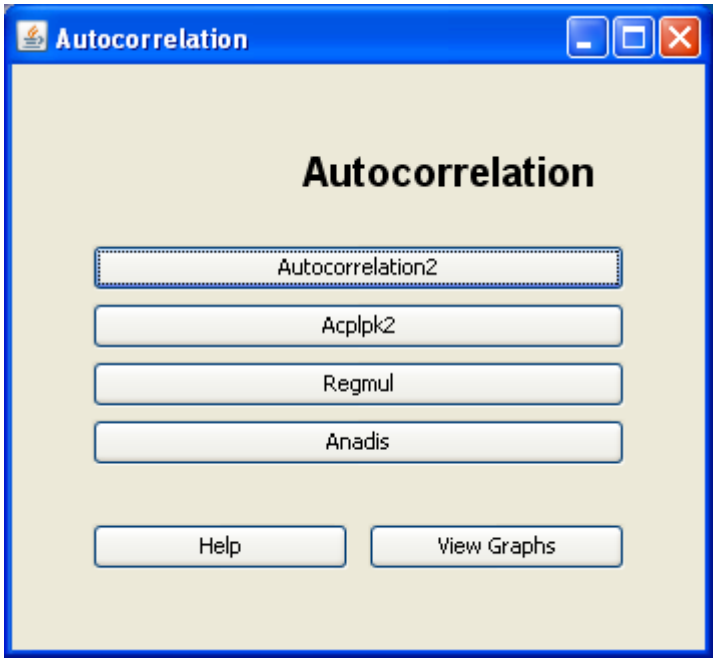
Elle comporte deux champs textes, trois boutons et trois boites de dialogue. Le premier champ de texte « The Pure Sdf File » permet en cliquant sur le bouton « Browse… », de charger et d'afficher le chemin du fichier au format sdf dit « pur », c'est le fichier obtenu après concaténation des structures moléculaires individuellement prises dans des fichiers portant l'extension « *.mol* »[77,78]. Le deuxième champ de texte « The IC50 File » permet en cliquant sur le deuxième bouton « Browse… », de charger et d'afficher le chemin du fichier qui comporte les valeurs des activités biologiques des structures moléculaires qui constituent la chimiothèque, prises dans l'ordre dans lequel elles apparaissent dans cette dernière. Ces deux fichiers doivent impérativement être copiés et collés dans des formats « texte » (le Notepad), autrement le programme ne fonctionnera pas.

En cliquant sur le bouton « Create », on crée un fichier au format sdf, complet et utilisable pour les calculs ultérieurs (Autocorrélation, SVM ou autres). Ce dernier comportera les informations relatives à la molécule en question, suivie de sa valeur d'activité. Le programme conçu, fait ceci pour toutes les molécules de la base de données, et le fichier résultant se trouve à l'endroit où les deux précédents fichiers étaient déjà enregistrés.

Remarque :

Cette interface génère un fichier au format sdf nommé « New_sdf_file », à chaque utilisation, il est donc nécessaire de renommer ce fichier à chaque fois pour des utilisations ultérieures.

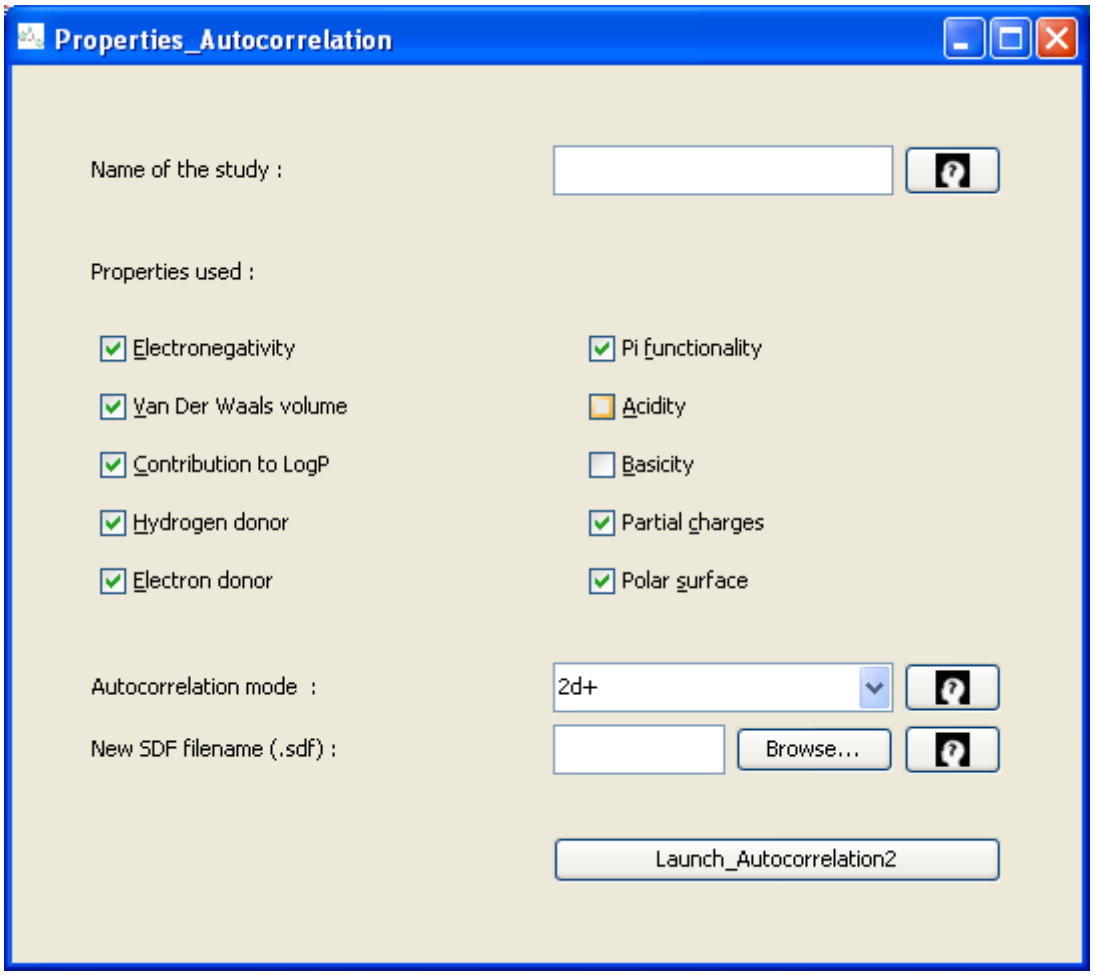
Le deuxième bouton de l'interface « Programs » intitulé « Autocorrelation » ouvre la fenêtre « Autocorrelation » qui assemble les quatre techniques QSAR :



Cette fenêtre englobe six boutons, les quatre premiers sont ceux correspondant aux exécutables de l'Autocorrelation, et les deux derniers représentent successivement, une aide et un outil d'illustration graphique.

Chacun des quatre premiers boutons de cette interface ouvre la fenêtre qui lui correspond. Chaque interface retraduit chacun des programmes de la chaine d'autocorrélation, qui, il faut le noter, fonctionnent en interactivité (c'est-à-dire qu'ils posent une suite de questions à travers une fenêtre DOS). L'idée était de créer des interfaces graphiques plutôt conviviales et simples d'utilisation, semblables à celles que comporte VSM-G. Ces interfaces doivent englober toutes les questions de chacun des programmes originaux et quelques unes des réponses (celles qui restent inchangées bien évidemment), et ce dans le but de faciliter à l'utilisateur, leur exploitation et leur manipulation. Il est à noter que l'ordre dans lequel ces questions ont été posées, a été respecté.

Ainsi, si on souhaite générer les vecteurs d'Autocorrelation, on clique sur le premier bouton « Autocorrelation2 » conçu pour le programme Autocorrelation2, et on aura l'interface « Properties_Autocorrelation » :



La fenêtre « Properties_Autocorrelation » prend cet aspect. Elle a été épurée de ce qui a été jugé non nécessaire à la visualisation, et ce dans le but d'alléger l'interface et de la rendre plus accessible et plus facile à l'utilisation.

« Properties_Autocorrelation » comporte des champs de texte, des boutons à cocher, une boite déroulante et des boites de dialogues. Pour effectuer une étude Atocorrelation2 afin de générer les vecteurs d'Autocorrélation, on remplit les champs de cette fenêtre tel que suit :

1- D'abord, on donne un nom pour l'étude qu'on veut accomplir, et qu'on incère dans le champ *Name of the study*. Ce nom ne doit comporter aucune extension. Ce nom sera le nom du fichier généré par Autocorrélation2, complété par l'extension .vec

Ecemple : si le nom de l'étude est *monEtude*, le fichier généré par le programme Autocorrelation2 sera *monEtude.vec*

2- Nous trouvons ensuite les boutons à cocher qui représentent les dix propriétés, *properties used*, calculables par le programme et qui sont : L'électronégativité, le volume de van der Waals, la contribution de chaque atome au logP calculé de la molécule, la capacité pour un atome de donner ou de ne pas donner un hydrogène pour la liaison hydrogène, le fait qu'un atome soit accepteur ou non accepteur d'hydrogène, la fonctionnalité Π (degré de saturation de l'atome), l'acidité, la basicité, les charges partielles et l'aire de la surface polaire (PSA). Dès l'ouverture de l'interface, on remarque que certaines propriétés sont préalablement cochées. Ce choix précis représente le meilleur calcul, mais il est tout à fait possible de cocher ou de décocher n'importe quelle propriété, cela dépend évidement de la nature de l'étude à effectuer et des besoins de l'utilisateur.

3- Ensuite il faut choisir le mode de l'autocorrélation en cliquant sur la flèche de la boite déroulante *Autocorrelation mode*. Si on ouvre cette boite déroulante on trouve les modes 2D, 2D+ et 3D. 2D et 2D+ utilisent la structure 2D et peuvent par conséquent être utilisés aussi avec des structures 3D (puisque la structure 2D y est contenue).

Le mode 2D est sensible à la taille des molécules, les molécules petites donnant moins de composantes que les plus grandes. Il convient de retrancher les vecteurs d'autocorrélation les plus grands avant de faire une analyse statistique, c'est pourquoi il est recommandé d'utiliser le mode 2D+.

2D+ est conseillé car il convient pour des structures de tailles très variées (aucun retranchement n'est nécessaire).

3D ne doit être utilisé que si on veut vraiment utiliser les structures tridimensionnelles.

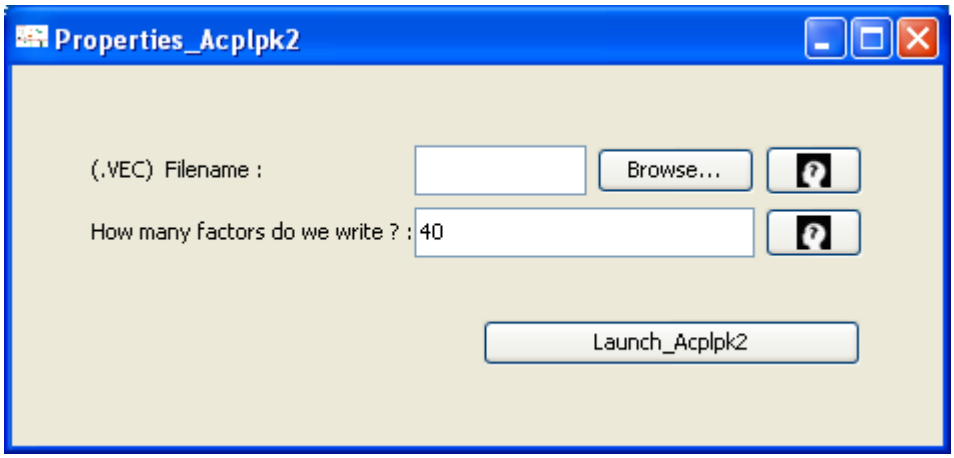
On remarque sur l'interface que le mode 2D+ est déjà coché, et que l'utilisateur a la possibilité de le changer.

4- Ensuite il faut charger le fichier des structures moléculaires au format sdf, soit celui déjà préparé moyennant l'interface « The Sdf File Construction », ou le

fichier que l'utilisateur aura préparé préalablement, sans cette dernière. En cliquant sur le bouton « Browse », une fenêtre de sélection s'ouvre. On remarque qu'un filtre a été mis à disposition pour ne pouvoir sélectionner que les fichiers qui auront pour extension le *.sdf*. Ceci est fait dans le but de faciliter à l'utilisateur la sélection du fichier voulu.

En cliquant sur le bouton « Launch_Autocorrelation2 », la fenêtre « Properties_Autocorrelation2 » disparaît, et on génère les vecteurs d'Autocorrélation dans un document *Wordpad*, comportant l'extension *".vec"*. On retrouve ce dernier au même endroit où le fichier des structures moléculaires (*.sdf*) est enregistré.

Pour effectuer une étude d'analyse en composantes principales, on clique sur le deuxième bouton de l'interface « Autocorrelation », pour ouvrir la fenêtre « Properties_Acplpk2 », qui prend cette forme :



Elle ne rapporte que les questions jugées nécessaires et suffisantes pour le fonctionnement du programme. C'est une fenêtre dégagée et très simple à utiliser. Elle comporte uniquement deux champs d'affichage, deux boutons et deux boites de dialogue. Pour réaliser une ACP sur les vecteurs d'autocorrélation il suffit de :

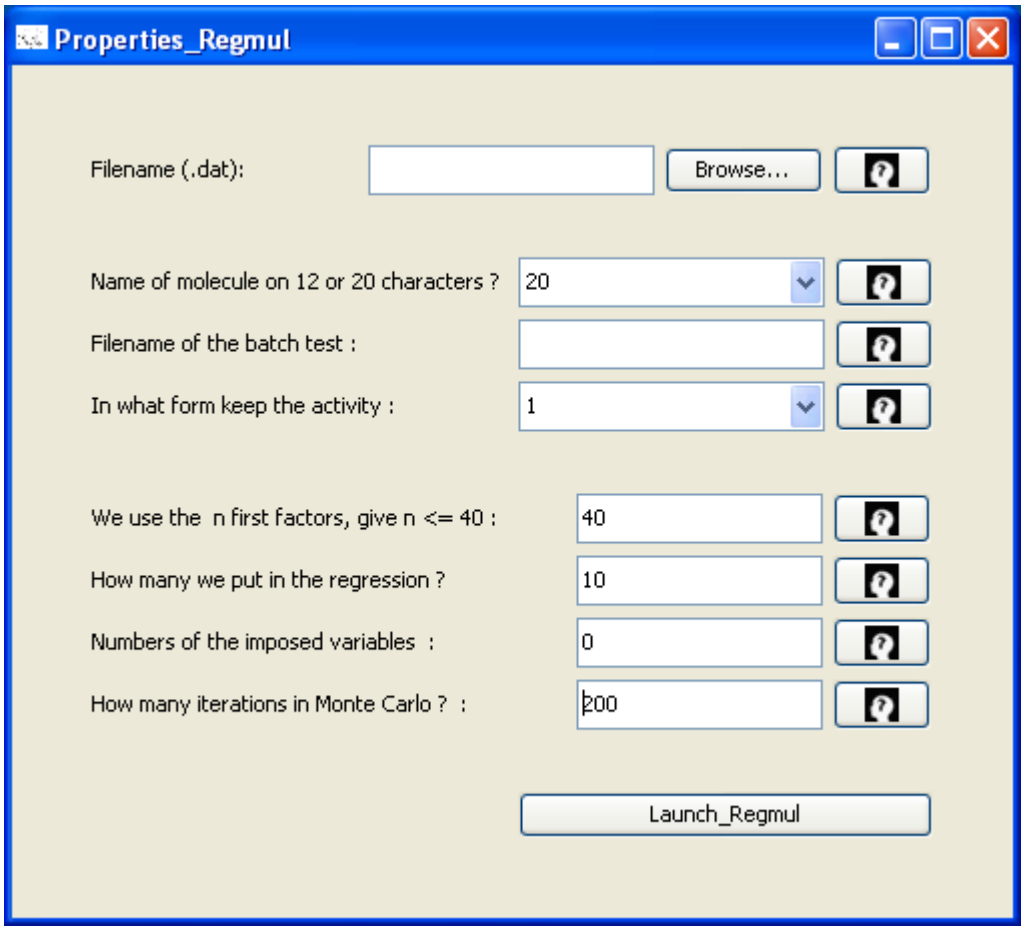
1- Charger le fichier comportant l'extension *.vec*. En cliquant sur le bouton « Browse », une boite de sélection de fichier s'ouvre comportant un filtre, facilitant ainsi l'accès au fichier en question.

2- Ensuite il faut préciser le nombre de coordonnées factorielles issues du fichier au format *.vec*, qu'il faut insérer. Le programme peut accepter jusqu'aux 40 premiers facteurs, alors cette valeur est mise dans ce champ à titre d'exemple, mais elle peut être prise ou modifiée, bien entendu.

En cliquant sur le bouton « Launch_Acplpk2 », la fenêtre disparaît et on retrouve les deux fichiers de résultats comportant le même nom *monEtude* de l'étude suivi des extensions *.acp* et *.dat*. Le fichier *monEtude.acp* contient le détail du travail et des résultats produits par le programme Acplpk2 (composantes éliminées s'il y en a, pourcentages d'information sur chaque dimension, valeurs et vecteurs propres,…). Le fichier *monEtude.dat* contient autant de lignes que de molécules sur lesquelles sont écrites leurs coordonnées factorielles et l'activité.

De même, pour réaliser une étude de régression multilinéaire, il suffit de cliquer sur le troisième bouton de l'interface « Autocorrelation », qui ouvre la fenêtre « Properties_Regmul », qui reproduit la contenance du programme Regmul tel qu'il était présenté dans son aspect initial. Ce programme également fonctionne en interactivité, mais nous l'avons reformulé en une fenêtre claire et agréable, afin de retracer les questions du programme, dont certaines ont été codées en dure. Ceci est dû dans le but d'aérer l'interface, et de la rendre plus accessible.

L'interface « Properties_Regmul » est une combinaison de plusieurs champs d'affichage, boites déroulantes, et de mini boutons d'aide. Elle se présente sous cette forme :



Le programme travaille sur un fichier de type *monetude.dat*, mais généralement si on n'est pas trop juste en nombre de molécules il est bon de retirer du fichier *monetude.dat* un certain nombre de molécules (de lignes), choisies au hasard, qui constitueront un nouveau fichier de même type, on aura donc deux fichiers *monetude1.dat* et *monetude2.dat* ; par exemple *monetude1.dat* contiendra 200 molécules et *monetude2.dat* en contiendra 50. Le premier fichier *monetude1.dat* est appelé parfois le fichier d'*apprentissage* parce que c'est lui qui sert au programme pour établir la régression multilinéaire. Ensuite le programme utilise cet apprentissage pour prédire la classe d'activité des molécules du fichier *monetude2.dat* : de la comparaison des activités prédites avec les activités réelles on peut conclure sur la qualité de la régression . Le fichier *monetude2.dat* est appelé parfois fichier *test*, mais il peut être aussi un fichier de structures nouvelles pour lesquelles on veut faire des prédictions. Il est à noter qui si on veut faire cela, il faut que les structures nouvelles soient placées dans le même repère factoriel que les molécules d'apprentissage.

Dans le but de rechercher les régressions multilinéaires entre l'activité et les facteurs principaux ainsi que leurs termes quadratiques, issus de l'Acp sur des vecteurs

d'autocorrélation, à travers l'interface « Properties_Regmul », il suffit de respecter cette façon de procéder :

1- Il faut charger le fichier portant l'extension *.dat*, issu de l'étude de l'analyse en composantes principales sur les vecteurs d'autocorrélation, contenant les X coordonnées factorielles (ou les facteurs principaux). Pour faciliter la recherche de ce fichier, un filtre a été confectionné pour ces besoins.

2- Ensuite il faut préciser si le nom de la molécule est écrit sous 12 ou 20 caractères. Les chiffres 12 et 20 représentent la longueur en nombre de caractères du champ, dans lequel on a mis le nom de la molécule. Ce champ permet d'entrer jusqu'à 12 ou 20 caractères. Il faut faire en sorte que le champ soit toujours 20 quand le fichier au format *sdf* est fabriqué (il faut écrire en format 20F10 en langage Fortran, langage d'écriture de ces exécutables).

3- Ensuite il faut introduire le nom du fichier du lot d'essai dans le champ texte *Filename of the batch test*. Le lot d'essai permet de mesurer la qualité du modèle ou de le valider. Ce fichier contient des molécules prises au hasard du fichier au format *.dat (monEtude.dat).* Si le fichier du lot d'essai éxiste, il faut mettre son nom, suivi de l'extension *.dat* bien évidemment. S'il n'existe pas, dans ce cas là il faut mettre l'expression *vide.dat*

4- Il faut ensuite, préciser sous quelle forme est présentée l'activité dans la boite déroulante *In what forme keep the activity*. En fait cette boite contient trois réponses pour permettre les changements sur l'unité de l'activité.

Réponse « 1 » : Si on veut transformer l'activité en log(activité)
Réponse « 2 » : Si on veut transformer l'activité en 10*tanh(activité/activité0)
Réponse « 3 » : Si on veut transformer l'activité en arctang(activité/activité0)
Pour cette question, la réponse est choisie d'office, généralement on travaille avec des log(activité).

5- Après, il faut donner le nombre *n* des premiers facteurs principaux, parmi lesquels seront choisis les *p* facteurs qui entrent dans la régression.

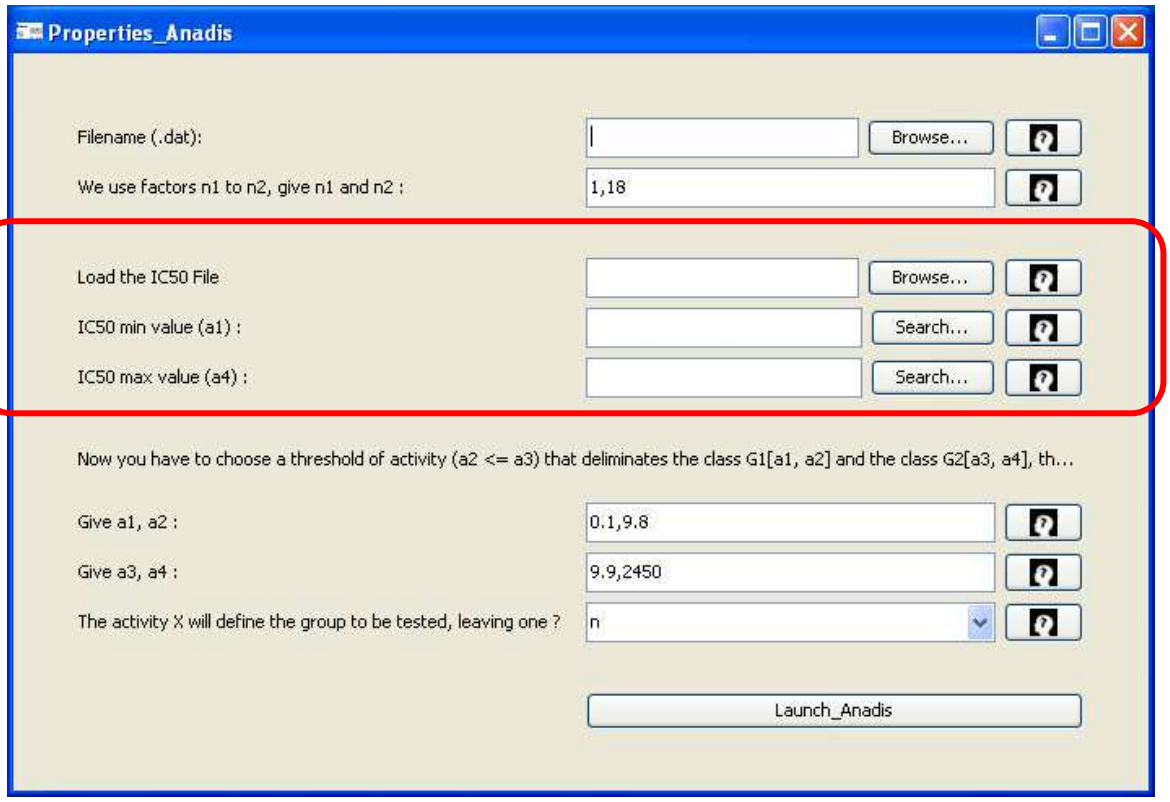
6- Il faut aussi préciser le nombre des *p* facteurs effectivement présents dans la régression, ce nombre doit être inferieur à *n*. Ce nombre *p* doit de préférence être inférieur au nombre de molécules divisé par 5.

7- Il faut donner le nombre de variables imposées. C'est-à-dire qu'on peut imposer à des facteurs d'être présents dans la régression (dans la forme linéaire) recherchée. Ces nombres doivent être >=2 et inférieurs à *n*. Si toute fois il n'y aurait pas de variables imposées, on insère le chiffre 0.

8- En fin, la dernière étape consiste en la précision du nombre d'itérations à utiliser par la méthode de Monté Carlo [(79)]. La recherche des meilleurs jeux de *p* facteurs choisis parmi les *n*, se fait à l'aide d'un algorithme de Monte-Carlo. Un jeu initial est essayé, il conduit à une régression, dont la qualité est caractérisée par le coefficient de corrélation, entre activités calculées, activités expérimentales et le facteur statistique F. Ce jeu est modifié par la sortie d'un facteur et l'entrée d'un autre ; la nouvelle régression est déterminée et comparée à la précédente. Selon qu'il y a amélioration ou non, une logique de Monte-Carlo est appliquée. On retient la meilleure régression obtenue au bout d'un certain nombre d'itérations. On signale que le programme peut faire jusqu'à 10 000 itérations.

En cliquant sur le bouton « Launch_Regmul », la fenêtre disparaît et on retrouve les deux fichiers de résultats portant des noms différents, mais suivis de l'extension *.dat*. Le premier fichier est noté *xxvide.dat* dont xx est un nombre hexadécimal, et le deuxième fichier prend le nom inséré du fichier du lot d'essai (en l'occurrence *vide.dat* si ce fichier n'existe pas). Ces fichiers se retrouvent également dans le même endroit où le fichier *.dat*, que le programme Regmul prend en entrée, est enregistré.

Pour procéder à une étude d'analyse en composantes principales, on clique sur le quatrième bouton de l'interface « Autocorrelation », pour ouvrir la fenêtre « Properties_Anadis ». Cette interface englobe différents boutons, champs d'affichage, boites déroulantes et boites de dialogues. Elle s'illustre comme suit :



Pour entamer une étude, (qui a pour but la séparation des molécules étudiées en deux classes. La première contenant les molécules actives et la deuxième les molécules peu ou pas actives en fonction d'un seuil qu'on choisit à la suite de divers essais), depuis l'interface « Properties_Anadis », il suffit d'appliquer les instructions suivantes :

1- Il faut charger le fichier portant l'extension *.dat*, issu de l'étude de l'analyse en composantes principales sur les vecteurs d'autocorrelation, contenant les X facteurs principaux. Pour faciliter la recherche et la sélection de ce fichier, un filtre a été confectionné pour ces besoins. Ce dernier ne laisse apparaître que les fichiers portant l'extension voulue.

2- En suite, il faut choisir un intervalle, qui délimitera les facteurs principaux numéro n1 à n2

3- La portion entourée en rouge n'est pas encore fonctionnelle, elle est en cours de traitement. Cette partie de l'interface sert au calcul de la valeur maximale et la valeur minimale du fichier qui comporte les valeurs des activités. Ce code est écrit dans le but de faciliter l'utilisation d'Anadis à travers sa nouvelle interface, et pour que l'utilisateur n'ait à fixer que le seuil d'activité. Il faut noter que le fait que cette portion de l'interface ne soit pas encore

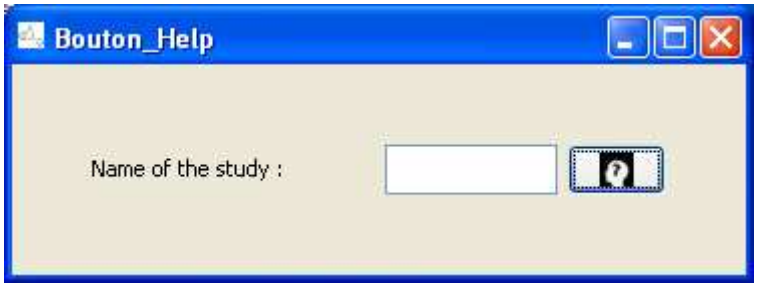
opérationnelle, cela ne compromet pas le bon fonctionnement de notre maquette.

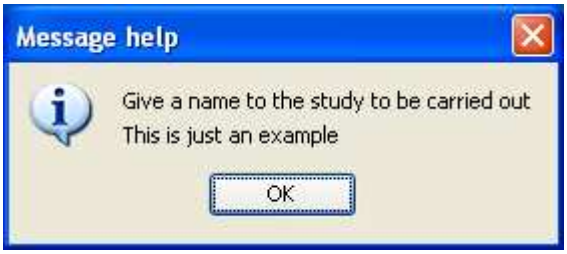
4- Après avoir choisi un seuil d'activité, il faut essayer de classer les molécules dans deux classes distinctes par rapport à ce dernier. La première classe comporte un intervalle comprenant les valeurs des activités de molécules actives. La deuxième classe comporte également un intervalle comportant les valeurs des activités des molécules peu actives ou inactives. Un exemple illustratif est inséré dans chaque champ, afin d'expliquer la façon d'écrire ces valeurs. Le programme peut ainsi choisir les molécules du fichier en extension *.dat* qui répondent aux conditions requises

5- En fin, on a le champ qui comprend en une boite déroulante, la faculté de laisser ou non un groupe à tester. Après qu'on ait choisi des intervalles pour les classes des molécules, s'il y aura certaines valeurs activités qui n'appartiennent à aucune des deux classes, alors elles constitueront le groupe à tester. Un choix de réponses est donc possible à faire. Généralement, on ne laisse pas un groupe à tester, on veut que le programme reclasse les molécules actives et inactives.

En cliquant sur le bouton « Launch_Anadis », la fenêtre disparaît et on retrouve le fichier des résultats portant le même non *monEtude* de l'étude suivi de l'extension *.mda*, qui se trouve au même endroit où le fichier d'entrée portant l'extension *.dat*, est enregistré. Le fichier *monEtude.mda* comporte l'ancien et le nouveau classement des molécules dans les classes d'activité (avant mda : c'est le groupe 1 ou 2 définis par l'activité réelle et après mda : c'est le groupe auquel la molécule se trouve rattachée par le calcul de l'analyse discriminante).

Dans toutes les précédentes interfaces, nous trouvons, à côté de chaque composant de la fenêtre, de minis boutons d'aide (illustrés en une tête marquée d'un point d'interrogation).





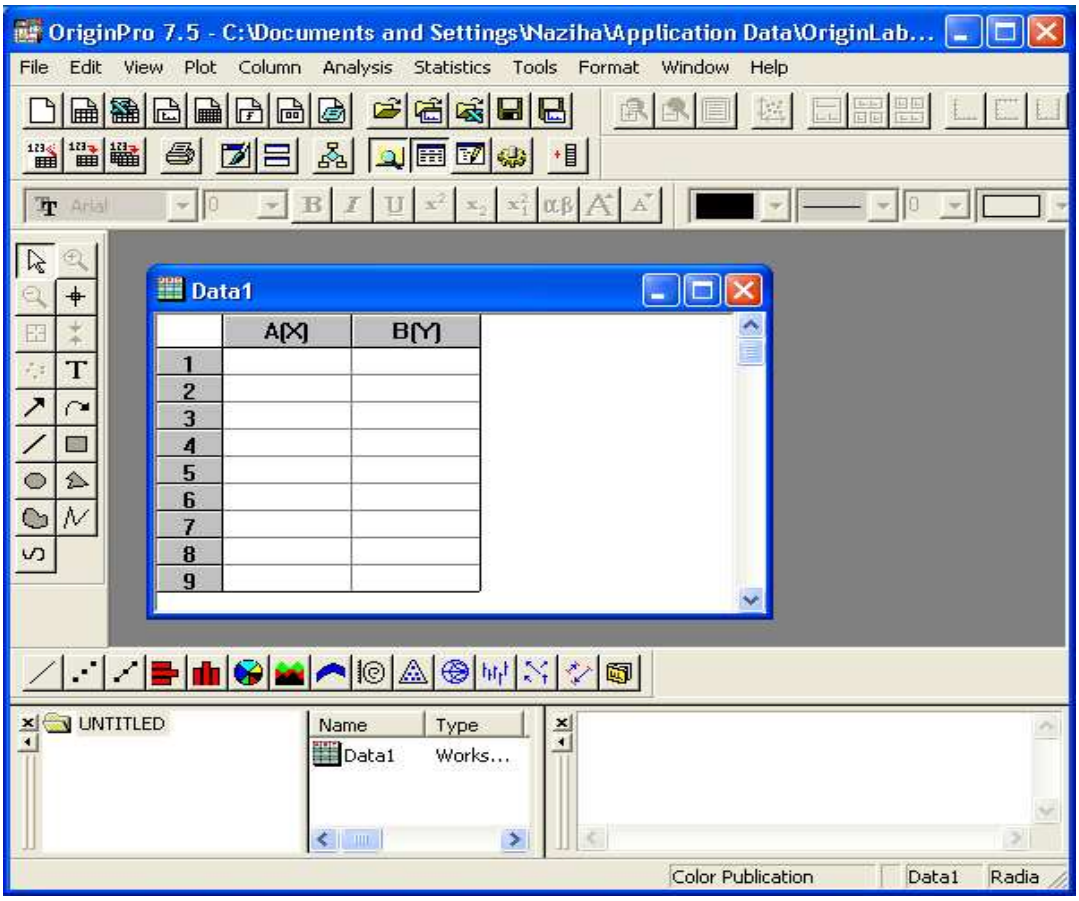
À chaque clic, une boite de dialogue (ou d'information) s'ouvre et nous trouvons des explications concernant la question à traiter afin de l'éclaircir, dans le cas où il y a confusion des instructions d'utilisation des programmes cités, à travers ces interfaces.

Comme nous l'avons déjà annoncé, l'interface « Autocorrelation », contient un outil permettant la visualisation graphique des résultats obtenus, au cours des calculs statistiques issus des techniques précédemment citées. En cliquant sur le cinquième bouton de cette interface, portant le nom « View Graphs », le programme OriginPro 7.5 se lance. Ce programme est un outil très sophistiqué et pointu, servant à faire n'importe quel tracé de graphe, il est à la fois simple, complet et facile à l'exploitation. Pour pouvoir utiliser OriginPro 7.5 directement à partir de l'interface de VSM-G, il faut l'avoir préalablement installé sur l'ordinateur.

Si on veut, à titre d'exemple, tracer le nuage de points issu de l'étude de l'analyse en composantes principales sur les vecteurs d'autocorrelation, moyennant OriginPro 7.5, on procède de la façon suivante :

1- Il faut cliquer sur le bouton « View Graphs »

   Afin de pouvoir ouvrir l'interface centrale du programme OriginPro 7.5, qui apparaît sous cet aspect :



2- Ensuite il faut charger le fichier au format *.dat* issu de l'étude de l'ACP, à travers l'interface d'Origin. Ceci se fait en cliquant sur : « File » → « Import… » → « Simple Single ASCII » → Ici, on cherche le fichier *.dat* voulu, et on le sélectionne pour le charger → Le contenu du fichier *.dat* se colle automatiquement sur l'interface d' « OriginePro 7.5 », dans son tableau principal

3- Après, il faut sélectionner deux colonnes distinctes sur tout le tableau, ensuite cliquer sur « Plot » → et puis sur « Scatter », ou choisir le style avec lequel on souhaiterait tracer le graphe

4- Finalement, le nuage de points voulu, se dessine et est facilement interprétable.

OriginPro 7.5 présente réellement une façon très conviviale de l'utiliser et d'en faire profit, il ne prend que peu de temps pour s'habituer à son interface, et bien l'exploiter pour des besoins aussi bien divers que différents.

❖ ***Remarque :*** Visualisation avec Excel

Pour les utilisateurs qui ne disposent pas du logiciel OriginPro 7.5, il leur est possible de visualiser leurs résultats grâce au programme « Microsoft Excel ». Tracer ce même nuage de points est possible, et pour cela il suffit de :

1- Lancer en premier abord, le programme « Microsoft Excel » à partir du post de travail.

2- Après il faut charger le fichier issu de l'étude de l'ACP sur les vecteurs d'autocorrelation, en extension *.dat.*

Cela se fait comme suit : cliquer sur Fichier → Ouvrir → ensuite il faut chercher sur l'ordinateur le fichier *xxxx.dat* voulu → Il faut garder le choix de la largeur fixe → Ensuite il faut cliquer sur : **Suivant** → Mettre **les délimiteurs** s'il y en a pas → Et finir par cliquer sur **Suivant**, puis sur **Terminer**

3- Une fois le fichier affiché sur la fenêtre d'Excel, il ne faut pas oublier de changer tous les **points** par des **virgules**, comme ceci : Sélectionner tout le tableau → Après cliquer sur **Edition** → Puis aller sur **Remplacer…** → Saisir le caractère « **.** » dans le premier champ et le caractère « , » dans le deuxième → Et finir par cliquer sur **Ok**, puis sur **Fermer**

4- Après cela, il faut sélectionner deux colonnes sur tout le tableau

5- Dans la barre de menus, il faut cliquer sur l'icône avec un histogramme nommé « **Assistant graphique** », sinon l'ouvrir en cliquant sur **Insertion** → et puis sur **Graphique…**

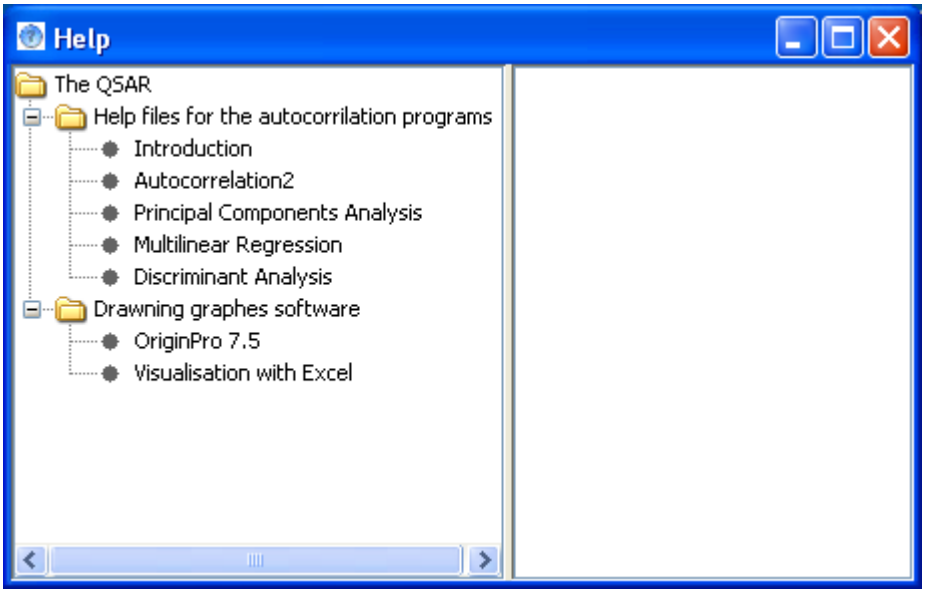
6- Pointer le choix sur « nuage de points » (à gauche de la fenêtre)

7- Finir par cliquer sur **Suivant** et puis sur **Terminer**

8- En fin aboutir au tracé du nuage de points, facilement interprétable.

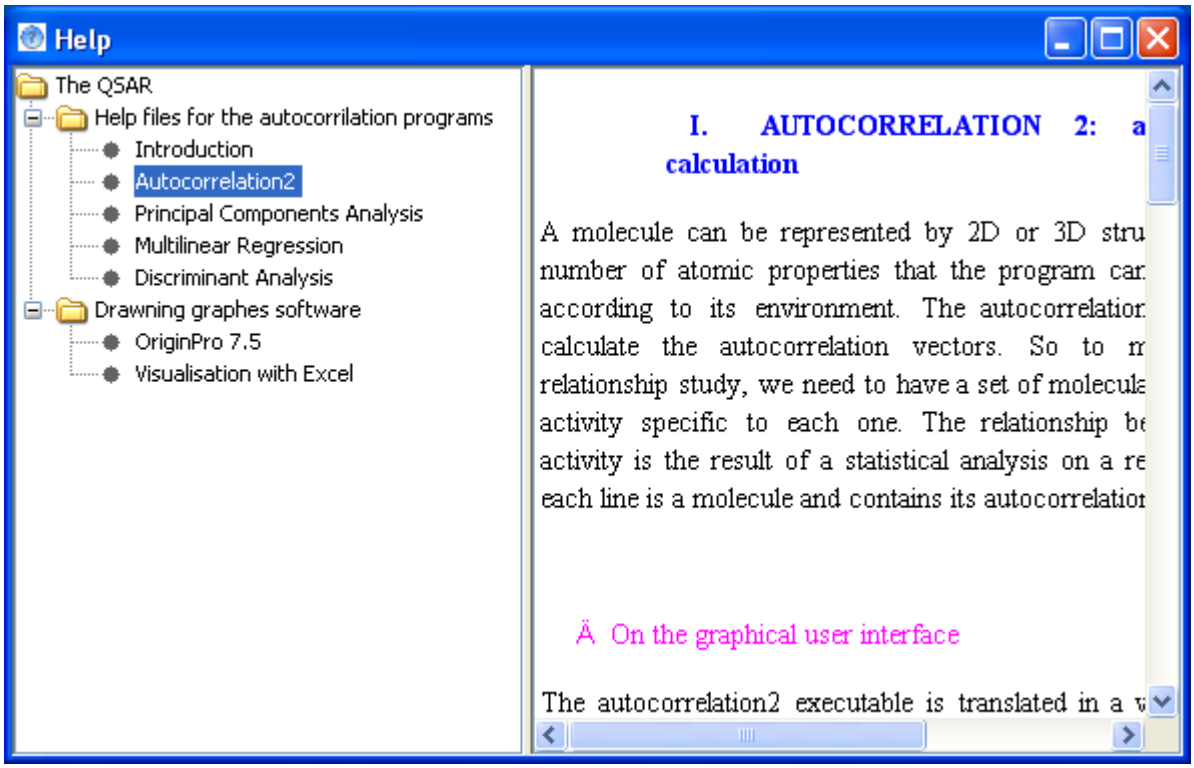
Le sixième et dernier bouton que comporte l'interface « Autocorrelation » est le « Help ». Il représente une aide contenant des instructions et des explications, conçues dans le

but d'aider n'importe quel utilisateur à travailler avec les outils statistiques QSAR, incorporés dans la plateforme VSM-G. En cliquant sur ce bouton, on ouvre la fenêtre suivante :

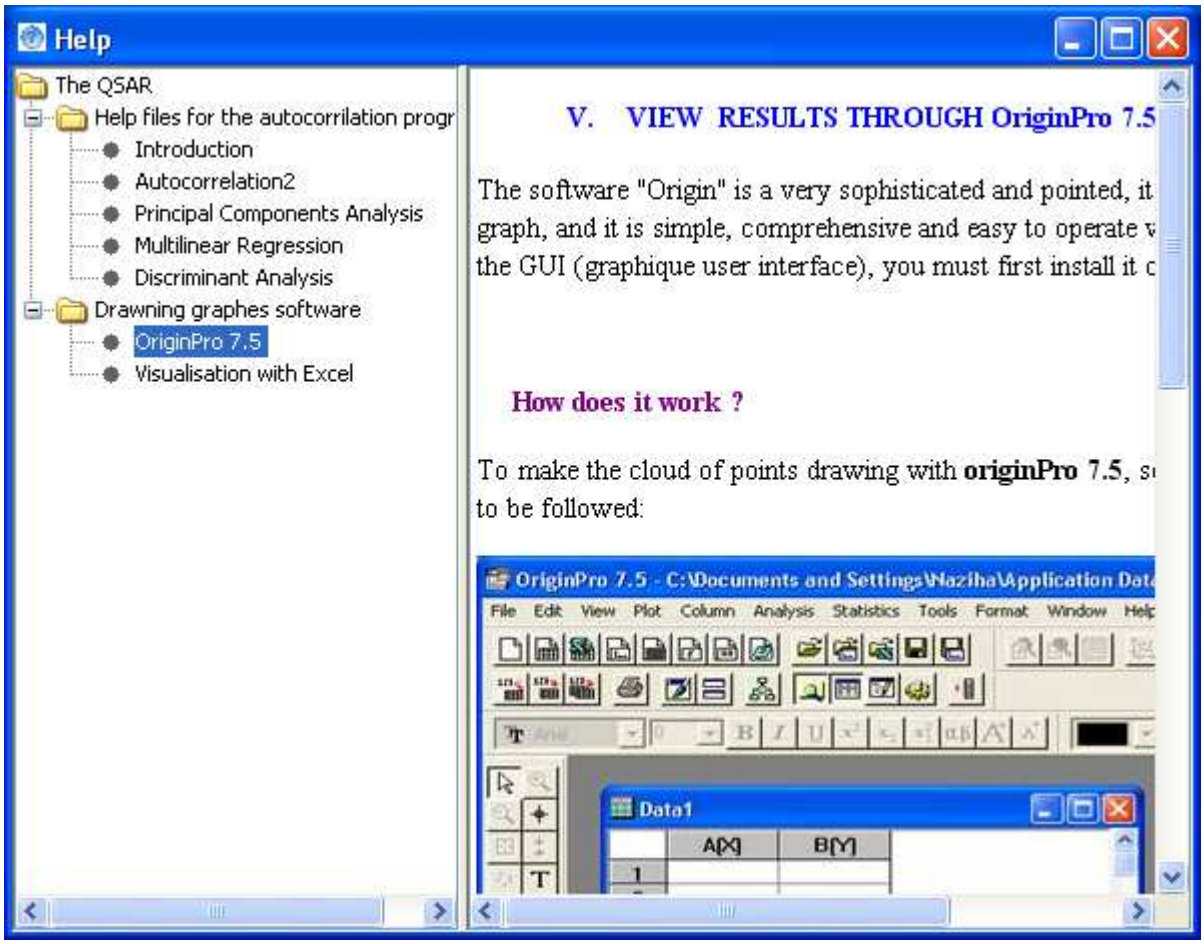


Cette aide emploi la logique de l'arborescence. Autrement dit, l'interface « Help » a été conçue, sur le plan des racines que contient un répertoire donné. Le premier panneau, se trouvant à gauche de l'interface, comporte le dossier nommé *QSAR* et deux sous dossiers. Chaque sous dossier est ouvert par un simple clic, et contient des éléments, tel que c'est illustré sur la fenêtre ci dessus.

Le premier sous dossier, portant le nom de *Help files for the autocorrelation programs*, englobe une introduction suivie des quatre programmes de l'autocorrelation, injectés dans la plateforme VSM-G. Le contenu de chacun de ces éléments, peut être visualisé sur le deuxième panneau se trouvant à droite de l'interface, en cliquant dessus.



Le deuxième sous dossier, portant le nom de *Drawning graphs software*, englobe les techniques qui servent à l'illustration graphique, des résultats issus des calculs de l'autocorrelation. Le contenu qui est étalé sur le panneau de droite, obtenu par un clic sur la technique souhaitée. Chacune des deux techniques représente les différentes façons de procéder, afin de bien exploiter cette dernière.



Le choix de conception de cette interface, reposant sur le système d'arborescence, est dû à son aspect agréable et le fait de présenter une façon facile de structurer le contenu de cette aide.

IV. INTEGRATION DE LA MAQUETTE DANS VSM-G

Comme la plateforme de screening VSM-G est écrite en langage Java, l'intégration de la maquette comportant tous les outils QSAR n'a pas été une tâche difficile. Il a juste fallu ajouter dans VSM-G, une partie de code correspondant à cette intégration. Nous avons commencé par déclarer le bouton QSAR qui représente le point de lancement de la maquette, à partir de VSM-G :

```
private JRadioButton rbLigands, rbCibles, rbQSAR, rbEntonnoir;
rbQSAR = new JRadioButton("QSAR ");
```

Nous avons ensuite ajouté un gestionnaire d'événement (*ActionListener*) ce bouton au panneau qui contient les trois autres boutons à savoir : Ligand Preparation, Target preparation et screening Funnel.

```
rbQSAR.addActionListener(new EcouteurQSAR());

panneau = new JPanel();
panneau.add(rbQSAR);

/** Classe interne : EcouteurQSAR qui permet de lancer la fenêtre
de la chaine de programmes de l'autocorrelation */

private class EcouteurQSAR implements ActionListener {
      public void actionPerformed(ActionEvent event){
            qsarFrame = new Programs();
      }
  }
```

Nous trouvons ci-dessous la nouvelle interface de VSM-G, contenant le bouton « QSAR ». Ce dernier, permet l'ouverture des interfaces contenant les programmes servant à faire de l'autocorrelation et des calculs QSAR.

V. VALIDATION DU PROTOTYPE IMPLEMENTE DANS VSM-G

Pour pouvoir tester et valider le bon fonctionnement du prototype que nous avons injecté dans la plateforme VSM-G, nous avons considéré un jeu de données comportant des ligands des récepteurs aux oxystérols (Liver X Receptors : LXR)[(80).]. Les structures sont

données en annexe. Ces récepteurs sont des protéines de la superfamille des récepteurs nucléaires [(81)] liant naturellement les oxystérols et régulant le métabolisme du cholestérol [(82)] et des lipides [(83)] dans l'organisme ainsi que la réponse inflammatoire au niveau des macrophages [(84)].

Nous avons commencé cette application par la création de deux fichiers textes « The Pure Sdf File » et « The IC50 File » contenant successivement les structures au format *sdf* de l'ensemble des molécules et leurs valeurs d'activités correspondantes. En utilisant l'interface « Sdf File Construction » nous avons créé, le fichier de structures moléculaires au format *sdf* contenant les informations liées à la structure de chaque molécule suivie de sa valeur d'activité. Nous avons calculé par la suite, les vecteurs d'autocorrelation, servant à mettre en évidence une corrélation entre les propriétés structurales et l'activité biologique, à l'aide de l'interface « Properties_Autocorrelation2 ». Nous avons utilisé les propriétés cochées sur l'interface (voir l'illustration de l'interface dans le paragraphe 6) et nous avons gardé le choix du mode sur « 2D+ ». Nous avons chargé le fichier au format *sdf* préalablement construit, et nous obtenons un fichier portant l'extension *.vec*, comportant les vecteurs d'autocorrelation que nous avons soumis à une analyse en composantes principales.

❖ ***Etude de l'analyse en composantes principalesv(ACP)***

A partir du fichier au format *.vec*, nous avons effectué une ACP permettant le calcul des nouvelles composantes des molécules portées par les facteurs principaux, dont les premiers apportent des informations. En chargeant ce dernier moyennant l'interface « Properties_Acplpk2 », et en fixant le nombre de facteurs principaux à calculer, à 40. Nous avons obtenu deux fichiers portant successivement les extensions *.dat* et *.acp* . Cette analyse nous a permis de générer les nouveaux axes principaux correspondant à des vecteurs propres, ordonnés par valeurs propres décroissantes :

Facteurs	Valeurs propres	Pourcentages cumulés
Facteur1	0.35E+04	0.33E+02
Facteur2	0.28E+04	0.60E+02
Facteur3	0.15E+04	0.74E+02
Facteur4	0.11E+04	0.85E+02
Facteur5	0.54E+03	0.90E+02
Facteur6	0.33E+03	0.93E+02

Facteur7	0.29E+03	0.96E+02
Facteur8	0.16E+03	0.97E+02
Facteur9	0.95E+02	0.98E+02
Facteur10	0.69E+02	0.99E+02
Facteur11	0.57E+02	0.99E+02
Facteur12	0.28E+02	0.99E+02
Facteur13	0.16E+02	0.10E+03
Facteur14	0.12E+02	0.10E+03
Facteur15	0.10E+02	0.10E+03
Facteur16	0.57E+01	0.10E+03
Facteur17	0.47E+01	0.10E+03
Facteur18	0.45E+01	0.10E+03
Facteur19	0.28E+01	0.10E+03
Facteur20	0.23E+01	0.10E+03

Tableau 1 : Valeurs propres et pourcentages cumulés issus de l'ACP

(0.23E+01 correspond à $0.23*10^{1}$)

Comme le tableau l'indique, la plus grande valeur propre correspond au premier facteur. Elle décroît jusqu'à devenir insignifiante à partir du quinzième facteur. Ces valeurs propres nous renseignent sur la quantité d'information contenue sur chaque axe factoriel. Les valeurs des pourcentages cumulés augmentent jusqu'à atteindre la limite, puis elles demeurent constantes.

A l'aide du programme OriginPro 7.5, lancé directement depuis l'interface « Autocorrelation », nous avons tracé le diagramme de corrélation des vingt premières valeurs propres de l'ACP (figure 1), ainsi que le nuage de points issu du fichier *.dat* (figure 2).

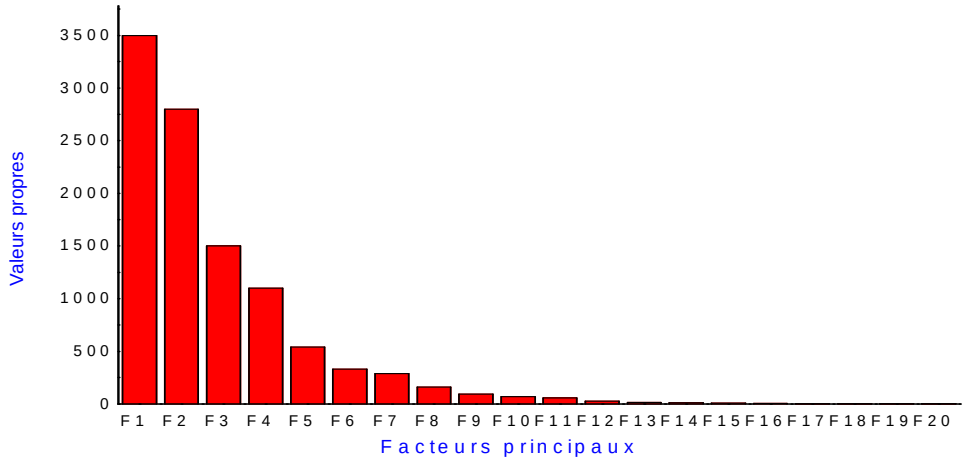
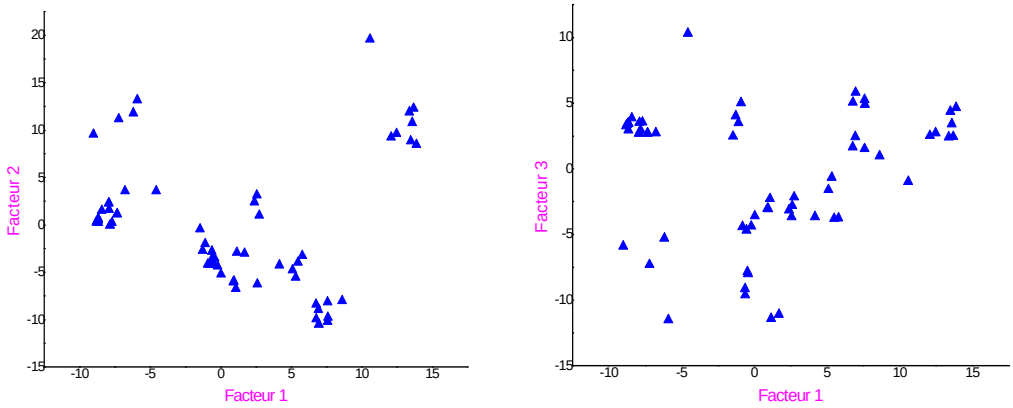


Figure 1 : Histogramme de corrélation des vingt premières valeurs propres de l'ACP

Cette étude d'ACP nous a permis de calculer les valeurs propres. Ces dernières montrent que le premier axe factoriel est celui comportant le maximum d'informations. Les axes qui suivent contiennent à leur tour des informations qui diminuent en atteignant le quinzième facteur.

a) b)



c) d)

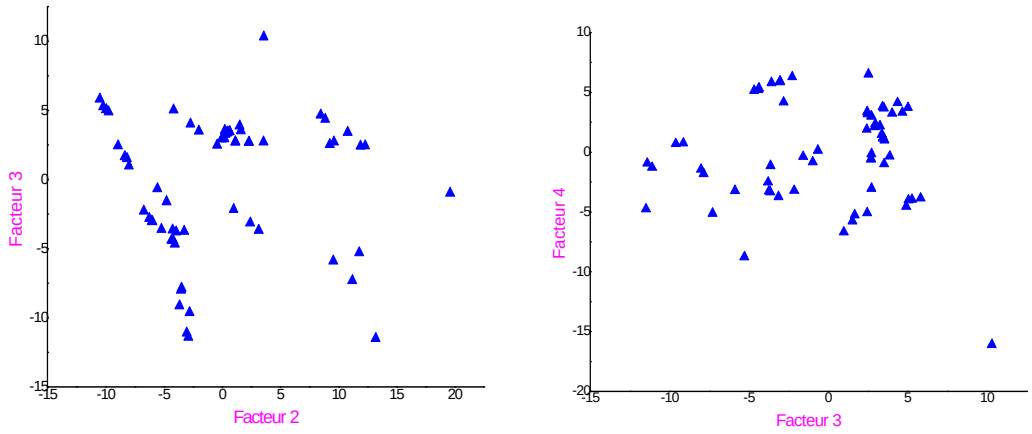


Figure 2: Distribution des points sur les plans factoriels

Ces quatre graphes, représentant la distribution des points sur les axes factoriels, nous montrent la dispersion du nuage de points des variables sur les premiers plans factoriels. Le graphe (a) représente la distribution du nuage de points sur le premier plan factoriel formé à partir des deux premiers facteurs principaux. Les trois graphes suivant représentent la dispersion du même nuage sur d'autres plans factoriels. Le premier plan, contenant le maximum d'information, représente la meilleure dispersion, par conséquent, le meilleur plan de projection.

❖ ***Etude de la régression multilinéaire RM***

A partir du fichier au format *.dat*, récupéré de la précédente étude, nous avons effectué une régression multilinéaire permettant de rechercher des régressions linéaires entre l'activité et les facteurs principaux issues de l'ACP sur des vecteurs d'autocorrelation. Moyennant l'interface « Properties_Regmul », nous avons inséré dans les champs à remplir, les critères représentant le nombre de facteurs à utiliser que nous avons fixé à 40, le nombre de variables indépendantes que nous avons choisi parmi ces facteurs et le nombre d'itérations à effectuer avec la méthode de Monté Carlo. Nous avons recherché des régressions à 5, puis 9, ensuite à 14 variables et cela en utilisant les vingt premiers facteurs et 200 itérations. Les coefficients de corrélations obtenus dans les trois cas sont respectivement 0.78975, 0.85534 et 0.90304.

Afin d'avoir une meilleure valeur de coefficient de corrélation, nous avons effectué un calcul en utilisant les 40 premiers facteurs. Ce calcul a conduit à un coefficient de corrélation de 0.99464.

Nous avons reporté les valeurs des activités observées en fonction de celles prédites sur le graphe que nous présentons ci dessous :

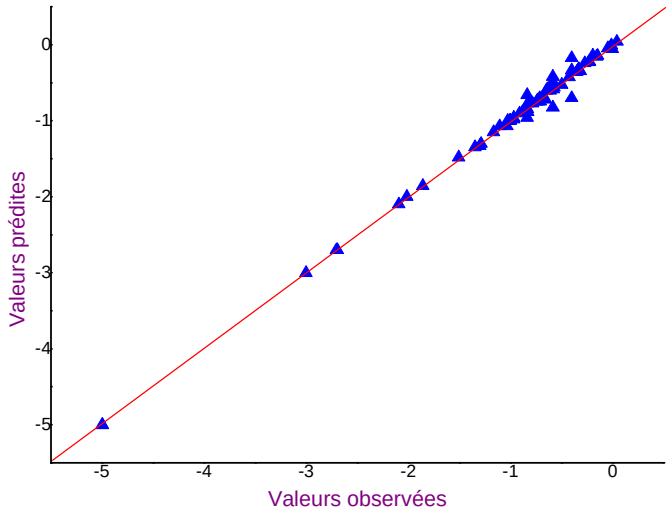


Figure 3 : Distribution des activités observées et prédites autour de la droite idéale recueillie du programme Regmul

Ces valeurs d'activité donnent une bonne distribution autour de la droite idéale. Ceci nous permet de conclure sur la qualité du calcul des propriétés des molécules.

Afin de pouvoir valider ce modèle, nous avons retiré au hasard 7 molécules du jeu de données de départ. Nous avons formé deux nouveaux fichiers de données, le premier est le fichier d'apprentissage, et le deuxième est le fichier d'essai qui contient les molécules retirées. Nous avons calculé pour le lot d'essai les vecteurs d'autocorrelation de la même façon que pour le fichier de départ et en gardant les mêmes propriétés. Nous avons ensuite remis les molécules du lot d'essai dans le même repère factoriel que celui du lot d'apprentissage. Nous avons pu prédire les activités des molécules du lot d'essai en utilisant le modèle QSAR obtenu pour le lot d'apprentissage. Pour illustrer cela, nous donnons ci dessous la distribution des molécules des deux lots dans le premier plan factoriel :

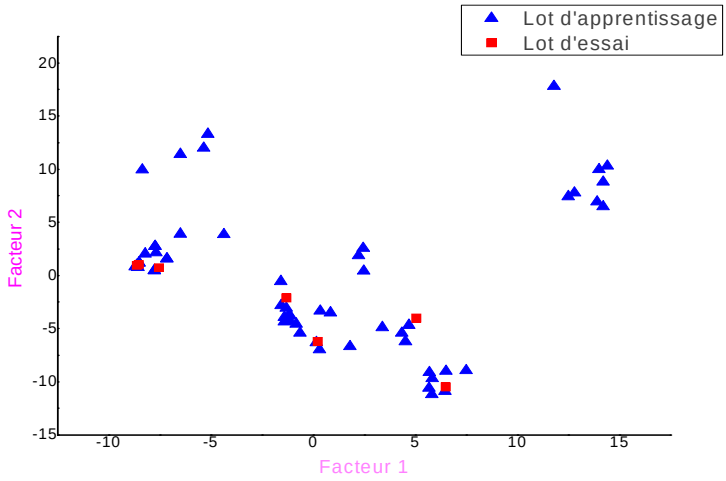


Figure 4 : Représentation des molécules des lots d'apprentissage et d'essai dans le premier plan factoriel.

Nous remarquons que la distribution du nuage de points du lot d'essai est semblable à celle du lot d'apprentissage.

Nous avons effectué de nouveau une étude de régression multilinéaire sur les molécules du lot d'essai, en utilisant les mêmes conditions que celles déjà utilisées pour les molécules du lot d'apprentissage, et nous avons obtenu un coefficient de corrélation : $R = 0.99529$. Nous donnons ci dessous la représentation de la régression. En rouge, sont représentés les activités des molécules du lot d'essai, et en bleu celles du lot d'apprentissage.

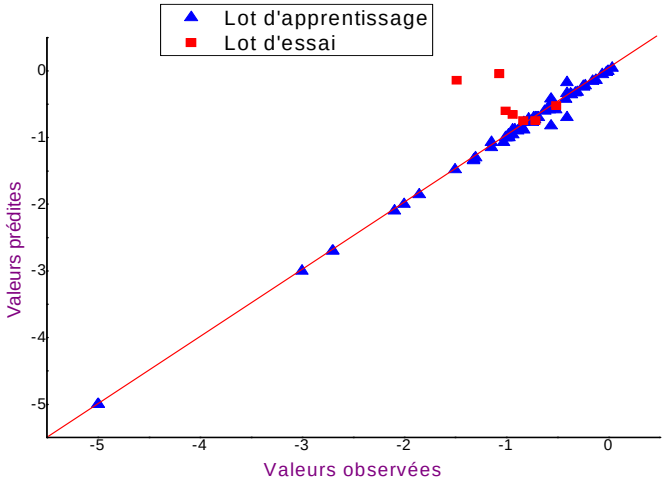


Figure 5 : Distribution des activités observées et prédites par le modèle QSAR autour de la droite idéale

❖ ***Etude de l'analyse discriminante***

Partant du fichier contenant le lot d'apprentissage, en format *.dat*, nous avons effectué une étude d'analyse discriminante permettant de classer qualitativement une molécule d'activité inconnue et cela en se basant sur les composantes principales. Nous avons rempli les champs de l'interface « Properties_Anadis » en prenant les facteurs principaux de 1 à 8 et en fixant la valeur du seuil d'activité à 0.5. Nous avons divisé le fichier de départ (le fichier du lot d'apprentissage), en deux classes selon qu'elles soient actives ou non. Suivant les valeurs d'activités de ces dernières, nous avons regroupé les molécules actives dans l'intervalle [0.001, 0.5], et les molécules peu actives ou inactives dans l'intervalle [0.6, 1.1]. Nous avons récupéré le fichier des résultats au format *.mda.*

Le fichier au format *mda* nous renseigne sur le reclassement des molécules dans les deux précédentes classes. En effet, nous avons constaté que les molécules actives sont classées selon un pourcentage de 68.08% de succès, et les molécules peu actives sont classées selon un pourcentage de 83.33% de succès. Ces pourcentages nous donnent une idée concernant les capacités de prédiction de ce programme. Nous avons testé ce modèle sur le lot d'essai et nous avons eu un résultat de 100% pour les deux classes de molécules. Le tableau ci-dessous représente le classement des molécules avant et après l'étude de l'analyse discriminante :

Numéro de la molécule	**Avant ADM**	**Après ADM**
1	2	2
2	1	1
3	1	1
4	1	1
5	1	1
6	2	2
7	1	1

Tableau 2 : Résultats de l'analyse discriminante sur les molécules du lot d'essai
Avant ADM : correspond au classement défini par l'activité réelle
Après ADM : correspond au classement déterminé par l'analyse discriminante.

Nous illustrons les résultats de cette étude en présentant ce dessous le diagramme de la répartition des molécules :

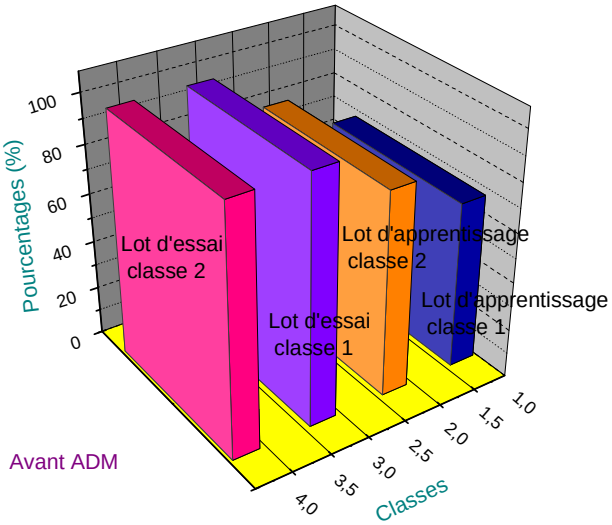


***Figure 6* :** Représentation de la répartition des molécules avant l'ADM

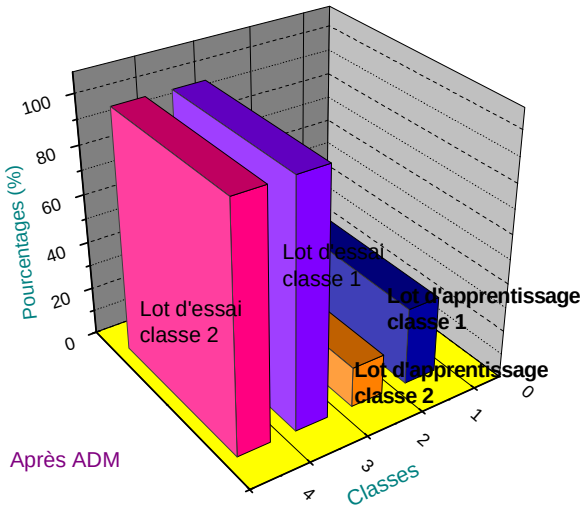


***Figure 7* :** Représentation de la répartition des molécules après l'ADM

VI. CONCLUSION

Les nouvelles interfaces sous lesquelles apparaissent les outils QSAR insérés dans la plateforme de screening virtuel VSM-G, montrent une souplesse comparable à celle dont témoignent les interfaces de VSM-G. Les nouvelles interfaces des programmes de l'autocorrelation, reproduisent le contenu de chacun de ces derniers. Les contraintes qu'ils présentaient, ont été supprimées. En effet, tous ces exécutables fonctionnaient en interactivité, sous un système de fenêtre DOS. L'ensemble des questions engendrées par chacune de ces techniques, a été repris, en éliminant celles jugées non indispensables à la visualisation. Effectuer des études statistiques à partir de VSM-G est devenu aujourd'hui, possible et automatique et cela a été validé sur un jeu de données qui donne des résultats satisfaisants.

REFERENCES BIBLIOGRAPHIQUES

1- http://fr.wikipedia.org/wiki/Interface_graphique

2- http://www.loria.fr

3- http://www.commentcamarche.net/contents/java/javaintro.php3

4- http://fr.wikipedia.org/wiki/Langage_de_programmation

5- http://fr.wikipedia.org/wiki/Orient%C3%A9_objet

6- http://fr.wikipedia.org/wiki/James_Gosling

7- http://fr.wikipedia.org/wiki/Patrick_Naughton

8- http://fr.wikipedia.org/wiki/Sun_Microsystems

9- http://fr.wikipedia.org/wiki/Bill_Joy

10- http://fr.wikipedia.org/wiki/Portabilit%C3%A9_(informatique)

11- http://fr.wikipedia.org/wiki/Syst%C3%A8me_d%27exploitation

12- http://fr.wikipedia.org/wiki/Unix

13- http://fr.wikipedia.org/wiki/Windows

14- http://fr.wikipedia.org/wiki/Mac_OS

15- http://fr.wikipedia.org/wiki/Linux

16- http://www.commentcamarche.net/contents/cpp/cppintro.php3

17- http://fr.wikipedia.org/wiki/H%C3%A9ritage_multiple

18- http://fr.wikipedia.org/wiki/JavaOS

19- http://fr.wikipedia.org/wiki/JDK

20- http://fr.wikipedia.org/wiki/Environnement_d%27ex%C3%A9cution_Java

21- http://fr.wikipedia.org/wiki/Machine_virtuelle_Java

22- http://fr.wikipedia.org/wiki/Java_2_Micro_Edition

23- http://fr.wikipedia.org/wiki/Swing_(Java)

24- http://fr.wikipedia.org/wiki/Classe_(informatique)

25- http://fr.wikipedia.org/wiki/M%C3%A9thode_(informatique)

26- http://fr.wikipedia.org/wiki/H%C3%A9ritage_(Informatique)

27- http://fr.wikipedia.org/wiki/Bytecode

28- http://www.larcher.com/eric/guides/javactivex/IV.htm

29- http://www.commentcamarche.net/contents/c/cpoint.php3

30- http://www.dotnetguru.org/articles/GC/GC.html

31- http://fr.wikipedia.org/wiki/Virus_informatique

32- http://java.sun.com/j2se/1.4.2/docs/api/java/lang/SecurityManager.html

33- http://www.3ie.fr/nouvelles_technologies/fiche/fiche_Java.htm
34- http://fr.wikipedia.org/wiki/Framework
35- http://fr.wikipedia.org/wiki/Application_programming_interface
36- http://fr.wikipedia.org/wiki/J2SE
37- http://fr.wikipedia.org/wiki/J2EE
38- http://fr.wikipedia.org/wiki/J2ME
39- http://java.sun.com/javacard/
40- http://java.sun.com/j2se/1.4.2/docs/tooldocs/windows/javac.html
41- http://fr.wikipedia.org/wiki/Environnement_de_d%C3%A9veloppement_int%C3%A9gr%C3%A9
42- http://fr.wikipedia.org/w/index.php?title=CodeWarrior&action=edit&redlink=1
43- http://www.jetbrains.com/idea/
44- http://www.borland.com/jbuilder/
45- http://fr.wikipedia.org/wiki/NetBeans
46- http://www.oracle.com/technology/products/jdev/index.html
47- http://fr.wikipedia.org/wiki/Xcode
48- http://www.jcreator.com/
49- http://fr.wikipedia.org/wiki/Eclipse_(environnement_de_d%C3%A9veloppement)
50- http://www.bluej.org/
51- http://fr.wikipedia.org/wiki/Abstract_Window_Toolkit
52- http://dictionnaire.phpmyvisites.net/definition-AWT--7006.htm
53- http://fr.wikipedia.org/wiki/Swing_(Java)
54- http://java.sun.com/docs/books/tutorial/uiswing/
55- http://java.sun.com/docs/books/tutorial/uiswing/components/button.html
56- http://java.sun.com/docs/books/tutorial/uiswing/components/textfield.html
57- http://java.sun.com/docs/books/tutorial/uiswing/components/label.html
58- http://www.commentcamarche.net/java/javapack.php3
59- http://java.sun.com/docs/books/tutorial/uiswing/layout/using.html
60- http://java.sun.com/docs/books/tutorial/uiswing/events/actionlistener.html
61- http://java.sun.com/j2se/1.4.2/docs/api/java/awt/Frame.html
62- http://fr.wikipedia.org/wiki/PC-DOS
63- http://java.sun.com/j2se/1.4.2/docs/api/javax/swing/JButton.html
64- http://java.sun.com/docs/books/tutorial/uiswing/components/dialog.html
65- http://java.sun.com/j2se/1.4.2/docs/api/javax/swing/JTextField.html

66- http://en.wikipedia.org/wiki/Chemical_table_file

67- http://www.epa.gov/ncct/dsstox/MoreonSDF.html

68- Programmes écrits par Gilles Moreau utilisant la librairie mathématique LAPACK.

69- c'est une adaptation du programme régression de Legendre http://www.bio.umontreal.ca/casgrain/fr/telecharger/index.html#Rgressionlinairemultiple

70- C'est une adaptation du programme mda par Murtagh trouvé sur le web : http://astro.ustrasbg.fr/~fmurtagh/mda-sw/

71- OriginPro 7.5 SRO, © Copyright, **1991-2003 ;** OriginLab Corporation

72- http://fr.wikipedia.org/wiki/SVM

73- http://pubs.acs.org/cgi-bin/abstract.cgi/jcisd8/2008/48/i04/abs/ci700368p.html

74- http://www.ccl.net/cgi-bin/ccl/message-new?2002+12+05+005

75- ISIS™/Draw 2.3., © Copyright, **1990-2000 ;** MDL Information Systems, Inc.

76- http://www.ch.cam.ac.uk/cil/SGTL/MDL/ISISbase.html

77- http://www.ccdc.cam.ac.uk/support/documentation/mercury_csd/portable

78- http://graphics.med.yale.edu:5080/TriposBookshelf/sybyl/toolkit/mol_format.html

79- http://fr.wikipedia.org/wiki/M%C3%A9thode_de_Monte-Carlo

80- http://fr.wikipedia.org/wiki/Liver_X_receptor

81- http://fr.wikipedia.org/wiki/R%C3%A9cepteur_nucl%C3%A9aire

82- http://fr.wikipedia.org/wiki/Cholest%C3%A9rol

83- http://fr.wikipedia.org/wiki/Lipide

84- Zelcer, N., Tontonoz, P., Liver X receptors as integrators of metabolic and inflammatory signalling, J.Clin. Invest, 7, 607-614, **2006**

CHAPITRE III

APPLICATION DE L'AUTOCORRELATION AUX LIGANDS DES RÉCEPTEURS DE LA CHOLECYSTOKININE

I. INTRODUCTION :

La maquette comportant les outils statistiques QSAR, représentant les programmes de l'autocorrélation que nous avons développés, est maintenant intégrée à la plateforme de screening virtuel VSM-G. Dans le but d'effectuer une étude QSAR à travers la plateforme VSM-G, nous avons fait une application sur la base de données comportant les ligands des récepteurs de la cholécystokinine (CCK).

II. LA CHOLECYSTOKININE

II.1. Généralités

La cholécystokinine (CCK), est une hormone neuropeptidique connue pour ses actions régulatrices dans le système digestif et le système nerveux central des mammifères. Le nom de cette hormone provient de l'étymologie grecque signifiant hormone contractant la vésicule biliaire (kholé : bile, kustis : vessie ou sac, kinine : circuler, mouvoir). La CCK fut largement décrite dans un travail antérieur[(1)], nous nous limiterons dans ce chapitre à actualiser des connaissances relatives à cette hormone endogène et ses récepteurs.

La CCK participe à un grand nombre d'actions physiologiques et physiopathologiques. Cette hormone est localisée dans le système gastro-intestinal des mammifères[(1)].

Différentes formes moléculaires de la CCK sont connues, cette diversité des formes peut contenir de 4 à 83 acides aminés, et dérive de l'hydrolyse d'une préprohormone constituée de 115 acides aminés lors de son processus de maturation. L'activité biologique optimale de l'hormone est donnée par l'amidation de la phénylalaline et la sulfatation de la tyrosine (figure 1).

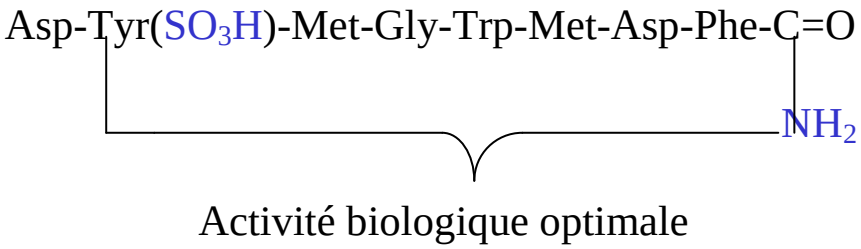


Figure 1 : Représentation de la chaîne peptidique de la CCK-8 sulfatée

La CCK est principalement localisée et produite dans le cortex cérébral (1 à 2mg dans le cerveau humain). Elle se situe également au niveau de l'hippocampe, les bulbes olfactifs, le septum, l'amygdale, l'hypothalamus et l'épine dorsale de la moelle épinière. Au niveau du

système gastrique, la CCK est produite par les cellules I endocrines de la muqueuse se trouvant dans le duodénum, le jéjunum et l'intestin proximal.

II.2. Les récepteurs de la cholécystokine

Les différents sites de liaison des récepteurs de la CCK sont possibles à caractériser et à localiser grâce à l'utilisation de ligands, agonistes et antagonistes. Tel est le cas pour la CCK-8[(2)] dont la sulfatation[(3)] laisse apparaître 2 types de récepteur de la CCK : le premier, noté CCK_1 localisé dans le système nerveux périphérique. Il représente une cible potentielle pour le traitement de nombreuses pathologies liées à la prise alimentaire et à la digestion. Le deuxième, noté CCK_2 est présent dans le cortex cérébral et dans le pancréas de certains mammifères.

Les gènes des récepteurs de la CCK ont pu être localisés grâce au clonage. Ainsi le gène du récepteur CCK_1 est situé sur le chromosome 4 et celui du récepteur CCK_2 se trouve sur le chromosome 11. La structure secondaire de ces récepteurs a été déterminée à partir de leurs séquences primaires et de leur profil d'hydrophobicité. Ces 2 récepteurs membranaires, de nature glycoprotéinique, présentent une homologie de 50% au niveau de leur structure primaire. Il existe une homologie de 80% entre deux récepteurs de même nature CCK (CCK1 ou CCK_2) appartenant à des espèces différentes.

Les récepteurs CCK_1 et CCK_2 sont des protéines membranaires appartenant à la famille des récepteurs couplés aux protéines G (RCPGs) présentant :

- ✓ 7 domaines transmembranaires de structure hélicoïdale reliés par 3 boucles intra et extra cellulaires
- ✓ Une extrémité N-terminale extra cellulaire et une autre C-terminale intra cellulaire
- ✓ 2 cystéines conservées situées dans la $1^{ière}$ et la $2^{ième}$ boucle extra cellulaire, et formant le pont disulfure.

Les récepteurs de la CCK présentent des difficultés dans le processus de leur cristallisation; ce qui ne facilite pas l'accès à la structure 3D expérimentale de ces derniers. La famille, de ces récepteurs couplés aux protéines G est largement présente dans le corps humain et joue un rôle physiologique important.

II.3. Rôles physiologiques de la cholécystokinine

La CCK agit en tant qu'hormone dans le système gastro-intestinal, en tant que neurotransmetteur–neuromodulateur sur le système nerveux central et périphérique.

1- *Dans le système gastro-intestinal, la CCK :*

- ✓ Régule la concentration des organes tels que la vésicule biliaire et la paroi intestinale
- ✓ Stimule selon le besoin : la sécrétion d'enzymes digestives, en hydrolysant les hydrates de carbone, les graisses, les protéines et les acides nucléiques dans le pancréas exocrine, et la sécrétion de glucagon, d'insuline, de somatostatine et d'autres polypeptides pancréatiques dans le pancréas endocrine.
- ✓ Maintient la croissance normale du pancréas; ceci a été observé suite à l'atrophie induite lors d'une administration d'antagoniste ou par la diminution du taux de CCK dans le sang. Cependant, l'excès de cette dernière dans le sang, conduit au développement de nodules précancéreux du pancréas[(4)].

2- *Dans le système nerveux central et périphérique, la CCK :*

- ✓ Régule la prise alimentaire (satiété) : l'administration de fortes doses de CCK par voie centrale ou périphérique, conduit à une réduction de prise alimentaire. Cependant, les troubles de l'appétit telles que l'anorexie et la boulimie[(1-4-5)] peuvent être traités par des agonistes de cette hormone.
- ✓ Agit sur les capacités analgésiques (analgésie opioïde) : L'administration de la CCK atténue l'amplitude et la durée de la réponse analgésique à la morphine. Ceci implique que les antagonistes de la CCK vont dans le sens des opiacés dans le traitement de la douleur [(6-7)].
- ✓ Peut provoquer des troubles psychiques et un comportement d'anxiété : des crises d'anxiété, et des attaques de panique (chez l'homme et également chez l'animal) sont provoquées par l'administration de la CCK. Les antagonistes du récepteur CCK_2 présentent un réel potentiel dans le traitement des maladies neuropsychiatriques telles que la schizophrénie[(8)].

- ✓ Participe aussi au processus de mémorisation et d'apprentissage : fortement abondante dans le cortex cérébral et l'hippocampe[(9).]

3- *Conclusion*

La diversité des actions biologiques (physiologiques et physiopathologiques) de la CCK fait ressortir la complexité de son mode d'action, et a conduit à la découverte de nombreux ligands, potentiellement sélectifs sur les récepteurs CCK_1 et CCK_2. Sa distribution dans l'organisme et son activité à des doses très faibles (µg/kg) font d'elle une cible de choix dans le développement de nouvelles molécules à visées thérapeutiques.

III. APPLICATION DE L'AUTOCORRELATION

L'autocorrélation est une méthode qui s'applique à des familles de molécules ayant des structures différentes mais le même profile pharmacologique. Afin d'appliquer cette technique, il est nécessaire d'avoir les structures planes (2D) des molécules. Le but de cette étude consiste en la génération des vecteurs d'autocorrélation. Ces vecteurs, parviennent de la transformation d'une matrice de connectivité, permettant des études de reconnaissance de forme. Les composantes de ces vecteurs représentent la façon dont la propriété physique utilisée est distribuée sur la structure moléculaire.

Moreau et Broto [(10-11)] furent les premiers à publier l'utilisation de ces vecteurs comme des descripteurs moléculaires. Ils ont appliqué la notion mathématique d'une fonction d'autocorrélation à la structure topologique des molécules.

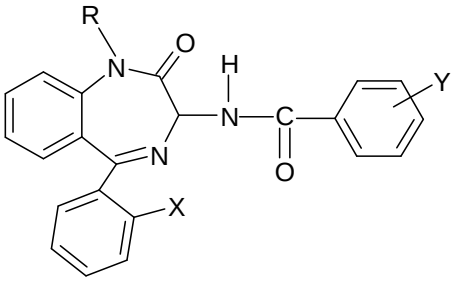
III.1 Méthodologie utilisée

Notre application, servant à tester le prototype injecté dans VSM-G, a été faite sur une base de données comportant des dérivés des Benzodiazépines. Les molécules constituant cette base, ont été sélectionnées dans la littérature et appartiennent à des familles différentes. Ces molécules présentent une diversité structurale et biologique adéquate pour une analyse statistique.

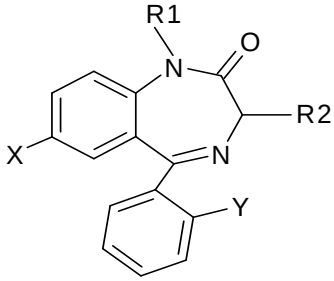
Les activités de ces molécules sont estimées en termes de cologarithme du IC_{50}, qui représente la concentration nécessaire pour inhiber à 50% l'activité du récepteur CCK_1.

Comme nous avons tiré ces molécules de publications différentes[12-13] et pour minimiser la différence des conditions expérimentales, nous avons calculé les activités relatives, évaluées par rapport à la même molécule de référence (Benzodiazépine). Nous donnons en annexe le tableau comprenant les structures et les valeurs de leurs activités relatives. Le nombre de composés dans chaque famille est donné en gras entre parenthèses dans la figure 2.

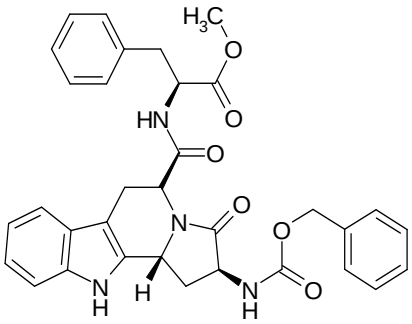
Pour des raisons d'espace, nous nous sommes limités dans la *figure 2*, à ne donner que la structure générique des molécules appartenant à ces familles.



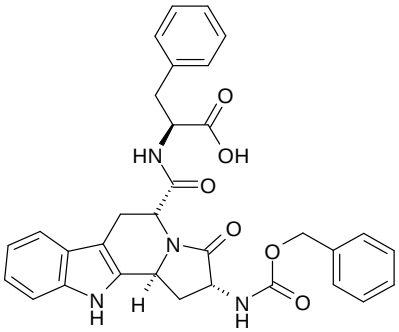
3-(Benzoylamino) benzodiazepines [12]
(**38 composés**)



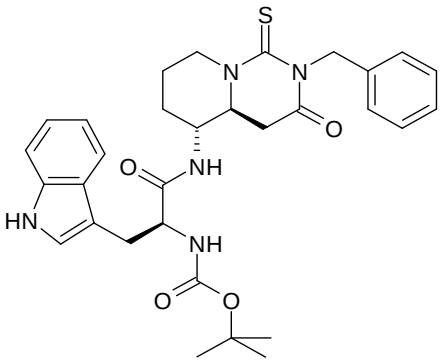
Benzodiazepine dérivatives [13]
(**20 composés**)



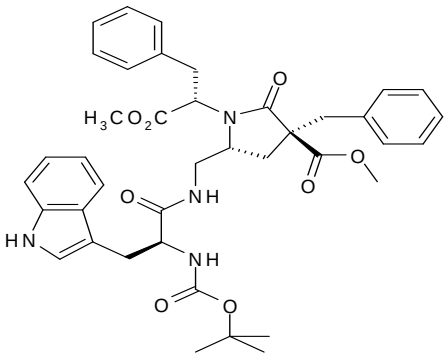
Famille des Dipeptoides
(04 composés) [14]



Famille des Pyridopyrimidines_Trp
(04 composés) [15]



Famille des Pyridopyrimidines_esquelto
(04 composés) [16]



Famille dePyridopyrimidines_esquelto
(02 composés) [17]

Figure 2 : Structures des molécules de la première chimiothèque

Pour pouvoir faire tourner les programmes d'Autocorrélation à partir de VSM-G, nous avons construit, dans un premier temps, les structures de l'ensemble des molécules constituant la base de données à l'aide du programme ISIS/DRAW [18] . Nous avons ensuite concaténé les fichiers contenant uniquement les structures de cet ensemble pour aboutir au fichier dit « pur » au format sdf. Moyennant l'interface « Sdf File Construction », nous avons créé, le fichier de structures moléculaires au format *sdf* contenant les informations topologiques et les activités des molécules. Dans un deuxième temps, nous avons lancé le calcul des vecteurs d'autocorrélation à partir de l'interface « Properties_Autocorrelation2 ». Nous avons choisi pour cela, les 8 propriétés suivantes : l'électronégativité, le volume de van der Waals, la contribution de chaque atome au logP, la faculté pour un atome à être accepteur ou donneur d'hydrogène dans la formation de la liaison hydrogène, la fonctionnalité Π (degré de saturation de l'atome), les charges partielles et l'aire de la surface polaire (PSA). Nous avons également pris le mode 2D+ pour cette autocorrélation, et nous avons chargé le fichier au format *sdf* construit dans la première étape.

III.2 Analyse statistique

A) Analyse en composantes principales

Nous avons récupéré les vecteurs d'autocorrélation dans un document « Notepad », et nous les avons soumis à une étude d'analyse en composantes principales moyennant l'interface « Properties_Acplpk2 ».

Étant donné que l'information apportée par une composante d'un rang donné est partagée, et pour que la représentation soit plus lisible et plus efficace, l'analyse en

composantes principales tente de trouver une représentation plus compacte non redondante, des structures. L'idée fondamentale est de trouver la dimension intrinsèque du domaine, c'est-à-dire la dimension minimale permettant de représenter les données sans perte d'information ou presque.

Pour ce faire, nous avons appliqué la technique de l'ACP sur les vecteurs d'autocorrélation. Le principe du programme Acplpk2[(19)] consiste en deux traitements, à savoir le centrage et la réduction des données : le centrage est la transformation de A(d) en A'(d)
Le centrage consiste à ramener les coordonnées initiales, à leur barycentre :

$$A'(d) = A(d) - \overline{A}(d)$$

$\overline{A}(d)$ représente la moyenne de A(d)

C'est-à-dire que le programme examine la variance dans chaque colonne (sur chaque coordonnée initiale). Si cette variance est trop faible (la coordonnée est presque la même pour toutes les molécules), alors il supprime cette coordonnée.

Le programme ensuite, calcule les axes principaux d'allongement, ou les axes factoriels, puis fait une réduction. La réduction est la transformation de A'(d) en A''(d). Le transposé du tableau centré réduit est X (il s'appelle matrice de corrélation). Ce tableau est carré dont la dimension est le nombre de colonnes (généralement plus petit que le nombre de lignes).

Ensuite ce programme recherche des valeurs propres et des vecteurs propres de cette matrice. Ce sont ces vecteurs unités qui définissent chaque axe factoriel. La valeur propre associée à chaque vecteur propre est la variance $\sum X_i^{\,2}$ selon cet axe.

Ensuite il calcule les nouvelles coordonnées X_i selon chaque axe et il finit par écrire le fichier de sortie comme suit :

	Nom de molécule	F1	F2	F3	……F40	Activité
Mol 1	----------	--	--	--	--	
Mol 2	----------	--	--	--	--	
Mol 3	----------	--	--	--	--	

Fi = facteur principale numéro i

Dans l'etude d'analyse en composantes principales que nous avons effectuée moyennant l'interface « Properties_Acplpk2 », nous avons pris les 40 premiers facteurs principaux, et nous avons recueilli les fichiers des résultats en extension *.dat* et *.acp*. Cette ACP nous a

permis de générer de nouveaux axes correspondant à des vecteurs propres, ordonnés par valeurs propres décroissantes :

Facteurs	**Valeurs propres**	**Pourcentages cumulés**
Facteur1	0.88E+04	0.73E+02
Facteur2	0.13E+04	0.84E+02
Facteur3	0.98E+03	0.92E+02
Facteur4	0.38E+03	0.95E+02
Facteur5	0.18E+03	0.97E+02
Facteur6	0.13E+03	0.98E+02
Facteur7	0.92E+02	0.99E+02
Facteur8	0.70E+02	0.99E+02
Facteur9	0.28E+02	0.99E+02
Facteur10	0.19E+02	0.10E+03
Facteur11	0.17E+02	0.10E+03
Facteur12	0.88E+01	0.10E+03
Facteur13	0.56E+01	0.10E+03
Facteur14	0.40E+01	0.10E+03
Facteur15	0.25E+01	0.10E+03
Facteur16	0.15E+01	0.10E+03
Facteur17	0.83E+00	0.10E+03
Facteur18	0.62E+00	0.10E+03
Facteur19	0.27E+00	0.10E+03
Facteur20	0.22E+00	0.10E+03

Tableau 1 : Valeurs propres et pourcentages cumulés issus de l'ACP

(0.15E+01 correspond à $0.15*10^{1}$)

Nous remarquons, à partir de ce tableau, que la plus grande valeur propre correspond au premier facteur. Elle décroit jusqu'à devenir insignifiante à partir du quinzième facteur. Ces valeurs propres expliquent la quantité d'information que contient l'axe factoriel correspondant. Les valeurs des pourcentages cumulés augmentent jusqu'à atteindre la limite, puis elles restent

constantes. Le pourcentage cumulé d'un facteur f_n est la somme des pourcentages de variance expliquée par les *n* premiers facteurs. Le pourcentage de variance expliqué par un facteur mesure l'épaisseur du nuage sur le sous-espace des composantes principales. Plus ce pourcentage est grand, meilleure est la représentation des données dans le sous-espace.

Moyennant OriginPro 7.5[(20)] qu'on lance directement depuis l'interface de VSM-G, nous avons pu tracer le diagramme de corrélation des vingt premières valeurs propres de l'ACP (figure 3) et le nuage de points issu du fichier *.dat* (figure 4).

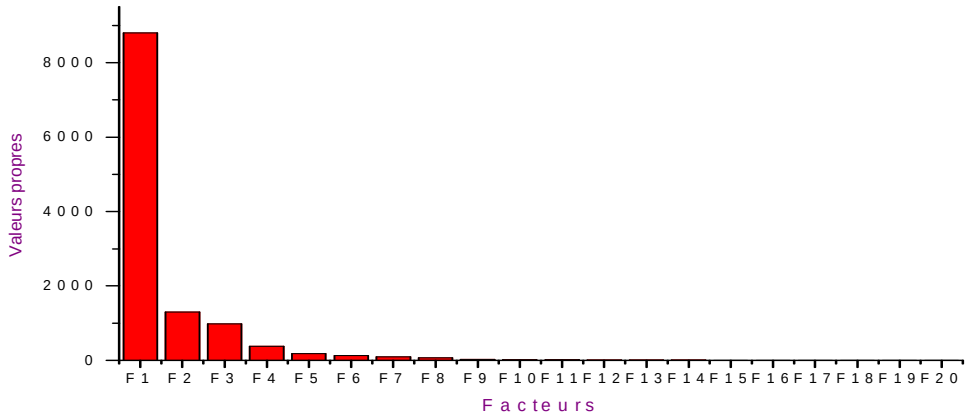


Figure 3 : Diagramme de corrélation des vingt premières valeurs propres de l'ACP

Le calcul des valeurs propres montre que le premier axe factoriel est celui contenant le maximum d'informations. Les axes suivants contiennent également des informations, mais beaucoup moins que le premier. En conséquence, la représentation du nuage de points des variables sur le premier plan factoriel est dispersée.

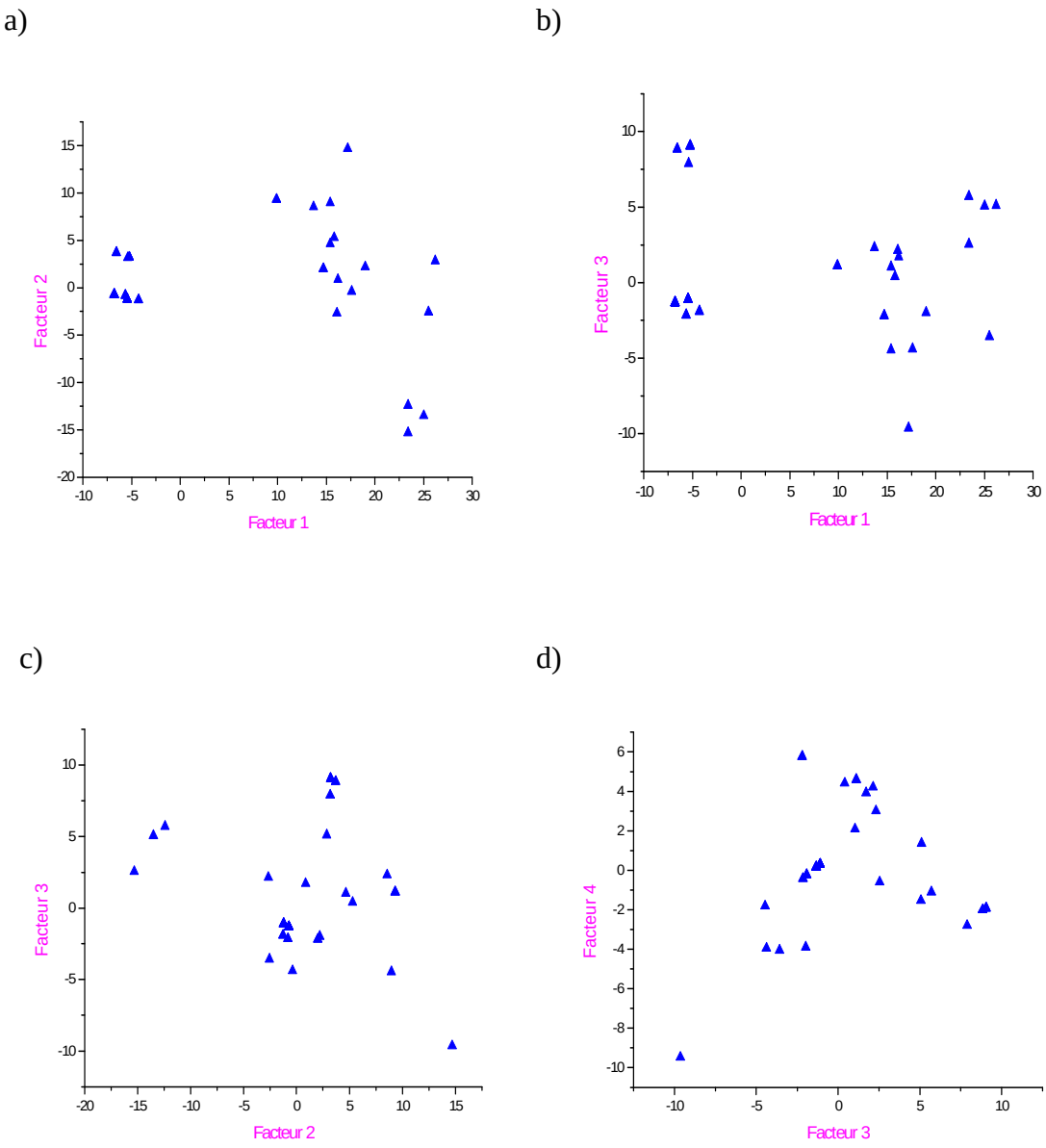


Figure 4 : Distribution des points sur les plans factoriels

Le graphe (a) représente la distribution du nuage de points sur le premier plan formé des deux premiers facteurs principaux issus de l'ACP. Il indique bien la dispersion du nuage la plus importante. Il correspond au meilleur plan de projection car il contient la meilleure dispersion du nuage.

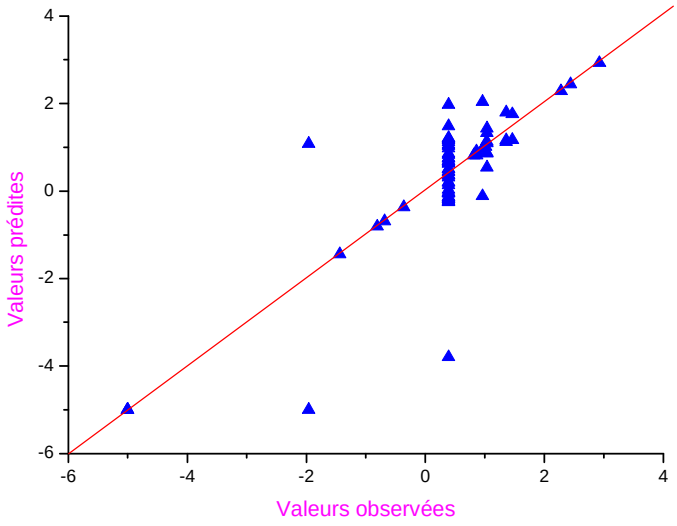
Les graphes (b), (c) et (d) représentent la dispersion du nuage de points sur d'autres plans factoriels, qui est pratiquement similaire dans les plans (b) et (c). Ceci est dû aux proximités des points qui sont maintenues.

B) Régression multilinéaire

Nous avons également effectué une étude de régression multilinéaire sur les facteurs principaux, issus de l'étude de l'ACP sur les vecteurs d'autocorrélation, à partir de l'interface « Properties_Regmul ». Nous rappelons que l'ACP génère un fichier classant les facteurs d'autocorrélation par ordre d'importance décroissante, ce qui facilite la recherche des corrélations en utilisant seulement les premiers facteurs ayant de l'importance. Cette recherche a été faite à l'aide du programme Regmul[(21)] interfacé et retraduit en « Properties_Regmul » qui fonctionne selon des critères fixés par l'utilisateur. Ces critères sont le nombre de facteurs à utiliser pour rechercher ces corrélations, le nombre de variables indépendantes qui sont choisies parmi ces facteurs et le nombre d'itérations à effectuer avec la méthode de Monté Carlo qui choisie les variables de la régression. Comme, d'un point de vue statistique, le nombre de variables doit rester inferieur au cinquième du nombre de molécules, nous avons recherché des régressions à 3, puis 6, ensuite à 9 variables et cela en utilisant les vingt premiers facteurs et 200 itérations. Les coefficients de corrélations obtenus dans les trois cas sont respectivement 0.51511, 0.55371 et 0.67535

Nous avons tenté d'améliorer le coefficient de corrélation en poussant la recherche à vingt variables et en utilisant les 20 premiers facteurs (la diversité des structures moléculaires étudiées justifie le nombre de variables élevé que nous avons pris. Cette dernière est traduite par le nombre de facteurs qui apportent significativement une information). Cette recherche a conduit au coefficient de corrélation égale à 0.71269.

L'exploitation des résultats de cette étude, moyennant OriginPro 7.5, nous a permis de tracer le graphe qui représente les activités prédites en fonction de celles observées autour de la droite idéale.



Fgure 5 : Distribution des activités observées et prédites autour de la droite idéale recueillie du programme Regmul

Sur cette distribution, on constate qu'il y a des points qui s'écartent des autres ; ce résultat est peut être dû à des molécules dont les propriétés sont mal calculées ou la mesure expérimentale mal effectuée. La distribution des points autour de la droite est pratiquement homogène dans les deux côtés.

❖ *Validation du modèle*

Nous avons retiré, tout à fait au hasard, 11 molécules (représentant le $\frac{1}{5}$ du nombre des molécules de la base de données) du fichier contenant les structures moléculaires au format sdf de départ. Nous avons donc formé deux nouveaux fichiers de données, le premier est le fichier d'apprentissage, et le deuxième est le fichier d'essai qui contient les molécules retirées du fichier initial

Afin de valider le modèle obtenu, nous l'avons utilisé sur le lot d'essai (figure 6). Nous avons calculé les vecteurs d'autocorrélation en utilisant les mêmes propriétés que celles utilisées pour le lot d'entrainement. A l'aide du programme Insert2[(19)] (que nous n'avons pas encore interfacé), nous avons remis ces molécules du lot d'essai dans le même repère factoriel que celles du lot d'apprentissage c'est à dire dans la première ACP (figure 6). Ensuite, à l'aide du modèle QSAR obtenu pour le lot d'apprentissage, nous avons prédit les activités des molécules du lot d'essai.

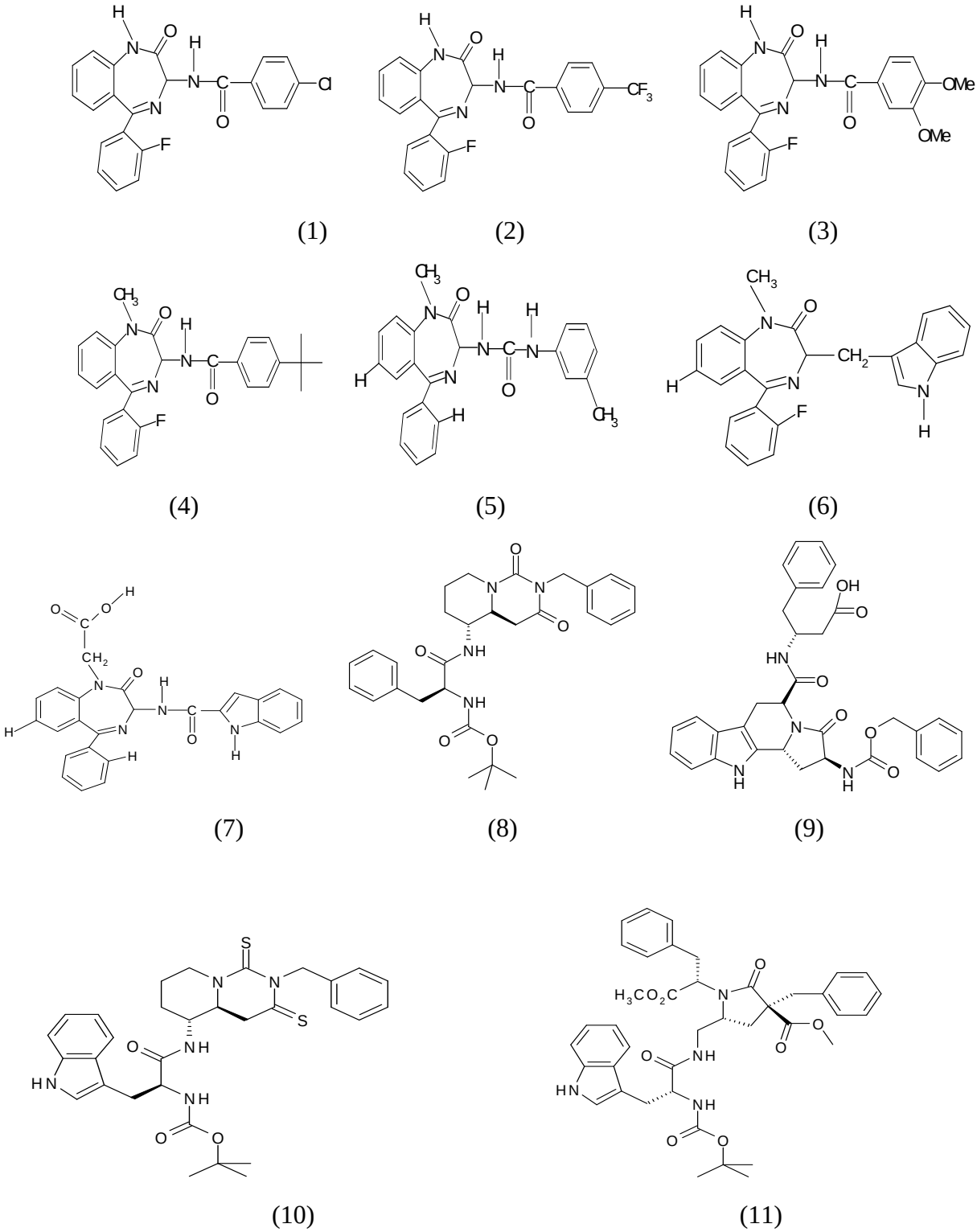


Figure 6 : Structure des molécules écartées du lot d'apprentissage

Nous donnons ci-dessous la représentation des molécules des deux lots de molécules dans le premier plan factoriel. Nous constatons que la distribution relative au lot d'apprentissage (en bleu) est pratiquement semblable à celle du lot d'essai (en rouge).

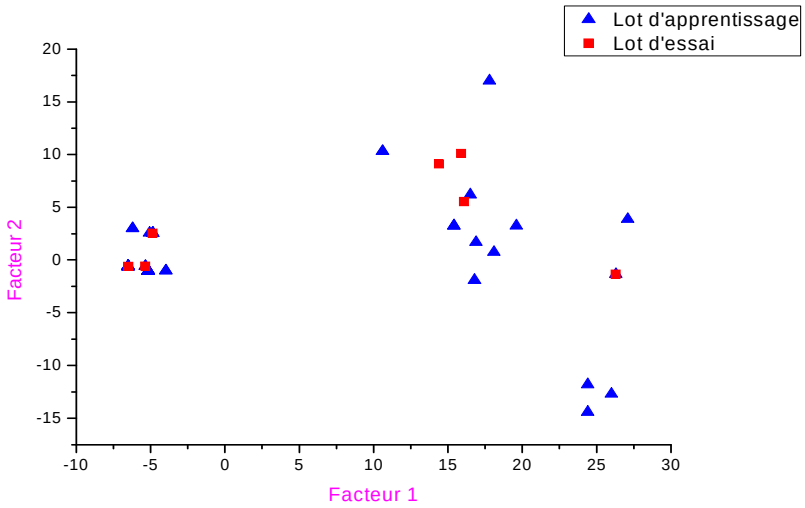


Figure 7 : Représentation des molécules des lots d'apprentissage et d'essai dans le premier plan factoriel.

Nous avons appliqué une régression multilinéaire sur les molécules du lot d'essai en utilisant les mêmes conditions utilisées pour les molécules du lot d'apprentissage, et nous avons recueilli un coefficient de corrélation : R = 0.75136. Nous reportons ci-dessous (figure 8), le graphe représentant la distribution des activités prédites et des activités observées autour de la droite idéale. Nous avons représenté les activités du lot d'essai en rouge, et celles du lot d'apprentissage en bleu.

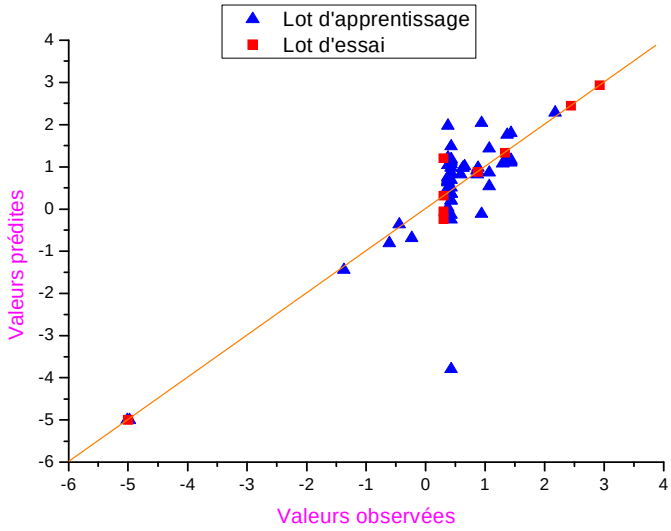


***Figure 8* :** Distribution des activités observées et prédites par le modèle QSAR autour de la droite idéale

C) Analyse discriminante

Nous avons terminé cette application par une analyse discriminante sur les mêmes facteurs principaux. Cette analyse réalisée à l'aide du programme Anadis[(22)] réorganisé en l'interface « Properties_Anadis », qui a pour but de classer qualitativement une molécule d'activité inconnue. Cette technique qui se base sur les descripteurs, qui sont dans notre cas des vecteurs d'autocorrélation, permet de déterminer ceux qui distinguent le mieux les deux classes de molécules.

Le fichier d'apprentissage contient deux classes, selon la valeur de l'activité. La première classe contient les molécules dites actives dont l'activité est incluse dans le domaine [0,035 ; 7,5], et la deuxième contient les molécules dites peu actives dont l'activité est incluse dans le domaine [8 ; 150].

Nous avons donc chargé en entrée le fichier d'apprentissage (*.dat)*. Nous avons ensuite pris les facteurs principaux de 1 à 4, et nous nous sommes fixé un seuil d'activité de 7.5. Enfin, nous avons récupéré le fichier de sortie en extension *.mda*.

Dans ce lot d'apprentissage, les molécules actives sont classées selon un pourcentage de 96.15% de succès, et les molécules peu actives sont classées selon un pourcentage de 43.75% de succès. Ces pourcentages nous donnent une idée de ce que nous pouvons espérer de ce programme. En effet, le test de ce modèle sur le lot d'essai conduit à un pourcentage de 88.9% pour la première classe contenant les molécules actives et à un pourcentage de 60.23% pour la

deuxième classe contenant les molécules peu actives. Nous donnons dans le tableau ci-dessous, le classement des molécules, expérimentalement et après l'application de l'analyse discriminante :

ADM : Analyse Discriminante Multifactorielle

Numéro de la molécule	**Avant ADM**	**Après ADM**
1	1	2
2	2	1
3	1	2
4	1	1
5	2	2
6	2	2
7	1	1
8	2	2
9	1	1
10	1	1
11	1	1

Tableau 2 : Résultats de l'analyse discriminante sur les molécules du lot d'essai
Avant ADM : correspond au classement défini par l'activité réelle
Après ADM : correspond au classement déterminé par l'analyse discriminante.

Nous donnons les résultats de cette analyse discriminante dans le diagramme suivant :

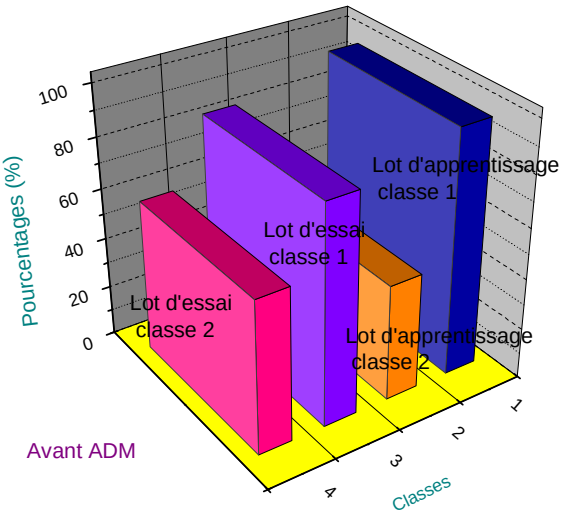


***Figure 9* :** Représentation de la répartition des molécules avant l'ADM

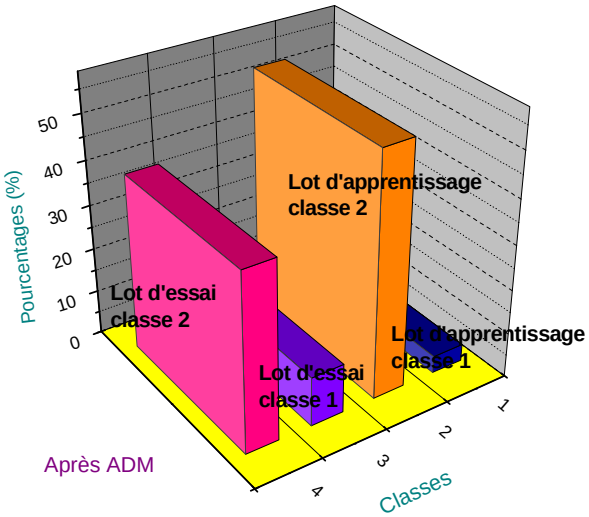


Figure 10 : Représentation de la répartition des molécules après l'ADM

IV. CONCLUSION

Dans ce chapitre nous avons présenté, dans un premier temps, un aperçu de la cholécystokinine (CCK), ses récepteurs et ses rôles physiologiques. Ensuite, nous avons effectué, dans un deuxième temps, une application portant sur l'étude d'autocorrelation 2D sur plusieurs séries de molécules constituant la base de données. Cette base comporte des ligands des récepteurs de la CCK. L'étude a été faite à partir de la maquette comprenant la chaîne de programmes de l'autocorrélation, insérée dans la plateforme VSM-G.

Dans l'étude de l'autocorrélation, nous avons utilisé les techniques de l'analyse en composantes principales qui nous a permis de choisir les meilleurs facteurs, issus des vecteurs d'autocorrélations calculés sur la base de huit propriétés atomiques. Les résultats de l'ACP montrent que c'est le premier facteur qui contient le plus d'informations sur les structures étudiées que les autres facteurs, ce qui conduit à conclure qu'il faut s'intéresser à la représentation sur le premier plan factoriel (plan décrit par les deux premiers facteurs ou axes principaux). Ensuite, nous avons effectué une étude de régression multilinéaire pour trois, six, neuf, puis vingt variables. Les trois premiers cas donnent des coefficients de corrélation moyens mais concluants. Le dernier conduit à un modèle statistiquement acceptable sur le plan prédiction puisque le coefficient de corrélation obtenu est R = 0.71269. L'application de ce modèle QSAR à un lot d'essai de 11 molécules, fournit des activités qualitativement bonnes.

Ensuite nous avons fini notre application par effectuer une analyse discriminante, qui nous a permis de classer les molécules en deux classes : actives et peu actives. Cette technique prédit qualitativement l'activité d'une molécule selon qu'elle appartienne à l'une des deux classes des molécules étudiées. Les résultats issus du lot d'apprentissage montrent un meilleur classement pour les molécules actives et peu actives puisqu'il a attribué un pourcentage de succès élevé pour les molécules actives. Les résultats obtenus pour le lot d'essai vont dans le même sens que précédemment. .

Les modèles obtenus au cours de ces études permettent de prédire aussi bien qualitativement que quantitativement l'activité d'une nouvelle molécule. En effet, ces modèles peuvent apporter une amélioration au processus de criblage virtuel, et une accélération de ce dernier en faisant des pré-filtrages de molécules.

REFERENCES BIBLIOGRAPHIQUES

1- Taïri-Kellou, S., Thèse de Doctorat d'état, Conception Assistée par Ordinateur des Complexes ligand/récepteur de la Cholécystokinine, Alger, USTHB, N° d'ordre : 04/**2002**-E/CH.

2- Langer, I., Tikhonova, I. G., Travers, M. A., Archer-Lahlou, E., Escrieut, C., Maigret, B. et Fourmy, D., JBC, 280, 22198-22204, **2005.**

3- Gardner, J. D., Conlon, T. P., Kleveman, H. L., Adams, T. D. et Ondetti, M. A., J. Clin. Invest., 56, 366-375, **1975.**

4- Noble, F., Wank, S. A., Crawley, J. N., Bradwejn, J., Seroogy, K. B., Hamon, M. et Roques, B. P., International union of pharmacology. XXI. Structure, distribution, and functions of cholécystokinine receptors. Pharmacological Reviews, 51, Issue 4, 745-781, **1999**.

5- Rehfeld, J. F., Cholecystokinin, Best Practice & Research, 18, 569-586, **2004**.

6- Faris, P. L., Komisaruk, B. R., Watkins, L. R. et Meyer, D. J., Evidence for the neuropeptide cholecystokinin as an antagonist of opiate analgesia. Science, 219, 310-312, **1983**.

7- Agnes, R. S., Lee, Y. S., Davis, P., Ma, S., Badghisi, H., Porreca, F., Lai, J. et Hruby, V. J., Structure-Activity Relationships of Bifunctional Peptides Based on Overlapping Pharmacophores at Opioid and Cholecystokinin Receptors. J. Med. Chem., *49*, 2868 – 2875, **2006.**

8- Tachikawa, H., Harada, S., Kawanissi, Y., Okubo, T. et Shiraishi, H., Novel polymorphism in the promoter and coding regions of the human cholecystokinin B receptor gene: An association analysis with schizophrenia. Am J Med Genet, 88, 700–704, **1999**.

9- Lemaire, M., Piot, O., Roques, B. P., Böhme, G. A. et Blanchard, J. C., Neuroreport., 3, 925-932, **1992.**

10- G. Moreau, P. Broto, The Autocorrelation of a Topological Structure: a new Molecular Descriptor. Nouv. J. Chim. 4, 359, **1980**.

11- G. Moreau, P. Broto, Autocorrélation of Molecular Structures, Application to SAR Studies. Nouv. J. Chim. 4, 757, **1980**.

12- Huché, M., Legendre, J. J; Three Dimensional Molecular shape Analysis Quantitative Structure–Activity Relationship: Activity of a Series of Cholecystokinine-A Receptor

Antagonists; J.Med.Chem, 37, 3639-3654, **1994.**

13- Sinha, J., Kurup, A., Paleti, A. and Gupta, S.P; Quantitative Structure- Activity Relationship Study on Some Nonpeptidal Cholecystokinin Antagonists Bioorganic & Midicinal Chemistry, 7, 1127-1130, **1999.**

14- Martin-Martinez, M., et al, J.Med.Chem, 43, 3770-3777, **2000**

15- Bartolomé-Nebreda, J.M., et al, J Med. Chem 42, 4659-4668, **1999**

16- Bartolomé-Nebreda, J.M., et al, J Med. Chem 44, 4196-4206, **2001**

17- Martin-Martinez, M., et al, Chem. Pharm. Bull, 46, 782-786, **1998**

18- ISIS™/Draw 2.3., © Copyright, **1990- 2000;** MDL Information Systems, Inc

19- programmes écrits par Gilles Moreau utilisant la librairie mathématique LAPACK.

20- OriginPro 7.5 SRO, © Copyright, **1991-2003;** OriginLab Corporation.

21- C'est une adaptation du programme régression de Legendre

22- C'est une adaptation du programme mda par Murtagh trouvé sur le web : **http://astro.ustrasbg. r/~fmurtagh/mda-sw/**

CONCLUSION GENERALE ET PERSPECTIVES

Le criblage moléculaire *in silico* à haut débit s'avère comme une méthode utile voire incontournable pour qui veut éviter de procéder à de trop nombreux essais, coûteux, par la technique du criblage à haut débit *in vitro*, et cela tout en explorant une diversité moléculaire plus importante. En effet, le criblage virtuel permet d'accélérer et de rationaliser la recherche de molécules innovantes, têtes de série pour la recherche de molécules actives sur des cibles biologiques importantes en thérapeutique. Il faut noter que la stratégie consistant à effectuer un criblage préliminaire *in silico* de banques de données moléculaires fait actuellement l'objet d'un fort développement.

Les travaux présentés dans ce manuscrit consistent en l'implémentation, en amont du criblage, des techniques statistiques QSAR dans la plateforme de screening virtuel VSM-G et ce dans le but d'améliorer l'efficacité du criblage, basé sur le docking/scoring. Les modèles QSAR obtenus sont capables de trier, sur des critères topologiques, la chimiothèque de départ et ainsi la répartition en vrais positifs et en vrais négatifs est améliorée.

Dans le premier chapitre nous avons donné une description des méthodes du Drug Design en particulier celle du QSAR. Nous rappelons que le drug design comprend deux approches, celle du « Receptor based » reposant sur les études de docking moléculaire, et celle du « Ligand based » reposant sur les études de relation entre la structure et l'activité d'une molécule et ses propriétés structurales. L'approche « Ligand based » prend le plus souvent l'appelation QSAR (Quantitative Structure-Activity Relationships). Nous avons également donnée une description de la plateforme de screening VSM-G dans laquelle nous avons inséré les outils statistiques QSAR, utilisant des descripteurs issus de la methode d'autocorrelation.

Dans le deuxième chapitre nous avons présenté le langage Java, langage d'écriture de la plateforme logicielle VSM-G. Nous avons ensuite présenté la maquette développée comportant la chaîne de programmes de l'autocorrelation représentant les outils statistiques QSAR. Ces outils permettent d'accélérer les processus du criblage et d'améliorer son efficacité. Nous avons également validé cette implémentation sur une base de données contenant les ligands des récepteurs LXR.

Dans le troisième chapitre, nous avons donné un aperçu de la cholécystokinine (CCK), une hormone neuropeptidique impliquée dans un grand nombre d'actions physiologiques et physiopathologiques. Nous avons ensuite fait une application, à travers VSM-G, sur une chimiothèque comportant les ligands des récepteurs de la CCK.

En conclusion, VSM-G comprend désormais les outils QSAR-2D. Avec cette plateforme, il est possible d'effectuer des études utilisant les deux approches « structure based » et « ligand based ». L'approche « ligand based », reposant sur le QSAR, permet de faire un pré-filtrage de la chimiothèque sur la base de modèles prédictifs élaborés sur une base de molécules dont l'activité d'intérêt est connue. Par ces modèles permettant d'améliorer la répartition en vrais positifs et en vrais négatifs, le criblage sera plus efficace.

En perspective de nos travaux et dans l'espoir de faire du produit VSM-G un outil plus performant, voire plus général et riche en techniques de criblage, des modules « Extra » sont en attente d'être implémentés. Nous citons « SVM » : Support Vector Machine ou *séparateurs à vaste marge*, qui sont un ensemble de techniques d'apprentissage supervisées et utilisées dans le cadre de la résolution des problèmes de discrimination des bases de données de grandes dimensions. Les vecteurs machines SVM sont considérés comme des classificateurs qui permettent de classer dans des groupes des échantillons qui ont des propriétés similaires. Ces classificateurs ont la capacité de travailler avec des données de grandes dimensions. Nous citons aussi « Pharmacophore Search », qui est une technique améliorant le screening virtuel et basée sur la recherche de pharmacophores « di » ou « tri » dimensionnels dans une base de données. Cette technique est utilisée dans l'approche « Structure based », consistant à rechercher des modèles de pharmacophores topologiques. L'utilisation des informations liées à la structure de ces pharmacophores évite d'employer les détails des calculs du docking moléculaire.

ANNEXES

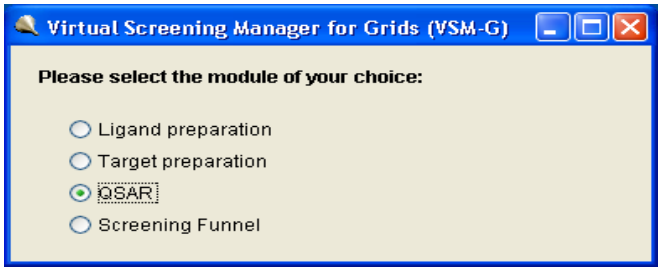
AUTOCORRELATION INTERFACE

User Guide

Contents :

I. How to start:

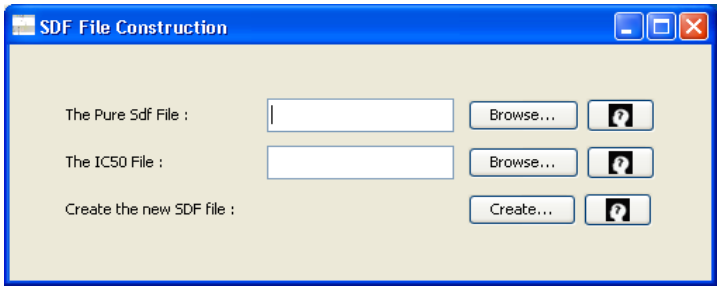
To begin QSAR calculations through VSM-G, we launch first the screening plateform program. Its main window is:



Only a click on the « QSAR » button is enough to access the "Programs" interface, which includes the different QSAR methods (Autocorrelation and all its programs, SVM (for the Support Vector Machine and Pharmacophore Search (to search for pharmacophores), which are not operational yet), and the program that builds the SDF file, needed for subsequent calculations. It is summarized as follows:

❖ Construction of the SDF file :

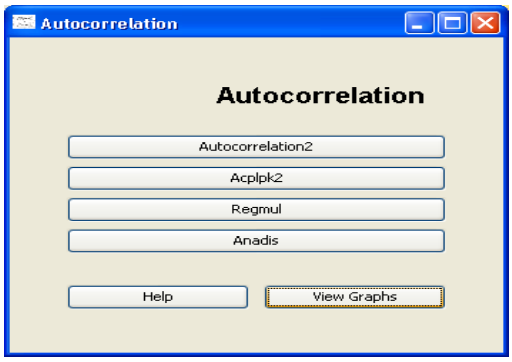
The « SDF File Construction » is designed to create the file that contains all the molecular structures that constitutes the database, and their activity values. By pressing the « SDF File Construction » button:



you can see that the interface contains two text fields: "The pure sdf File" to load the sdf file, built without the values of activities, "The IC 50 File" to load the file that contains all the activity values of the molecules data (in the same order), and the "Create" button, to create the final sdf file which assembles the structures and the molecules activities. This file will be used to generate autocorrelation vectors.

II. The Autocorrelation interface:

The "Autocorrelation" interface contains the executables: Autocorrelation2 which generates autocorrelation vectors, Regmul for multilinear regression, Acplpk2 for the principal components analysis and Anadis for the discriminant analysis.

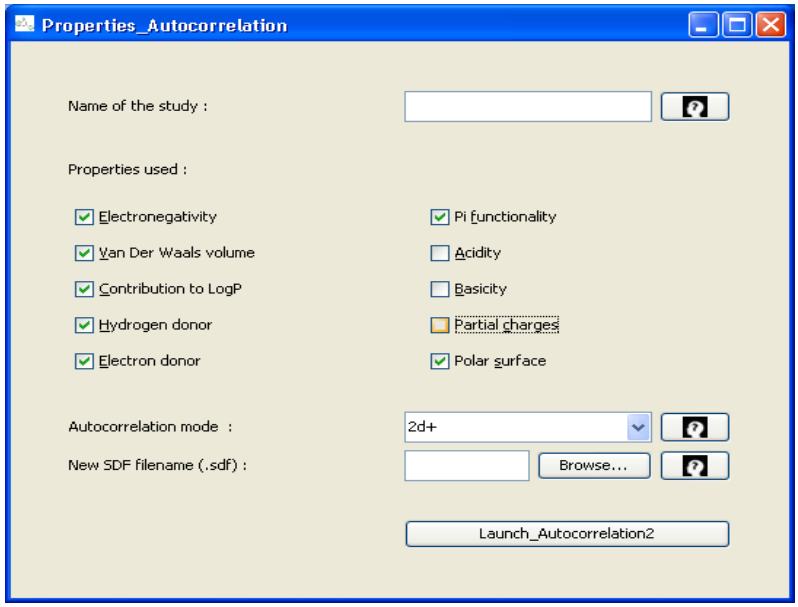


Each one of these programs is reflected on the Graphical User Interface (GUI) by buttons, each one opens its specific window. We also found on the same window two other buttons: the "Help" which is a small manual which gathers the technical tools mentioned above, and informations about all the interfaces, and the "View Graphs" which serves to draw graphs to exploit the results of the statistical studies.

II. 1 Autocorrelation2 interface:

The autocorrelation2 executable is translated in a very simple graphical user interface easy to use, it is named "Properties_Autocorrelation2", and it aims to facilitate access to each question raised by this program.

The graphical user interface includes several questions presented as text fields that should be filled out in order, respecting the specificity of each question. This interface includes the following items:



1- « Name of the study»

The name of the study will be the filename generated by autocorrelation2 program.

2- « Properties used »

In this part, it is not necessary to answer because the properties usually the most useful are already checked, but free the user to disable or check other properties.

3- « Autocorrelation mode »

The combo box contains 2D, 2D+ and 3D modes.

2D and 2D+ use 2D structure and can therefore be used also with 3D structures (since 2D structure is contained).

2D+ is recommended for structures of varying sizes.

3D should be used only if the structures are 3D and if you really want to use the three-dimensional shape.

It should be noted that the mode is already checked, but it is free to the user to change it.

4- « Molecules filename (.SDF) »

Load the new molecular structures file, with SDF format.

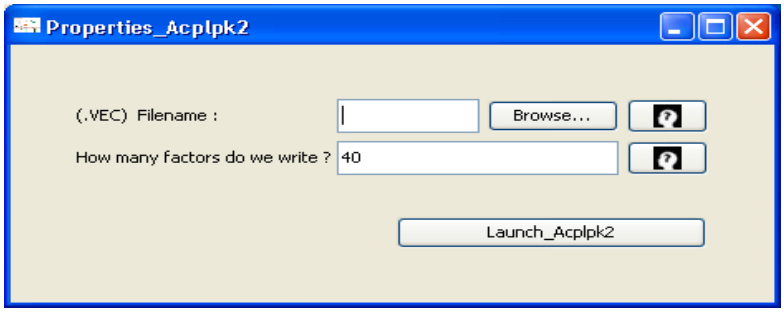
5- « Launch_Autocorrelation2 »

The calculation can be done only by a press on this button. The windox closes, the result file will be created where you have already registered your sdf file, and it will bear the name of the study followed by the extension .VEC.

II. 2 Acplpk2 interface:

The Acplpk2 executable is reflected by a very simple graphical user interface easy to use, it is named "Properties_Acplpk2", and it aims to facilitate access to each question raised by this program.

In the same way as the previous graphical user interface, this one also follows the same look, it contains three questions gathered in some text fields:



1- « (.VEC) Filename »

Load the .Vec file, which includes autocorrelation vectors, resulting from the Autocorrelation2 calculation.

2- « Haw many factors do we write? »

These are the coordinates factorials obtained from the .VEC file.

The program can accept up to 40 primary factors, therefore answer 40.

3- « Launch_Acplpk2 »

Only a press on this button can do the calculation. The window closes, and the result will be created where you have already registered your .vec file.

This result is two files monetude.acp and monetude.dat. The .acp file contains details of the work and results produced by ACPLPK2 (components eliminated if there are not, information percentages on each dimension, values and

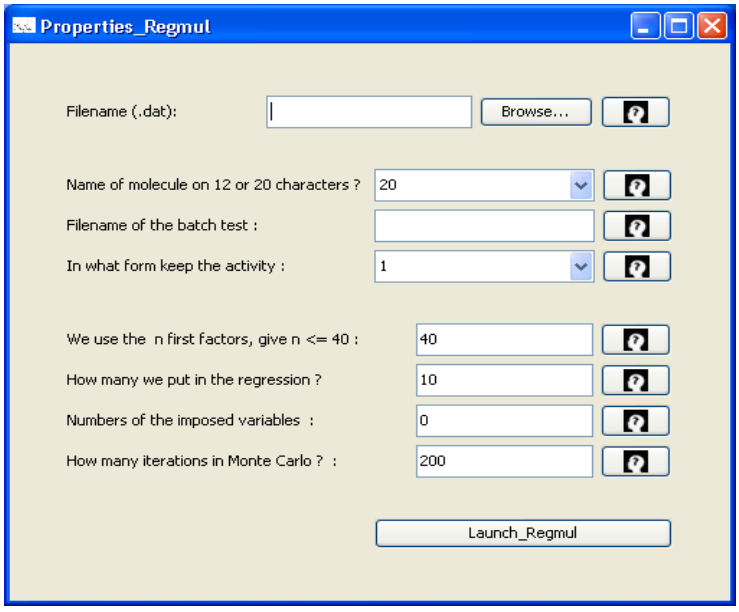
eigenvectors…), the .dat file contains as many lines as molecules on which are written their coordinates factorials and the activity.

II. 3 Regmul interface:

The Regmul executable is reflected by a very simple graphical user interface easy to use, it is named "Properties_Regmul", and it aims to facilitate access to each question raised by this program.

The programme will work on a “monetude.dat” file type, but generally if we have enough molecules it is good to withdraw from the monetude.dat file, a number of molecules (of lines), chosen at random, which will constitute a new file with the same type, we will have two files “monetude1.dat” and “monetude2.dat”, for example monetude1.dat will contain 200 molecules and monetude2 will contain 50. The first file monetude1.dat is sometimes called the learning file because the program needs it to establish the Multilinear regression. Then the program uses this learning file to predict the activity class, of the monetude2.dat file molecules: the comparison of predicted and real activities can conclude on the quality of regression. The monetude2.dat file is sometimes called test file, but it can also be a file of new structures, for which we want to make predictions.

In the same way as the previous graphical user interface, this one also follows the same look, it contains many questions gathered in some text fields:



1- « (.dat) filename »

Load the .dat file that includes the X coordinates Factors (or the main factors), obtained from the ACP study on the autocorrelation vectors. It will serve to establish a "model", a regression, therefore choose monetude1.dat.

2- « Name of molecule on 12 or 20 characters ? »

The combo box contains two values, it is free to the user to choose any answer. This is the length in number of characters of the field were we have put the name of the molecule, this field can enter up to 12 or up to 20 characters.

3- « Filename of the batch test »

The batch test is used to measure the quality of the model or to validate it. This
If the batch test file exists, for example monetude2.dat, then its name must be put followed by the .Dat extension
If the batch test file does not exist, then the words (vide.dat) must be put in the text field.

4- « In what form keep the activity »

The combo box contains three possible choices to answer for the activity transformation:
Answer 1: If you want to transform the activity to log(activity).
Answer 2: If you want to transform the activity to 10*tanh(activity/activity0)
Answer 3: If you want to transform the activity to arctang(activity/activity0)
activity0 is the reference activity
It is worth mentioning that the best calculation is given by the value already registered.

5- « We use the n first factors, give n <= 40 »

The Acplpk2 program generates a file that displays the main factors in order of importance of their own decreasing values, therefore it must be given the number n of the first main factors, which will be chosen among them the p factors, that goes into regression.

The program can take up to the value 40.

6- « How many we put in the regression? »

It is the p number of factors actually present in the regression, this number must be less than n. The number p should preferably be less than the number of molecules divided by 5.

7- « Numbers of the imposed variables »

You can impose that factors must be present in the regression (in the linear form) researched for.

These numbers must be > = 2 and less than n. If there are not, the user has to insert '0'.

8- « Ho many iterations in Monte Carlo »

The search of the best games of p factors chosen from the n ones is done using an Monte-Carlo algorithm: an initial set is tested, it leds to a regression, whose quality is characterized by the correlation coefficient between calculated activities and experimental activities and the statistical factor F; this game is modified by the exit of a factor and the entry of another; the new regression is determined and compared to the previous one; depending on whether or not improvement or a Monte-Carlo logic is applied. It retains the best regression obtained after a number of iterations.

Note that the program can do up to 10000 iterations.

9- « Launch_Regmul »

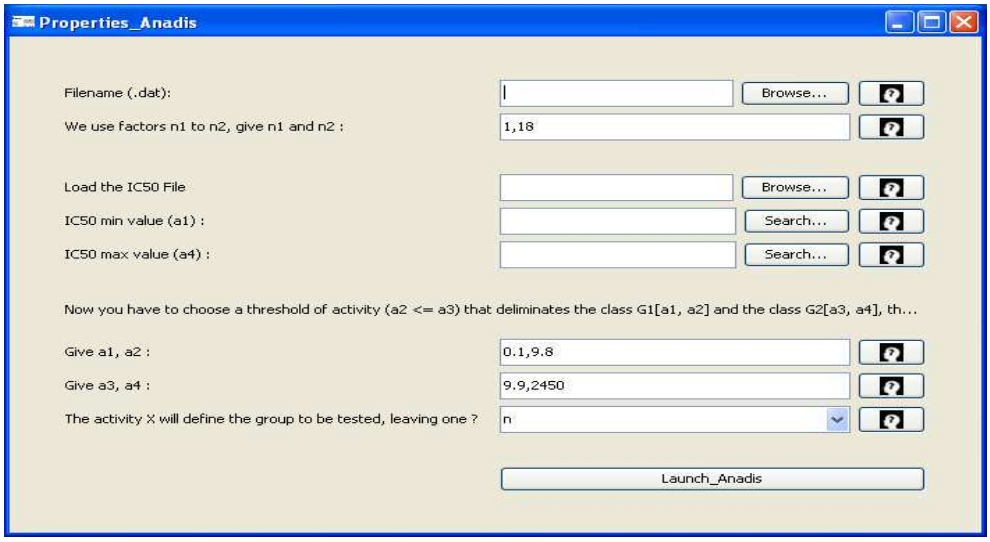
This button press can do the calculation. The window closes, and the result of this study will be created where you have already registered your .dat file. It is the files named XXvide.dat and batchTestfileName.dat where XX is a hexadecimal number from 10 to FF.

XXvide.dat corresponds to the file that contains the results of the regression study batchTestfileName.dat is generated and takes the name which is set for the batch test file. If it does not exist, it will take the name vide.dat (3rd answer on the GUI).

II. 4 Anadis interface:

The Anadis executable is also reflected by a very simple graphical user interface easy to use, it is named "Properties_Anadis", and it aims to facilitate access to each question raised by this program.

In the same way as the previous graphical user interface this one also follows the same look, it contains many questions gathered in some text fields:



1- « Find the .Dat file format »

Load the monetude1.dat file, that includes the X coordinates Factors (or the main factors) obtained from the ACP study on the autocorrelation vectors.

2- « We use factors n1 to n2, give n1 and n2 »

We will use the factors numbers n1 to n2 gathered in an interval

n1 corresponds to the lower bound of the interval.

n2 corresponds to the upper bound of the interval.

Making several tests is recommended.

3- « Load the IC50 file »

Load the IC50 file, which includes the Activities values.

4- « IC50 min value (a1) »

By pressing on the “search” button, the program will search for the minimum

value of activity.

Note that min = a1.

5- « IC50 max value (a4) »

By pressing on the "search" button, the program will search for the maximum value of activity.

Note that max = a4.

→ Now you have to choose a threshold of activity (a2 <= a3) that deliminates the class G1[a1, a2] and the class G2[a3, a4], then:

6- « Give a1, a2»

The sdf file have to be shared into two classes (or two goupes), G1 and G2 according to their activity values.

G1 contains the most active molecules, their activities are contained in an interval.

a1 corresponds to the lower bound of the interval.

a2 corresponds to the upper bound of the interval.

ie: This help us to take all the molecules with an activity noted Y which lies between the activity a1 and the activity a2

7- « Give a3, a4 »

The sdf file have to be shared into two classes (or two goupes), G1 and G2 according to their activity values.

G2 contains the most active molecules, their activities are contained in an interval.

a3 corresponds to the lower bound of the interval.

a4 corresponds to the upper bound of the interval.

ie: This help us to take all the molecules with activity noted Z which lies between the activity a3 and the activity.a4

8- «The activity X will define the group to be tested, leaving one ? »

The combo box containes two possible answers:

«**o**» for “yes”, if we want to have percentages prediction of the two classes of molecules, and whether they are active or very active.

«**n**» for “no”, if we want to have the classification of all the molecules of the data basis molecules.

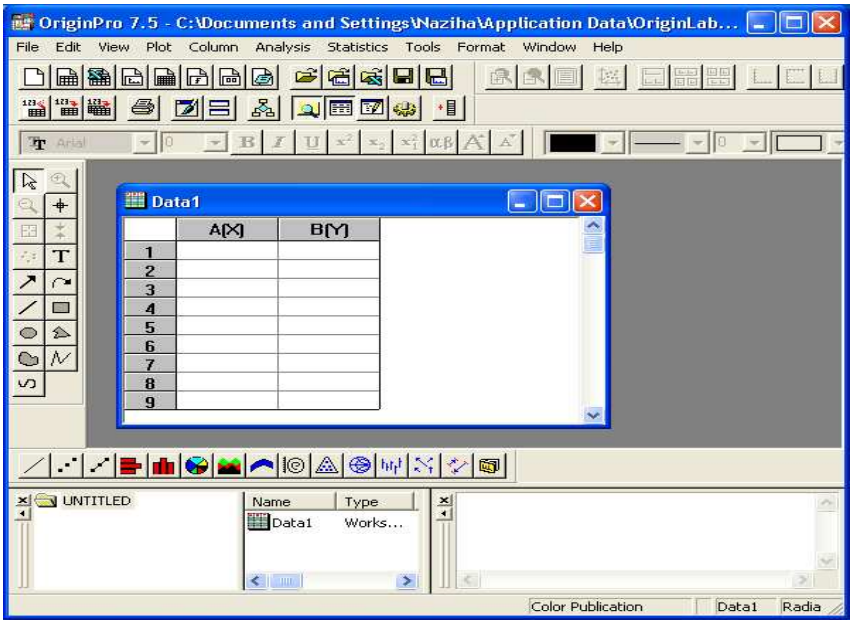
9- « Launch_Anadis »

This button press can do the calculation. The window closes, and the result of this study will be created where you have already registered your .dat file. It is a file wich bears the same filename as the one already put for the .dat file, with the “.mda” extension.

III. Visualisation of results:

III.1 Through OriginPro 7.5:

The software "Origin" is a very sophisticated and pointed, it is used to track any graph, and it is simple, comprehensive and easy to operate with it. To use it from the GUI (graphique user interface), you must first install it on the computer.



To make the cloud of points drawing with **originPro 7.5**, some instructions have to be followed:

1- Press the "ViewGraphs" button on the GUI.
   To open the central interface of originPro 7.5 software.

2- Click on « File » → « Import… » → « Simple Single ASCII » → Load the .dat file resulting from the Acp study → Its content will automatically be pasted in the main table on the originePro 7.5 interface.
3- Select any two columns on the table, then click on « Plot » → « Scatter » to choose the style, with which the graph would be drawn → The cloud of points emerges and is easily interpretable.

III.2 Through Microsoft Excel :

For users who do not have the software « OriginePro 7.5 », they can very easily see their results through the « Microsoft Excel » program,
Follow this way to proceed, to have a good graph drawing:

1- Launch « Excel » program.

2- Load the .dat file resulting from the Acp study (ie : click on File → Open… → Search for the xxxx.dat file on your computer → Choose the fixed width → Click on: **Next** → Put the delimiters if there are not → Click on **Next** then on **Finish**).

3- Once the file is displayed on the Excel window, do not forget to change all the **points** by **commas**, like this: Select the whole table → Click on **Edition** → go on **Replace…** → Put the character « **.** » in the first text field, and put the character « , » in the second one → Press on **OK** then **Close** buttons.

4- Select any two columns on the table.

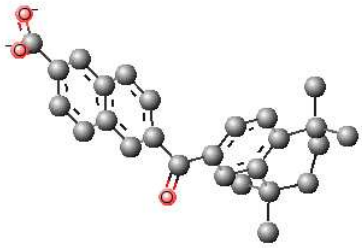
5- In the menu bar click on the icon with a bar named « Graphic Assistant », if not, open it by clicking on: **Insert** → **Graphic…**

6- Choose the « cloud of points » (on the left of the window).

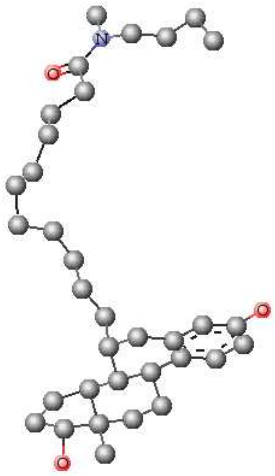
7- Click on **Next** then on **Finish.**

8- Finally, the cloud of points is drawn, and is easily interpretable.

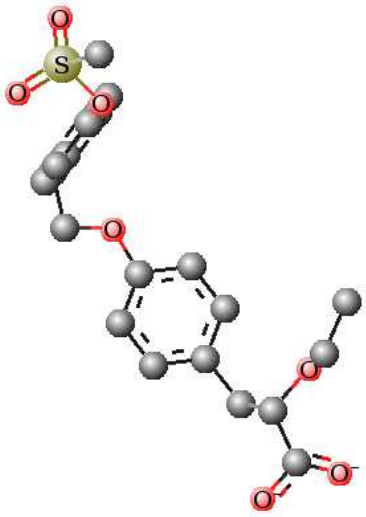
Les Molécules contenues dans la base LXR utilisée dans le chapitre II :



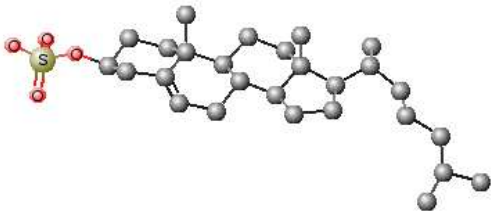
IC50 = 0.2 nM



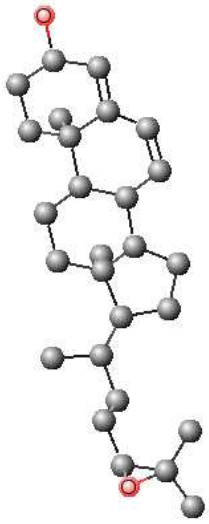
IC50 = 0.008 nM



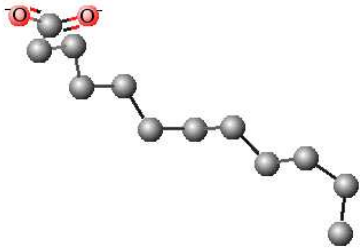
IC50 = 0.002 nM



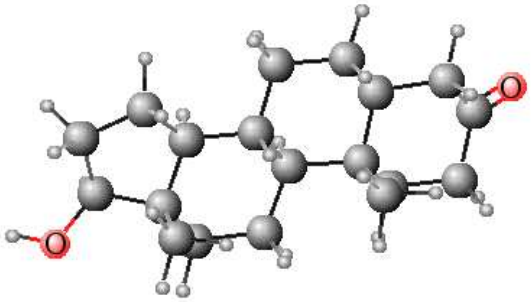
IC50 = 0.002 nM



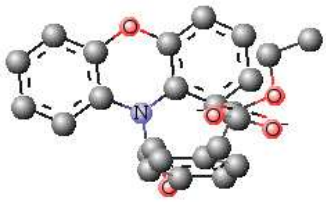
IC50 = 0.001 nM



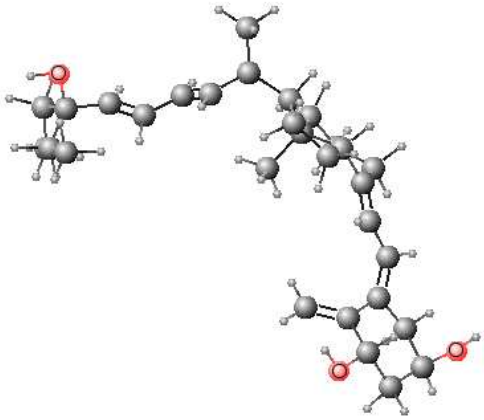
IC50 = 0.014 nM



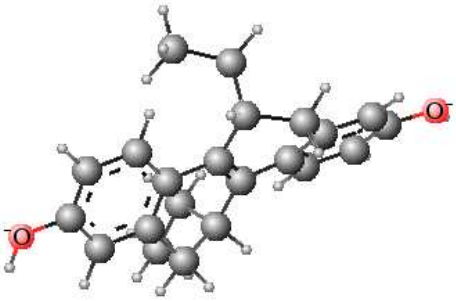
IC50 = 0.01 nM



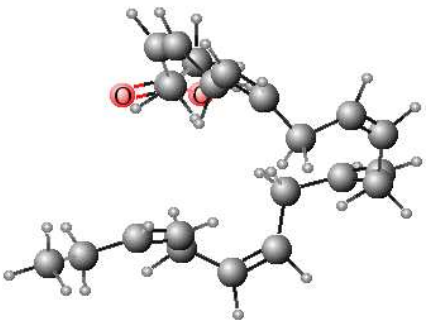
IC50 = 0.2 nM



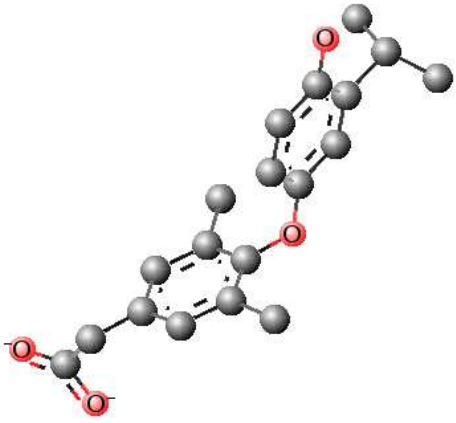
IC50 = 1 nM



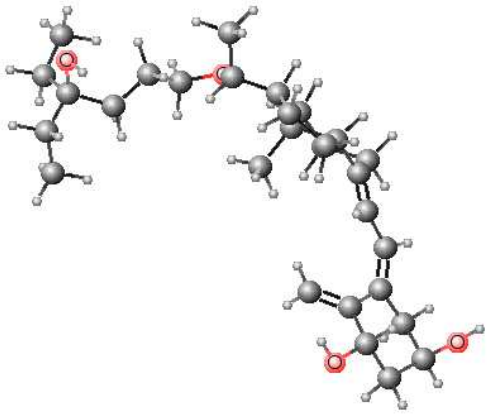
IC50 = 0.1 nM



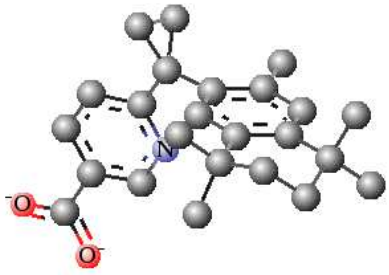
IC50 = 1.1 nM



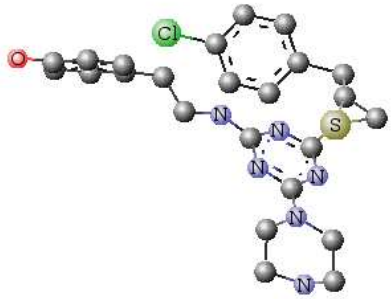
IC50 = 0.2 nM



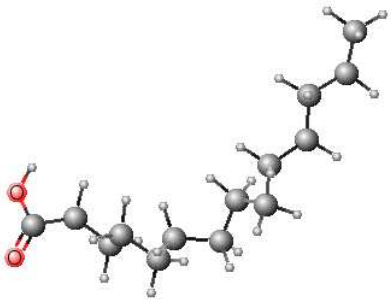
IC50 = 0.2 nM



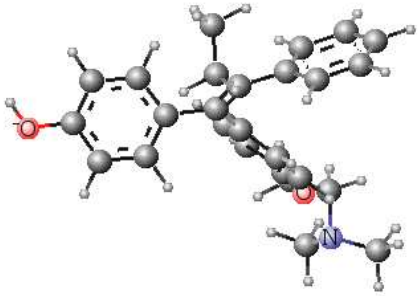
IC50 = 0.1 nM



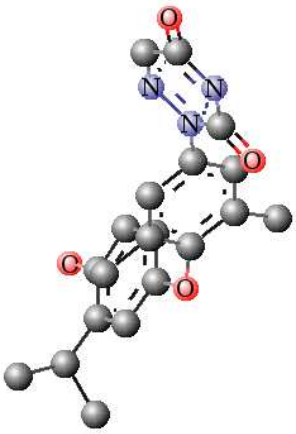
IC50 = 0.7 nM



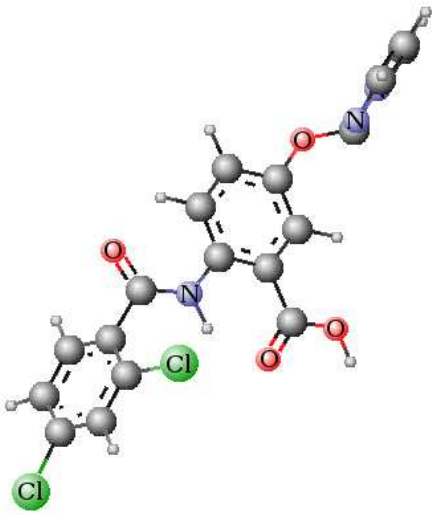
IC50 = 0.26 nM



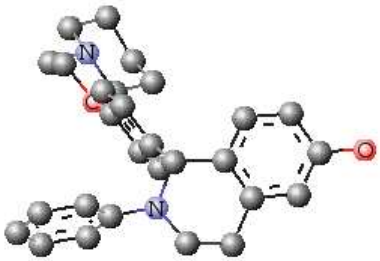
IC50 = 0.26 nM



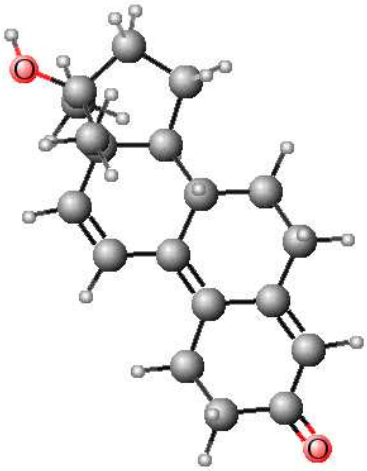
IC50 = 0.25 nM



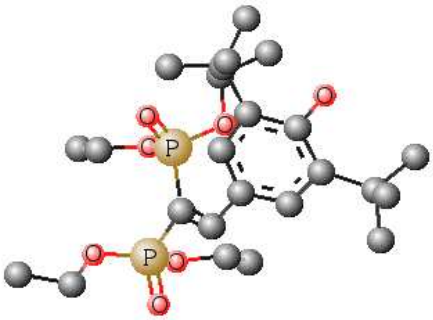
IC50 = 0.19 nM



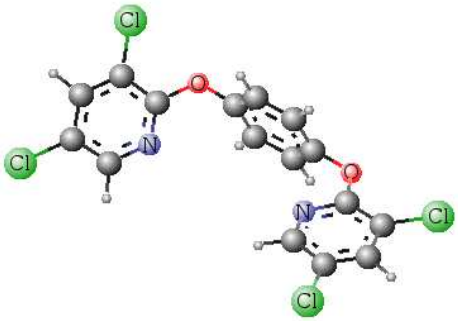
IC50 = 0.085 nM



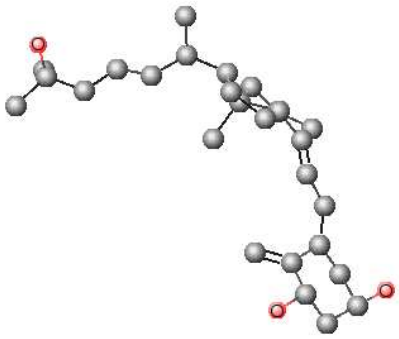
IC50 = 0.085 nM



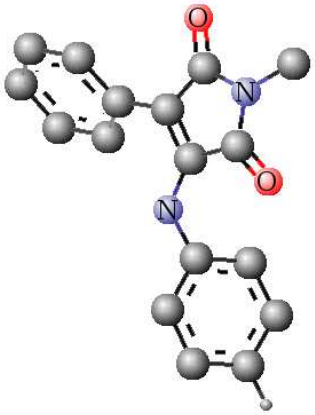
IC50 = 0.045 nM



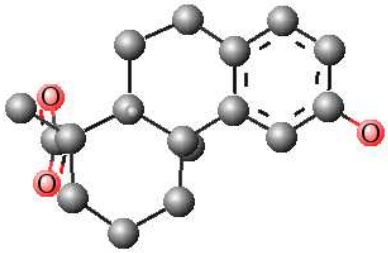
IC50 = 0.125 nM



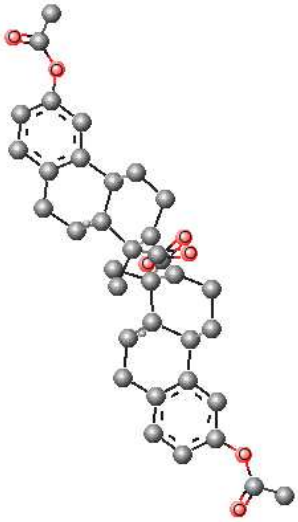
IC50 = 0.45 nM



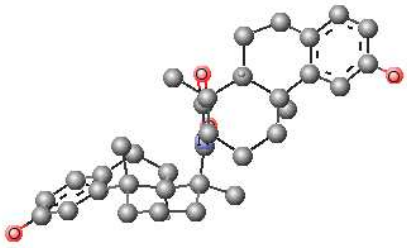
IC50 = 1 nM



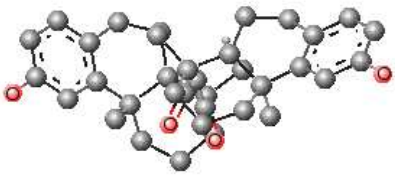
IC50 = 0.17 nM



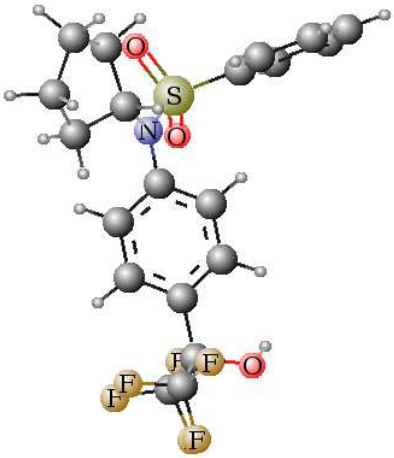
IC50 = 0.59 nM



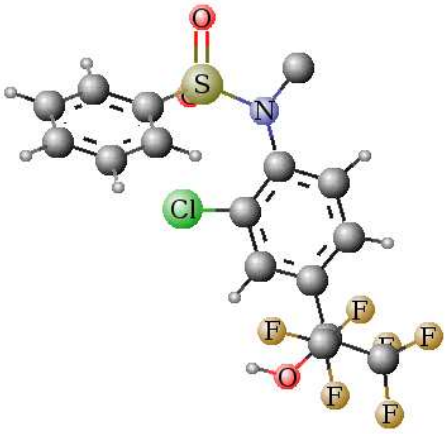
IC50 = 0.48 nM



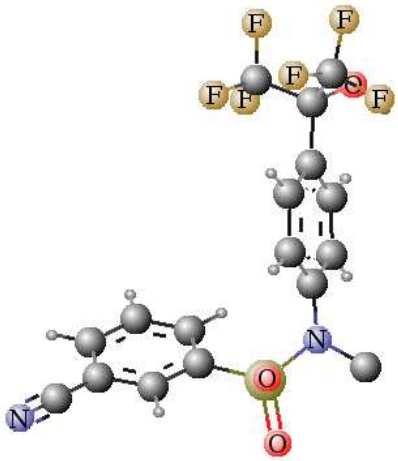
IC50 = 0.18 nM



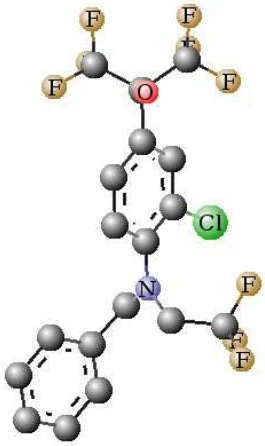
IC50 = 0.186 nM



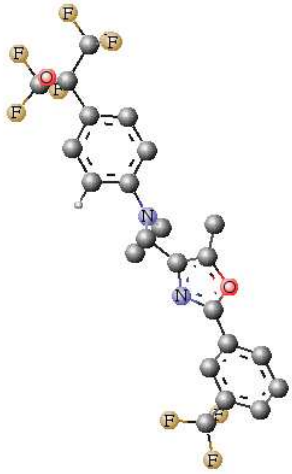
IC50 = 0.143 nM



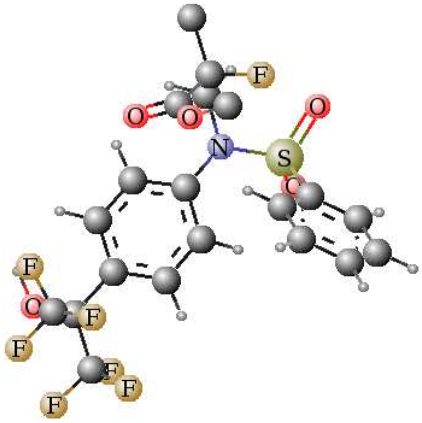
IC50 = 0.071 nM



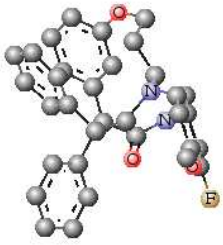
IC50 = 0.033 nM



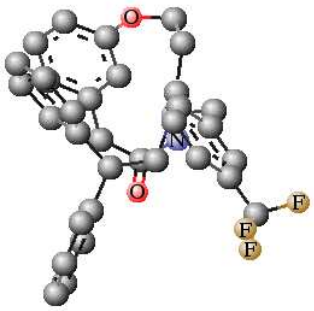
IC50 = 0,047 nM



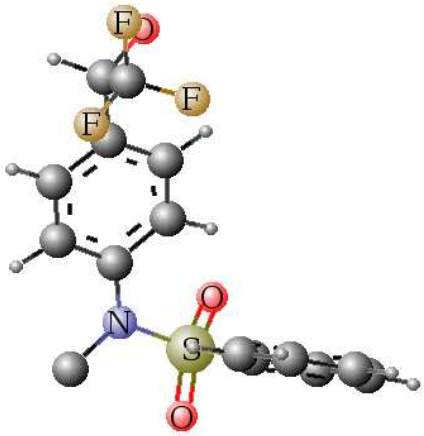
IC50 = 0.374 nM



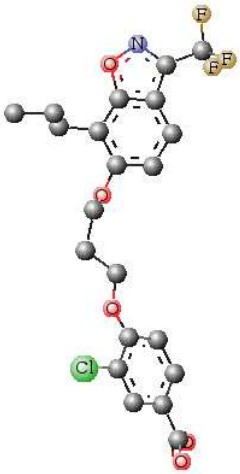
IC50 = 0.6 nM



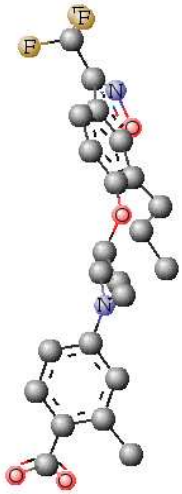
IC50 = 0.05 nM



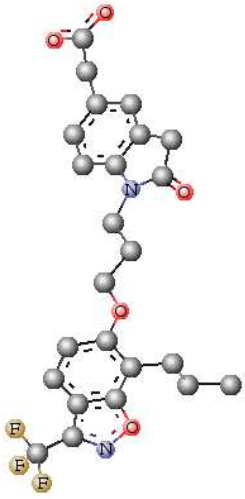
IC50 = 0.275 nM



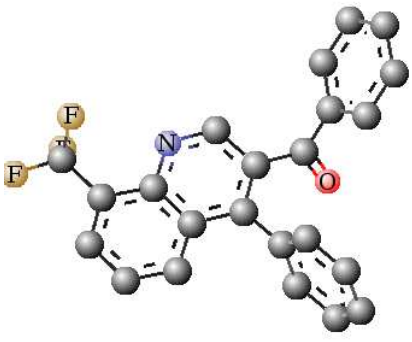
IC50 = 0.3 nM



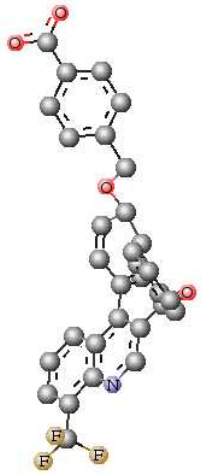
IC50 = 0.575 nM



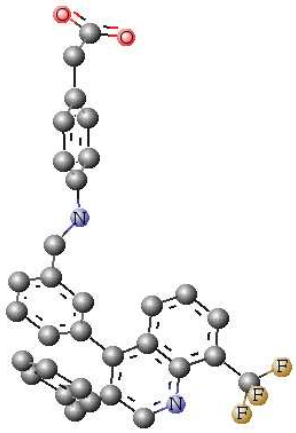
IC50 = 0.1 nM



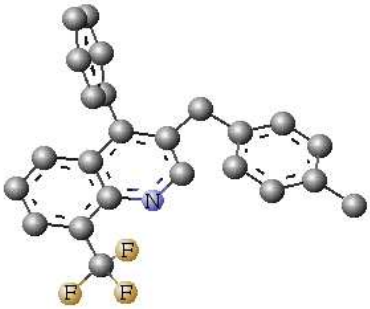
IC50 = 0.13 nM



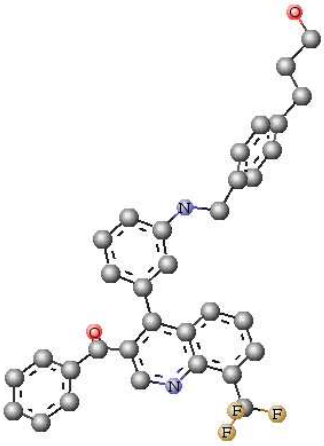
IC50 = 0.89 nM



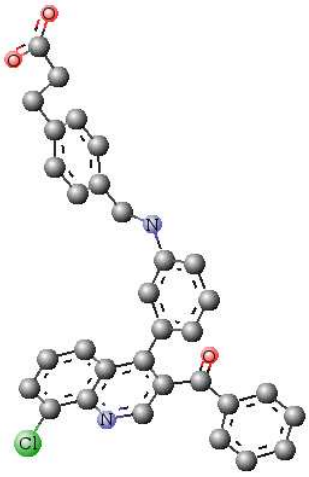
IC50 = 0.725 nM



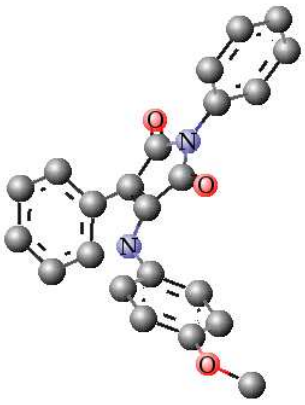
IC50 = 0.2 nM



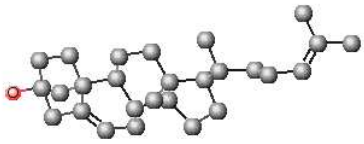
IC50 = 0.380 nM



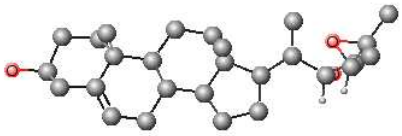
IC50 = 0.15 nM



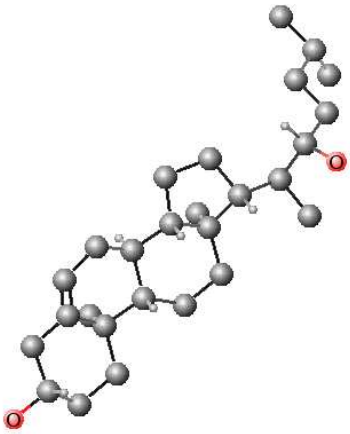
IC50 = 0.95 nM



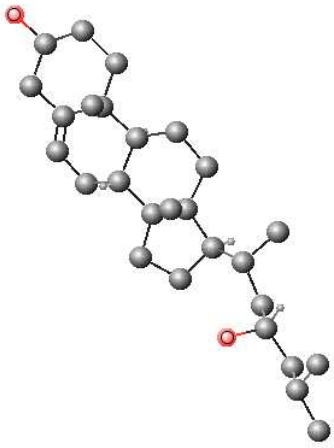
IC50 = 0.18 nM



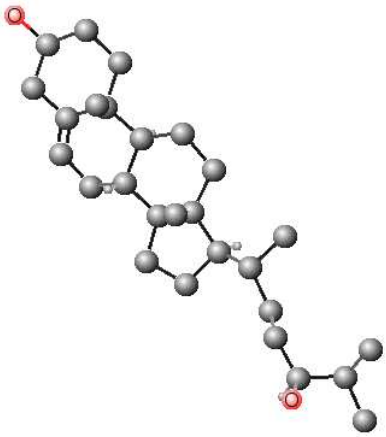
IC50 = 0.44 nM



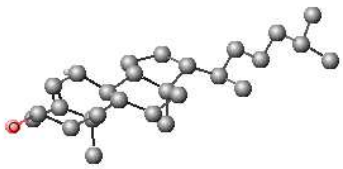
IC50 = 0.99 nM



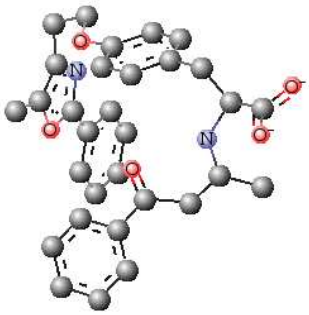
IC50 = 0.11 nM



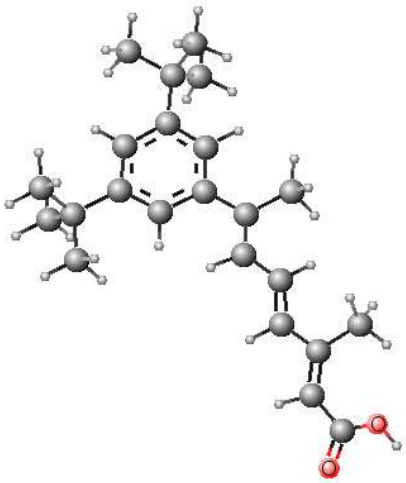
IC50 = 0.13 nM



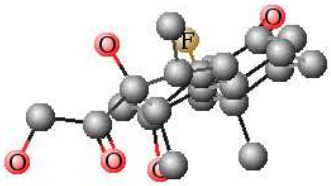
IC50 = 0.11 nM



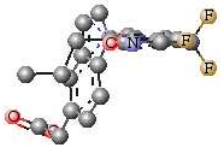
IC50 = 0.46 nM



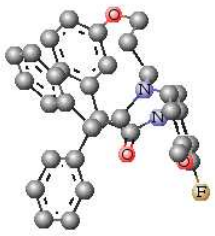
IC50 = 0.25 nM



IC50 = 0.18 nM

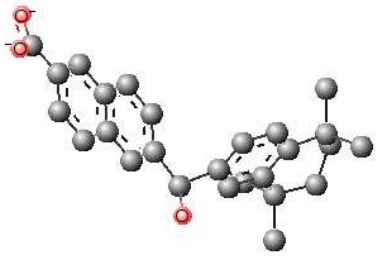


IC50 = 0.17 nM

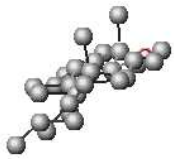


IC50 = 0.72 nM

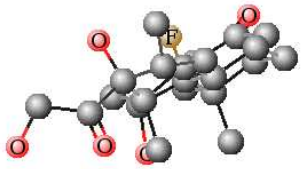
IC50 = 0.105 nM



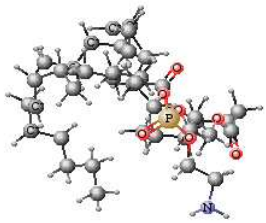
IC50 = 0.67 nM



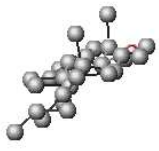
IC50 = 0.47 nM



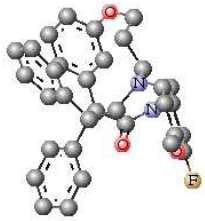
IC50 = 0.325 nM



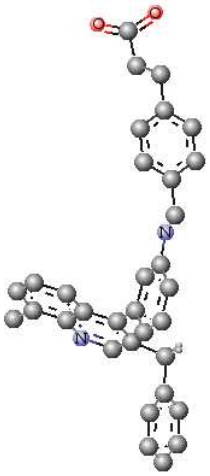
IC50 = 0.13 nM



IC50 = 0.22 nM

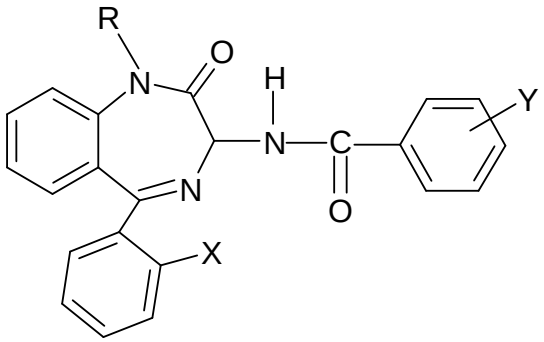


IC50 = 0.9 nM



IC50 = 0.1 nM

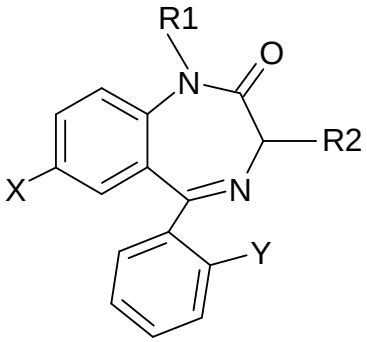
MOLECULES ETUDIEES DANS LE CHAPITRE III



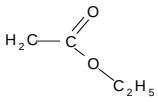
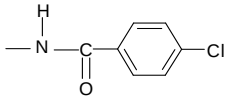
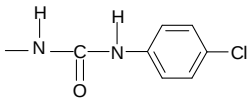
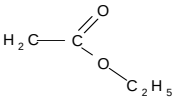
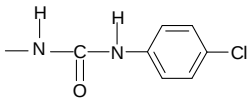
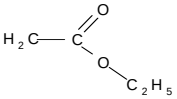
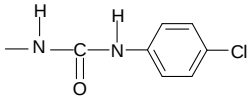
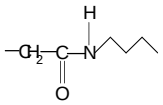
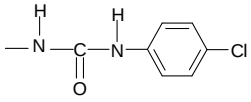
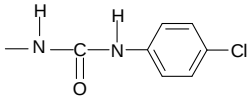
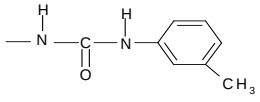
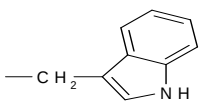
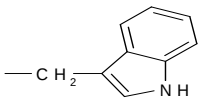
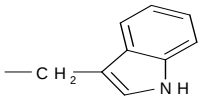
3-(Benzoylamino) benzodiazepines

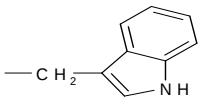
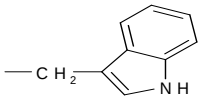
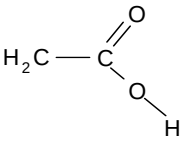
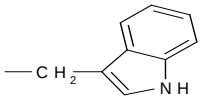
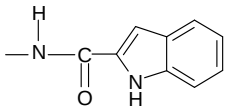
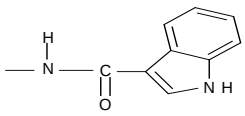
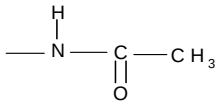
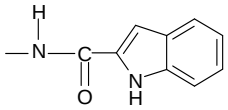
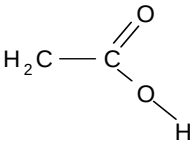
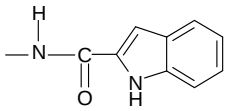
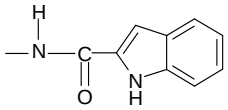
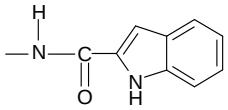
Compounds	**X**	**R**	**Y**	**-log(IC50)CCK-1**	**Ar**
MO 01	F	H	*p*-Cl	8.22	1.379
MO 02	F	H	*p*-NO_2	6.96	4.866
MO 03	F	-CH_3	*p*-Cl	8.64	0.9025
MO 04	H	-CH_3	*p*-Cl	8.08	1.5897
MO 05	H	H	*p*-Cl	7.39	3.164
MO 06	F	H	H	6.82	5.5897
MO 07	F	H	*o*-Cl	5.25	26.923
MO 08	H	H	*o*-Cl	5.12	30.615
MO 09	H	-CH_3	*o*-Cl	5.77	16
MO 10	F	H	*p*-CF_3	7.82	2.062
MO 11	F	H	*p*-CH_3	7.68	2.364
MO 12	H	H	*m*-Cl	7.09	4.272
MO 13	F	H	*p*-OCH_3	7.02	4.579
MO 14	F	H	*p*-$N(CH_3)_2$	6.57	7.179
MO 15	F		3,4-di-OCH_3	5.77	15.948
MO 16	H	H	3,4-di-Cl	7.54	2.723
MO 17	H	H	*p*-Br	8.60	0.943

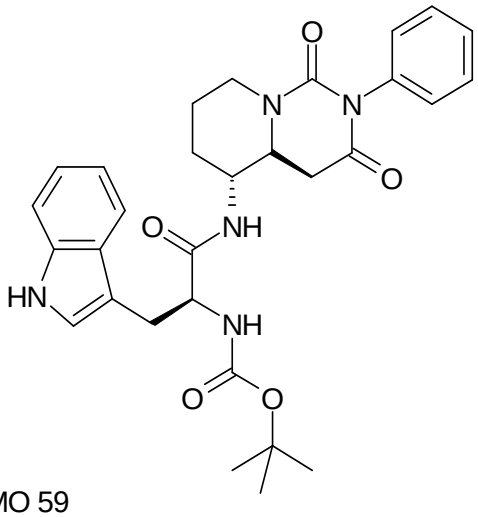
MO 18	H	$-CH_3$	*p*-SCH_3	6.66	0.656
MO 19	H	H	*p*-F	6.32	0.918
MO 20	F	H	*m*-Br	8.46	1.082
MO 21	H	$-CH_3$	*m*-SCF_3	5.82	15.179
MO 22	H	H	*p*-CF_3	6.62	6.821
MO 23	F	H	*p*-Br	9.10	5.743
MO 24	F	$-CH_3$	*p*-*t*-Bu	7.72	2.272
MO 25	F	$-CH_3$	*p*-I	9.10	0.574
MO 26	F	$-CH_3$	*o*-Br	5.21	2.8
MO 27	F	$-CH_3$	*p*-CN	7.37	3.225
MO 28	H	$-CH_3$	*p*-Ph	4.00	93.846
MO 29	H	H	*p*-*t*-Bu	5.85	14.718
MO 30	H	H	3,5-di-Cl	6.30	9.385
MO 31	H	H	*p*-OH	5.75	16.308
MO 32	F	H	*m*-I	9.12	0.559
MO 33	H	$-CH_3$	*p*-CN	6.14	11.026
MO 34	F	H	*o*-I	6.10	11.487
MO 35	H	$-CH_3$	*o*-NH_2, *p*-Cl	9.05	0.6
MO 36	H	$-CH_3$	*p*-CF_3	8.89	0.703
MO 37	H	$-CH_3$	*p*-CF_3	7.70	2.323
MO 38	H	$-CH_3$	*o*-I	5.96	13.179



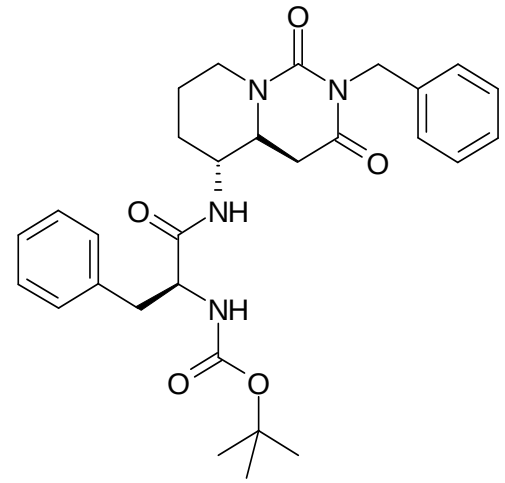
Benzodiazepine derivatives

Compounds	**X**	**Y**	**R1**	**R2**	**-log IC_{50}**	**Ar**
MO 39	H	F			6.96	4.866
MO 40	H	F	H		6.96	4.866
MO 41	H	F			6.66	6.564
MO 42	H	H			6.44	8.154
MO 43	H	H			6.29	9.487
MO 44	H	H	CH_3		7.30	3.462
MO 45	H	H	CH_3		8.10	1.569
MO 46	Cl	H	H		5.47	21.589
MO 47	Cl	H	CH_3		5.86	14.615
MO 48	H	H	H		5.93	13.641

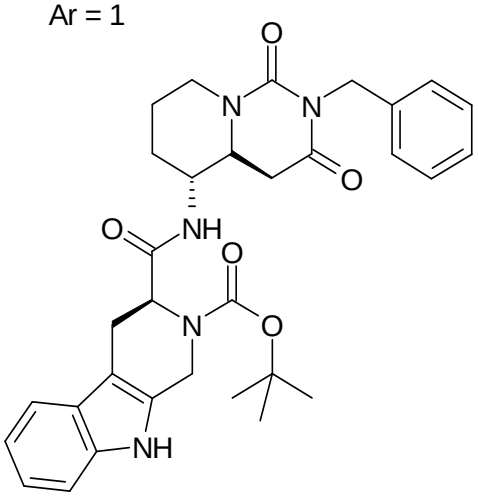
MO 49	H	F	H		6.30	9.436
MO 50	H	F	CH_3		6.56	7.282
MO 51	H	F			6.52	7.538
MO 52	H	H	H		8.32	12.462
MO 53	H	H	H		5.96	13.231
MO 54	H	F	H		4.40	63.077
MO 55	H	H	CH_3		8.95	0.662
MO 56	H	H			8.85	0.733
MO 57	H	F	H		8.67	0.882
MO 58	H	F	CH_3		8.85	0.733



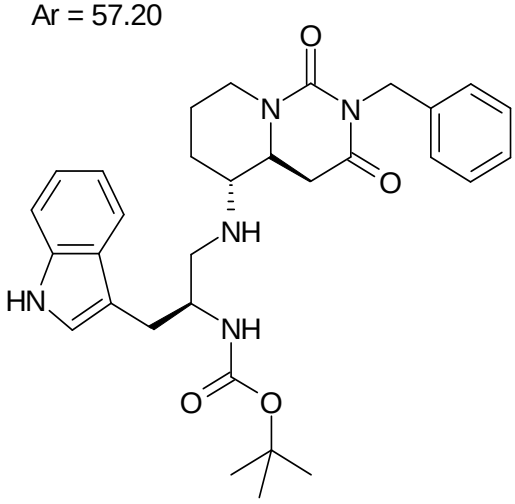
MO 59
Ar = 1



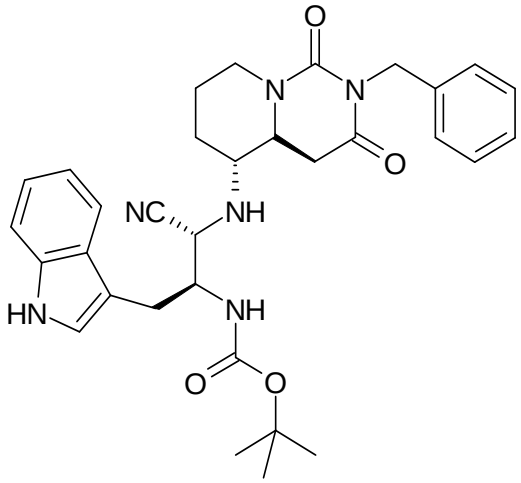
MO 60
Ar = 57.20



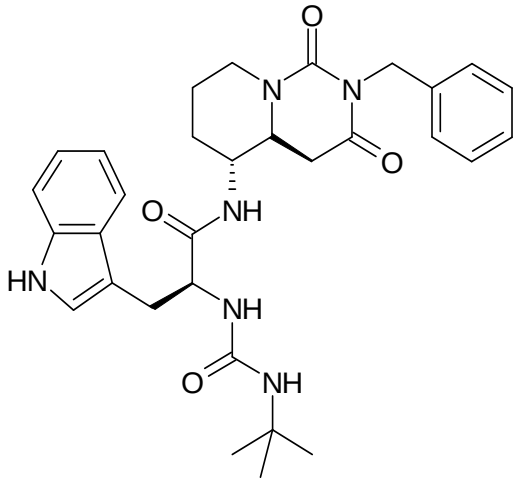
MO 61
Ar = 191.525



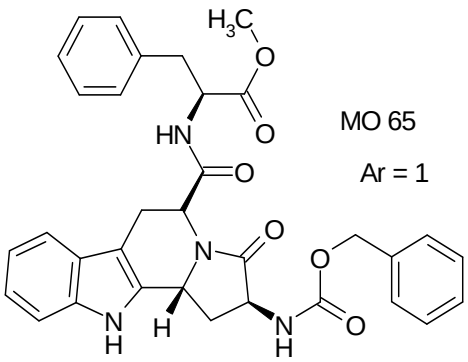
MO 62
Ar = 847.478



MO 63
Ar = 6.517

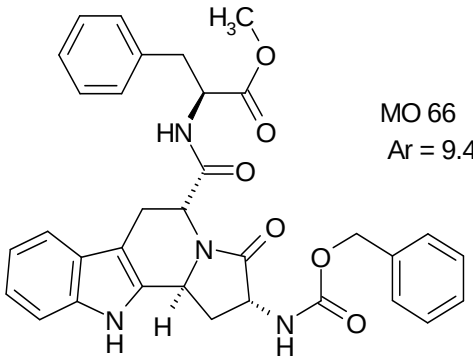


MO 64
Ar = 0.771



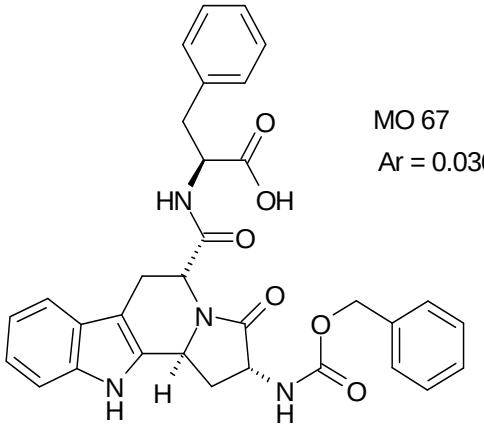
MO 65

Ar = 1



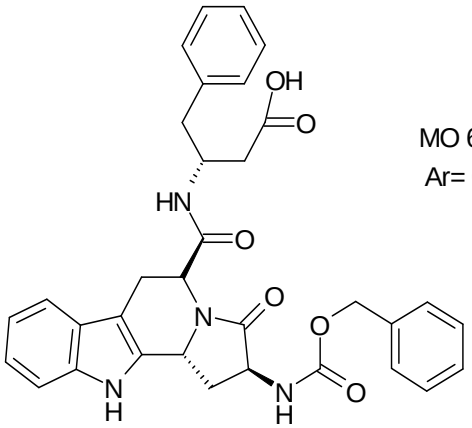
MO 66

Ar = 9.473



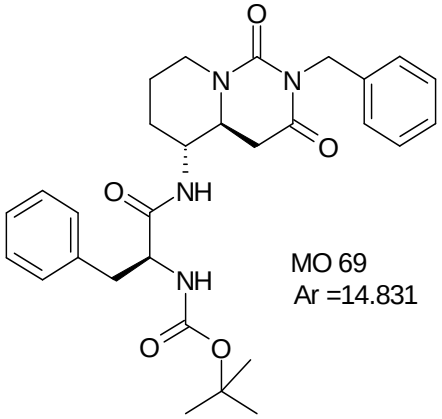
MO 67

Ar = 0.036



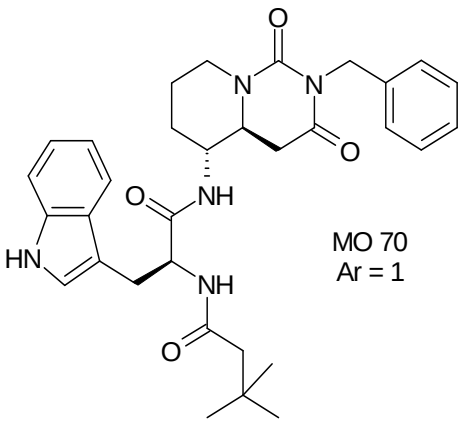
MO 68

Ar= 0.156



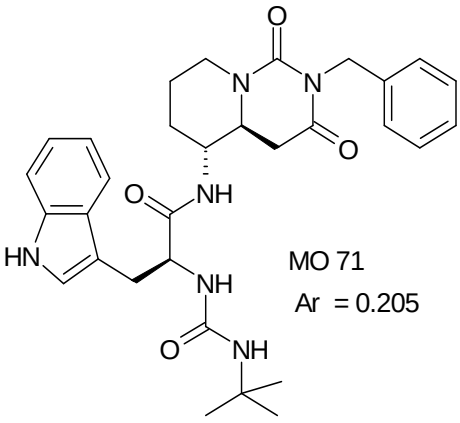
MO 69

Ar =14.831



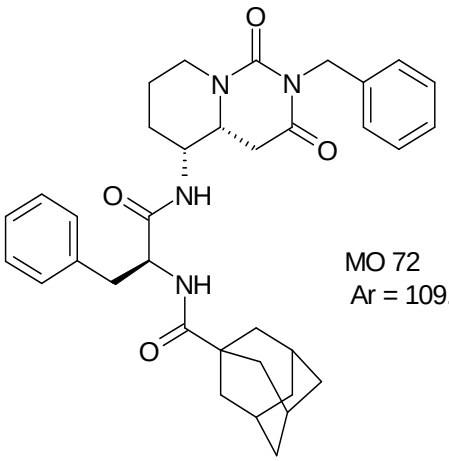
MO 70

Ar = 1



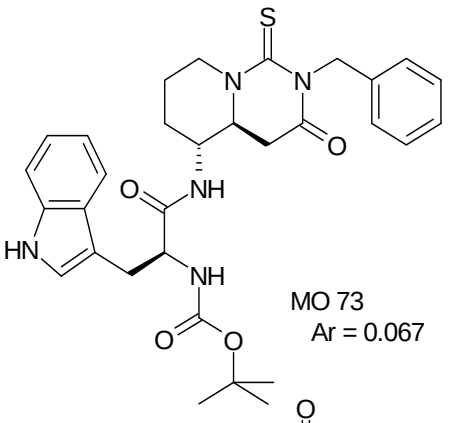
MO 71

Ar = 0.205

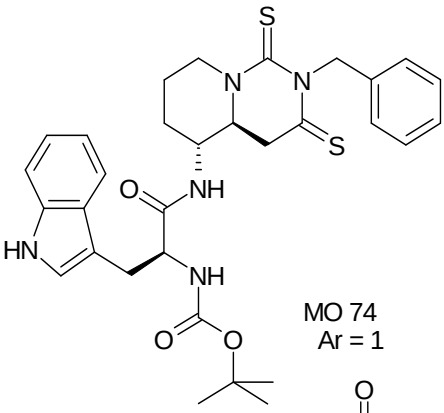


MO 72

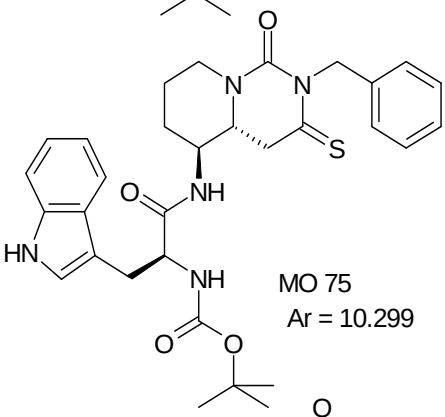
Ar = 109.707



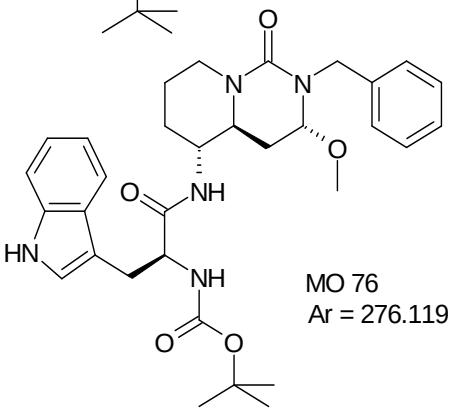
MO 73
Ar = 0.067



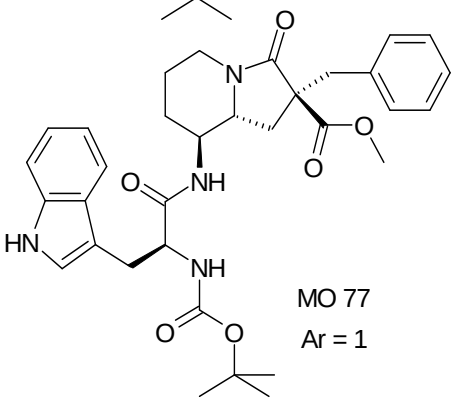
MO 74
Ar = 1



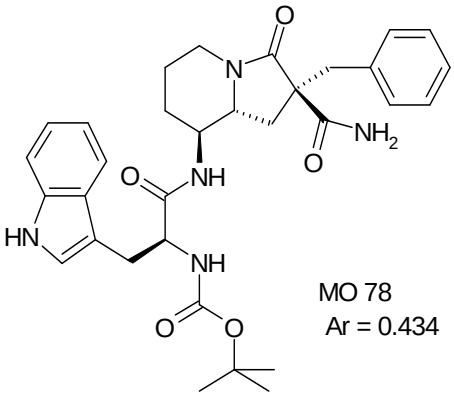
MO 75
Ar = 10.299



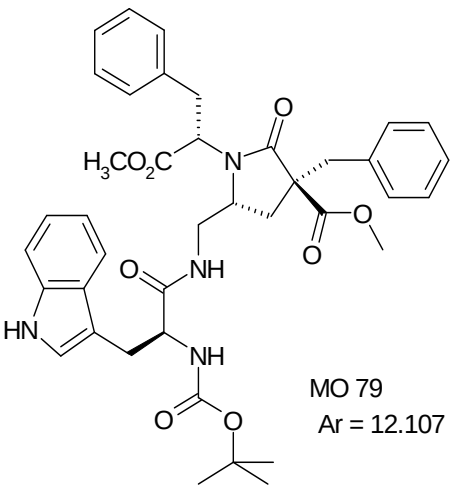
MO 76
Ar = 276.119



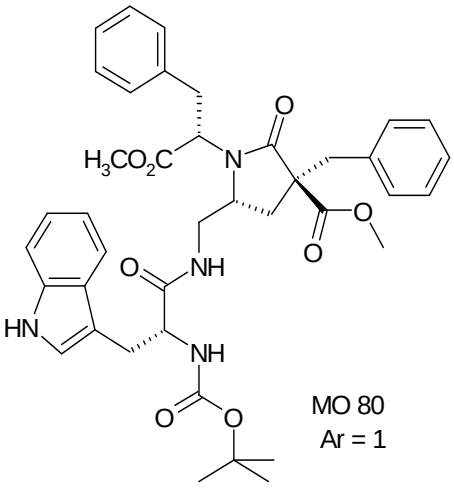
MO 77
Ar = 1



MO 78
Ar = 0.434



MO 79
Ar = 12.107



MO 80
Ar = 1