VI.1 Ajustement global de plusieurs courbes
(tutoriel *Sa*)

SA - EXERCICE 17
AJUSTEMENT GLOBAL DE PLUSIEURS COURBES (TUTORIEL *SA*)

1. Ajustement multi-variables

Bien qu'il s'agisse, ici encore, d'un exemple artificiellement construit, il illustre l'enchaînement des diférentes étapes que l'on retrouve assez systématiquement.

Il s'agit ici d'une reprise de l'exemple 1 fourni avec *Sa*, et expliqué dans le manuel, tome II. Le but de ces exemples est surtout d'illustrer les possibilités de programmation, que nous avons voulu garder la plus simple possible ici. Vous pourrez y trouver des compléments utiles.

Supposons que nous ayons suivi une réaction transformant un réactif unique, A, qui disparaît complètement, en un produit, P, unique également, par spectroscopie d'absorption UV-Visible, à deux longueurs d'onde significatives, λ_1 et λ_2, dans une

cuvette de trajet optique 1cm. La concentration initiale en composé A était 5×10^{-5} mol.L^{-1}, c'est donc également la concentration finale du produit P.

Les données sont tracées sur la figure 1 et sont disponibles dans le fichier *ex17_1.exp*. Ce fichier comprend trois colonnes : les temps, l'absorbance à λ_1 et l'absorbance à λ_2.

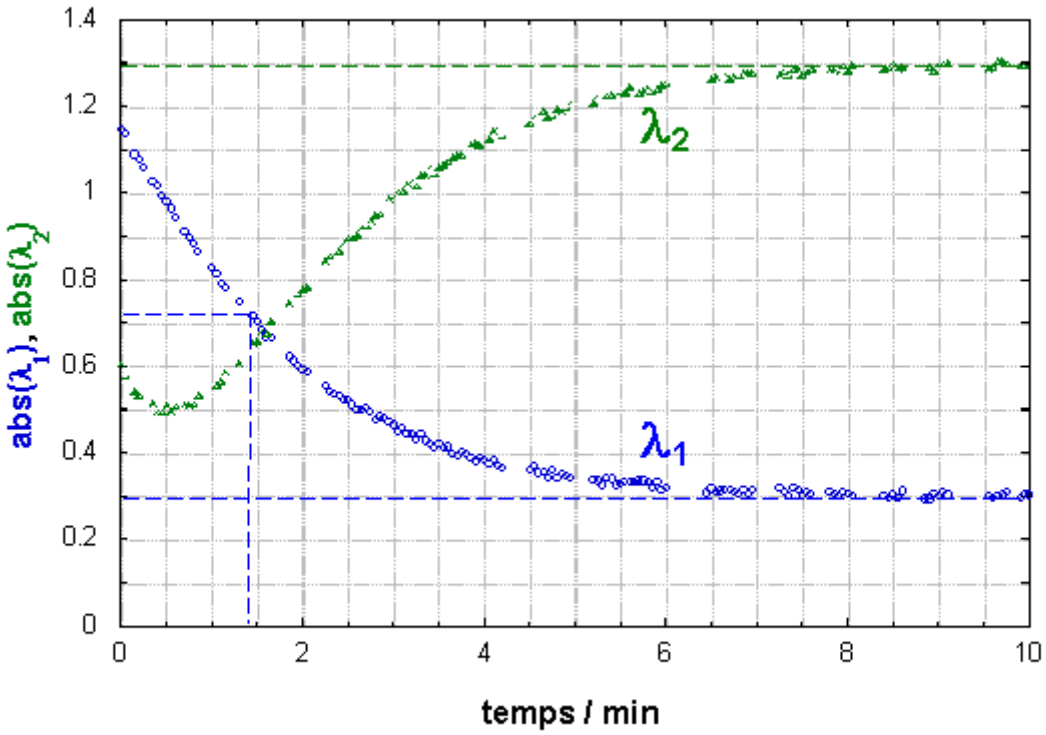

Fig. 1 Données expérimentales du fichier ex17_1.exp.

Il y a des "trous" dans ces données, correspondant par exemple à une perte de données accidentelle du système de mesure. Nous verrons que cela ne gênera pas le traitement.

1.1 Examen des données

La première chose à faire est d'examiner attentivement ces données afin de **proposer un modèle** capable de les expliquer, en commençant par le plus simple possible, et d'estimer manuellement d'abord ses paramètres.

Certains paramètres peuvent être estimés indépendamment du modèle. Ainsi, de l'absorbance à $t = 0$, on déduit immédiatement les coefficients d'extinction molaire de A :

$\varepsilon^{A}_{1} = 1.15 / 5\times10^{-5} = 2.3\times10^{4}\ \text{mol}^{-1}.\text{L.cm}^{-1}$
$\varepsilon^{A}_{2} = 0.6 / 5\times10^{-5} = 1.2\times10^{4}\ \text{mol}^{-1}.\text{L.cm}^{-1}$
et de l'absorbance à t_∞, ceux du produit P :
$\varepsilon^{P}_{1} = 0.3 / 5\times10^{-5} = 6\times10^{3}\ \text{mol}^{-1}.\text{L.cm}^{-1}$
$\varepsilon^{P}_{2} = 1.3 / 5\times10^{-5} = 2.6\times10^{4}\ \text{mol}^{-1}.\text{L.cm}^{-1}$

L'allure de la courbe à λ_2, d'abord décroissante puis croissante, suggère le passage par un intermédiaire réactionnel B. On essaiera donc de traiter ces données par le modèle le plus simple correspondant, c'est à dire composé de deux réactions monomoléculaires successives :

$$\begin{array}{lllll} A & \rightarrow & B & k_1 & \text{(r a)} \\ B & \rightarrow & P & k_2 & \text{(r b)} \end{array}$$

Le fait de savoir que le réactif A est seul présent au départ, et le produit P seul présent à la fin de la réaction, c'est-à-dire qu'il s'agit d'une réaction totale, constitue évidemment une donnée essentielle : sans cela, l'estimation des coefficients d'extinction molaire de A et P ne serait pas possible, d'une part, et les réactions (r a) et (r b) pourraient être réversibles, d'autre part.

Puisque l'absorbance à λ_2 décroît en début de réaction, le coefficient d'extinction molaire de B doit être inférieur à celui de A : on essaiera $\varepsilon^{B}_{2} = 8\times10^{3}\ \text{mol}^{-1}.\text{L.cm}^{-1}$. Et à λ_1, il doit avoir une valeur intermédiaire entre celles de A et P, soit $\varepsilon^{B}_{1} = 1.4\times10^{4}\ \text{mol}^{-1}.\text{L.cm}^{-1}$, par exemple.

Pour estimer les constantes de vitesse, on peut supposer en un premier temps que la courbe à λ_1 correspond principalement à la disparition de A. Le temps de demi-vie correspondant étant de 1.4 min, on choisira comme valeur d'essai $k_1 = 0.69 / 1.4 \approx 0.5\ \text{min}^{-1}$

On s'attendra cependant à ce que cette valeur soit trop faible. En effet, étant donné que $\varepsilon^{P}_{1} \neq 0$, la courbe à λ_1 correspond aussi à l'apparition du produit P, de sorte que le temps de demi-vie de la disparition de A est vraisemblablement plus court, et k_1 plus élevé.

Quant à k_2, on peut dire, toujours en première approximation, qu'il est du même ordre de grandeur, et on le mettra initialement à la même valeur.

Il y a toujours différentes façon d'estimer les paramètres de départ pour une optimisation. Nous en donnons une ici en exemple. L'essentiel est de tomber dans une plage réaliste de leurs valeurs. Plusieurs aller-retour entre simulation et estimation sont souvent nécessaires.

1.2 Programmation

Le programme correspondant au mécanisme (r a-b) ne présente pas de difficultés particulières. Les concentrations de A, B et P sont d'abord calculées par intégration numérique, puis les absorbances aux deux longueurs d'onde sont calculées dans une boucle sur le nombre de points `npt`.

```
void fappel()
{
   // tous les paramètres doivent rester > 0
   for (int i = 0; i < np; i++)  // np : nombre total de paramètres
      p[i] = fabs(p[i]);

   // intégration numérique :
   srkvi(n_diff, &ca[first_var], ind, npt, h0, tol, iset, jacob, h_compt,  c_min);

   // calcul algébrique des absorbances :
   for (int k = 0; k < npt; k++) // npt : nombre total de points calculés
   {
      ca[3][k] = p[2]*ca[0][k] + p[3]*ca[1][k] + p[4]*ca[2][k]; // abs1
      ca[4][k] = p[5]*ca[0][k] + p[6]*ca[1][k] + p[7]*ca[2][k]; // abs2
   }
}
```

Noter une petite nouveauté : comme nous allons utiliser ce programme pour une optimisation, nous avons ajouté, avant tout autre chose dans `fappel`, une boucle sur le nombre de paramètres `np` dans le but de maintenir ces-derniers à une valeur positive, quoi que tente de faire l'optimisateur, puisque des valeurs négatives de coefficients d'extinction molaire ou de constantes de vitesse n'auraient pas de sens. La fonction `fabs` renvoie la valeur absolue de son argument (un nombre en virgule **f**lottante, d'où le "`f`", absolument nécessaire : la fonction `abs` existe mais provoquerait une erreur car elle n'accepte que des arguments entiers).

Nous avons appelé `k` la variable de boucle dans le calcul des absorbances uniquement par souci de clarté. Elle pourrait en effet s'appeler tout aussi bien `i`, comme dans la boucle précédente, puisque de telles variables, déclarées à l'intérieur d'une boucle, sont strictement locales à leur boucle.

Télécharger *ABP.cpp* Télécharger *ABP.sac*

1.3 Simulation, estimation des paramètres de départ

Le fichier *ABP.sac* contient les paramètres de première estimation ci-dessus. Assurez-vous que les cases *obs* des variables 3 et 4 sont bien cochées et faites lire

le fichier expérimental *ex17_1.exp*. La simulation avec ces paramètres donne les résultats rassemblés sur la figure 2.

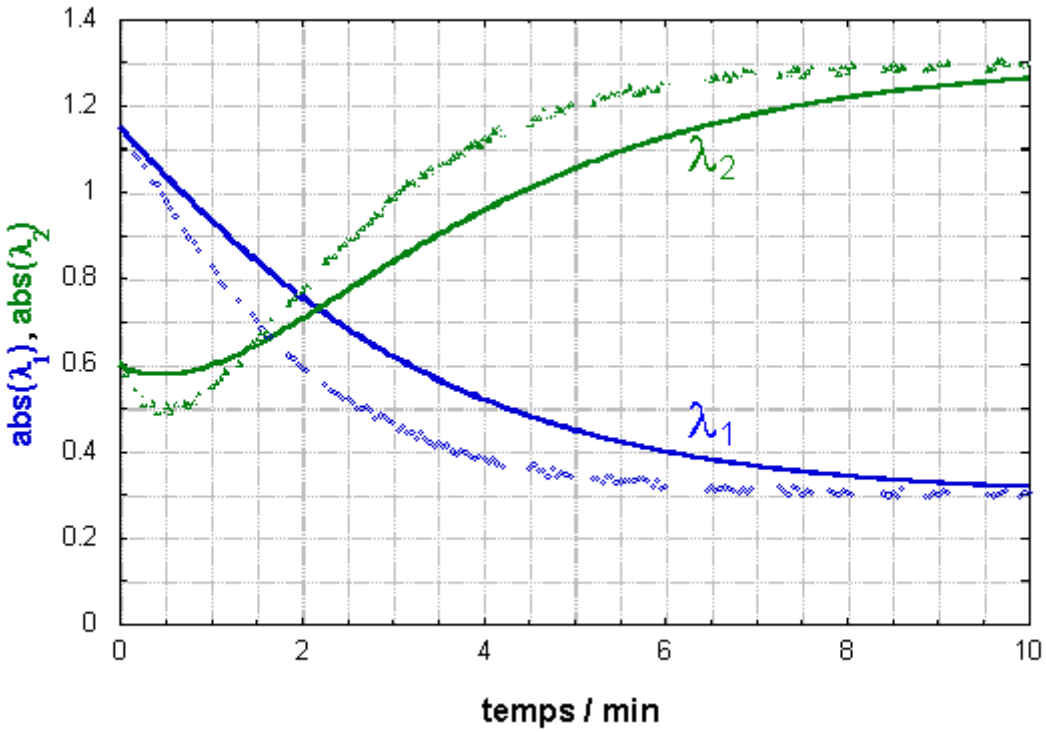

Fig. 2 Première estimation manuelle des paramètres.

On n'est pas très loin du compte, et il serait peut-être possible de lancer l'optimisation à partir de là (vous pouvez bien sûr l'essayer, au risque de ne trouver qu'un "faux minimum", et dans ce cas, de devoir affiner l'estimation initiale). Toutefois, on constate, comme attendu, que la vitesse est globalement un peu trop lente. On peut donc l'augmenter un peu : $k_1 = 0.8\ \text{min}^{-1}$, par exemple. D'autre part, l'absorbance ne descend pas suffisamment à λ_2, on peut donc essayer $\varepsilon^B_2 = 6\times10^3\ \text{mol}^{-1}.\text{L}.\text{cm}^{-1}$.

1.4 Ajustement

Avec ces valeurs de départ, ajustez, **uniquement $\mathbf{k_1}$, $\mathbf{k_2}$, $\boldsymbol{\varepsilon}^{\mathbf{B}}_{\mathbf{1}}$ et $\boldsymbol{\varepsilon}^{\mathbf{B}}_{\mathbf{2}}$** pour commencer.

Remarquez tout d'abord que les "trous" dans les données, ne perturbent pas l'optimisateur de *Sa*, qui ne demande à l'intégrateur que les valeurs correspondant à des données expérimentales. Cette particularité permet également de traiter des données expérimentales dont l'incrément de la variable indépendante (le temps en cinétique) n'est pas constant. Il peut être logarithmique par exemple, ou complètement aléatoire.

L'ajustement est par ailleurs excellent et doit donner des valeurs ajustées voisines de:

$k_1 = 0.998 \approx 1\ \text{min}^{-1}$
$k_2 = 0.698 \approx 0.7\ \text{min}^{-1}$
$\varepsilon^{B}_{1} = 1.59\times10^{4} \approx 1.6\times10^{4}\ \text{mol}^{-1}.\text{L.cm}^{-1}$
$\varepsilon^{B}_{2} = 3.07\times10^{3} \approx 3\times10^{3}\ \text{mol}^{-1}.\text{L.cm}^{-1}$
avec l'erreur résiduelle : 3.2×10^{-5}

Il se peut que vous obteniez des valeurs *très légèrement* différentes si vous ne partez pas exactement des mêmes valeurs ou si vous modifiez l'ordre initial des paramètres.

Tracez les résiduels, d'abord à λ_1 puis à λ_2 : ils sont tous deux bien répartis aléatoirement. Ce modèle semble correct.

Tracez également les diagrammes de sensibilité globale. Vous constatez d'abord qu'ils ont tous les quatre la forme parabolique indiquant qu'on est en présence d'un minimum réel (local ou global) . Mais ce qui est le plus remarquable, c'est la taille du diagramme de ε^{B}_{2} : il est considérablement plus petit que les autres. Sa sensibilité est donc beaucoup plus faible et, par conséquent, la précision sur la valeur calculée est beaucoup plus faible également. Pour en mesurer toute l'importance, fixez ε^{B}_{2} à une valeur différente, par exemple $3.5\times10^{3}\ \text{mol}^{-1}.\text{L.cm}^{-1}$ soit une valeur supérieure d'environ 16%, puis simulez. Vous ne verrez pratiquement pas de différence. En fait, d'une part, cette valeur (voisine de 3×10^{3}) correspond au plus petit des coefficients d'extinction molaire et, d'autre part, l'amplitude de variation de l'intermédiaire réactionnel B est aussi plus petite que celles de A et P, comme on peut le voir en traçant les concentrations, ce qui explique une plus faible contribution de ε^{B}_{2} à l'erreur résiduelle.

Bien que l'ajustement obtenu semble excellent, on peut se demander si on ne pourrait pas l'améliorer encore en ajustant également les coefficients d'extinction molaire de A et P, qui n'ont été estimés que graphiquement. Pour cela, il suffit de les déclarer ajustables **en plus des précédents**. Il est en effet important de laisser la liberté de réajuster simultanément tous les paramètres.

Vous constatez que le résultat de ce nouvel ajustement est pratiquement identique au précédent : aucun paramètre ne change significativement, et l'erreur résiduelle ne diminue que d'une infime quantité. Le tracé des résiduels est toujours excellent.

Quant aux diagrammes de sensibilité globale, ils montrent cette fois une sensibilité nettement plus grande de ε^{P}_{2}, ce qui s'explique simplement par le fait qu'il s'agit du plus grand coefficient d'extinction molaire, associé à la plus grande variation de concentration, à l'inverse de ε^{B}_{2}.

En effectuant un agrandissement des autres diagrammes (voir l'aide en ligne pour la façon d'utiliser le zoom), vous pouvez constater que tous les autres paramètres, sauf ε^{B}_{2} bien sûr, sont de sensibilité voisine.

Fixez maintenant ε^{P}_{2} a une valeur légèrement fausse, $2.65\times10^{4}\ \mathrm{mol^{-1}.L.cm^{-1}}$ par exemple, déclarez le non-ajustable et relancez une optimisation : cette sensibilité élevé de ε^{P}_{2} a pour conséquence de décaler assez fortement la valeur ajustée des autres paramètres ($k_1 \approx 1.27\ \mathrm{min^{-1}}$ et $k_2 \approx 0.55\ \mathrm{min^{-1}}$ par exemple). Vous pouvez constater alors que les tracés des résiduels indiquent un ajustement défectueux.

1.5 Unicité de la solution

Remettez ε^{P}_{2} à sa valeur correcte. Il reste maintenant à vérifier que la solution trouvée est stable, c'est à dire que l'on retrouve systématiquement les mêmes valeurs de paramètres ajustés en partant de valeurs initiales différentes, mais permettant toutefois à l'ajustement d'aboutir. Nous proposons, par exemple le jeu de paramètres de départ suivant, correspondant à une situation opposée à celle du premier ajustement :

$k_1 = 1.2\ \mathrm{min^{-1}}$
$k_2 = 0.8\ \mathrm{min^{-1}}$
$\varepsilon^{B}_{1} = 1.7\times10^{4}\ \mathrm{mol^{-1}.L.cm^{-1}}$
$\varepsilon^{B}_{2} = 2\times10^{3}\ \mathrm{mol^{-1}.L.cm^{-1}}$

Vous devez retrouver pratiquement les mêmes valeurs ajustées. Vérifiez tout : erreur résiduelle, tracé des résiduels, diagrammes de sensibilité.

Vous pouvez renouveler cette vérification à partir de points de départ encore différents, relancer l'optimisation avec un ordre initial différent des paramètres (utilisez le bouton *Brouiller*).

2. Ajustement multi-expériences

Nous nous proposons maintenant de tester le même modèle (r a-b), mais cette fois simultanément sur trois expériences réalisées avec des concentrations initiales différentes et suivies chacune par absorption Uv-Visible à deux longueurs d'onde. Le but poursuivi ici est principalement la programmation correspondante, ainsi que la constitution et l'utilisation des fichiers multi-expériences.

Il s'agit ici d'une reprise de l'exemple 2 fourni avec *Sa*, que l'on pourra consulter en supplément.

2.1 Programmation

Dans la version précédente, écrite pour une seule expérience, nous avons utilisé les cinq variables `ca[0][ ]` à `ca[4][ ]`. Pour prendre en compte trois expériences indépendantes, on utilisera ces mêmes variables 0 à 4 pour la première, `ca[5][ ]` à `ca[9][ ]` pour la seconde et `ca[10][ ]` à `ca[14][ ]` pour la troisième. La variable indépendante `ind[ ]` reste commune à toutes les expériences.

La fonction `eqdiff`, elle, sera strictement identique : les variables `y[0]` à `y[2]`, `dy[0]` à `dy[2]` désigneront toujours les concentrations ou les dérivées, respectivement, de A, B et P.

L'intégrateur numérique sera appelé trois fois :
une fois avec `first_var = 0`, comme précédemment
une fois avec `first_var = 5`
et une fois avec `first_var = 10`

Et les absorbances seront calculées également trois fois à partir des concentrations correspondant à chaque expérience.

La fonction `fappel` du programme pourrait, et peut réellement, s'écrire, de façon analogue au précédent :

```
void fappel()
{
   for (int i = 0; i < 8; i++) // i < 8 et non i < np !
      p[i] = fabs(p[i]);

   // intégration 1ère expérience :
   first_var = 0;        // numéro de la première variable à intégrer
   srkvi(n_diff, &ca[first_var], ind, npt, h0, tol, iset, jacob, h_compt, c_min);
   // intégration 2ème expérience :
   first_var = 5;        // numéro de la première variable à intégrer
   srkvi(n_diff, &ca[first_var], ind, npt, h0, tol, iset, jacob, h_compt, c_min);
   // intégration 3ème expérience :
   first_var = 10;       // numéro de la première variable à intégrer
   srkvi(n_diff, &ca[first_var], ind, npt, h0, tol, iset, jacob, h_compt, c_min);

   //  Calcul des absorbances :
   for (int j = 0; j < npt; ++k) {
```

```
      ca[3][j] = p[2]*ca[0][j] + p[3]*ca[1][j] + p[4]*ca[2][j] + p[8];
      ca[4][j] = p[5]*ca[0][j] + p[6]*ca[1][j] + p[7]*ca[2][j] + p[8];

      ca[8][j] = p[2]*ca[5][j] + p[3]*ca[6][j] + p[4]*ca[7][j] + p[9];
      ca[9][j] = p[5]*ca[5][j] + p[6]*ca[6][j] + p[7]*ca[7][j] + p[9];

      ca[13][j] = p[2]*ca[10][j] + p[3]*ca[11][j] + p[4]*ca[12][j] + p[10];
      ca[14][j] = p[5]*ca[10][j] + p[6]*ca[11][j] + p[7]*ca[12][j] + p[10];
   }
}
```

Lorsque plusieurs expériences indépendantes sont réalisées, il est fréquent que de petits détails de réglage introduisent un décalage indépendant de chaque jeu de données. Il est alors nécessaire de prévoir un paramètre supplémentaire pour en tenir compte lors de l'ajustement global. C'est le rôle ici des paramètres `p[8]` pour la 1ère expérience, `p[9]` pour la seconde et `p[10]` pour la troisième. Comme ces décalages peuvent être aussi bien positifs que négatifs, il ne faut pas les inclure dans la boucle de contrôle des paramètres positifs (3^{me} ligne de fappel), c'est pourquoi celle-ci ne doit s'exécuter que pour `i < 8`.

Mais il existe une écriture beaucoup plus élégante, plus facile à écrire et à corriger. Le module Modan utilise d'ailleurs systématiquement cette écriture :

```
1  void fappel()
2  {
3     for (int i = 0; i < 8; i++)     // i < 8 et non i < np !
4        p[i] = fabs(p[i]);

5     for (int k = 0; k < nexp; k++)    // boucle sur le nombre d'expériences
6     {
7        first_var = k*nv_mod;
8        srkvi(n_diff,&ca[first_var],ind,npt,h0,tol,iset,jacob,h_compt,c_min);

9        for (int j = 0; j < npt; ++j) {
10          Ca(k,3,j) = p[2]*Ca(k,0,j) + p[3]*Ca(k,1,j) + p[4]*Ca(k,2,j) + p[8+k];
11          Ca(k,4,j) = p[5]*Ca(k,0,j) + p[6]*Ca(k,1,j) + p[7]*Ca(k,2,j) + p[8+k];
12       }
13    }
14 }
```

(les numéros de lignes ont été ajoutés)

L'appel de l'intégrateur numérique et le calcul des absorbances ont été placés à l'intérieur d'une boucle sur la variable `nexp` (lignes 5 à 13). Celle-ci est initialisée à 3 dans la fonction `Identification`. Nous verrons qu'elle peut être modifiée par la suite.

Noter que `k` commence à zéro : pour une seule expérience, `nexp = 1`, le corps de la boucle est exécuté une fois, avec `k = 0`.

`first_var` est calculé (ligne 7) en fonction de l'indice de boucle, `k`, et du nombre total de variable du modèle `nv_mod`, qui a été évidemment préalablement initialisé à la valeur correcte (dans `Identification`). Ces variables prennent maintenant tout leur sens : `nv_mod` (= 5 ici) est différent de `nvar`, le nombre total de variable utilisées (15 ici).

La notation `Ca(k, i, j)`(lignes 10 et 11), utilisable aussi bien à gauche du signe égale qu'à droite, est strictement équivalente à `ca[k*nv_mod + i][j]`. Noter la différence d'écriture : C majuscule, parenthèses et virgules (pas de crochets).

Télécharger *ABPmultexp.cpp* Télécharger *ABPmultexp.sac*

Compilez et exécutez ce programme. Faites lire le fichier .sac et observez les onglets *Paramètres* et *Variables*.

Dans les paramètres, la seule nouveauté est l'addition des décalages (ou "offsets") mis à zéro pour l'instant. Les autres valeurs de paramètres pourront servir de point de départ.

Dans les variables, vous voyez que celles-ci ont été recopiées en trois groupes, de façon à correspondre au programme. C'est à vous normalement de le faire : cela n'est pas fait automatiquement. Les concentrations initiales des espèces pour chaque expérience correspondent aux données expérimentales de tous les fichiers qui suivent et ne doivent pas être changées.

2.2 Fichiers expérimentaux multi-expériences, mode standard

Télécharger *ABPmultexp.exp*

Les données provenant de toutes les expériences que l'on souhaite traiter simultanément doivent être rassemblées dans un fichier unique dont la première colonne contient les temps, communs à toutes les colonnes, et les colonnes suivantes contiennent les grandeurs expérimentales suivies, dans l'ordre où elles apparaissent dans le programme.

Le fait que la colonne de temps soit unique pose un problème si les données expérimentales ne sont pas échantillonnées exactement de la même façon dans chaque expérience. Il faut alors constituer une colonne de temps contenant toutes les valeurs qui seront utilisées par l'une ou l'autre des expériences, puis affecter une valeur particulière signifiant *donnée absente*, −1 par défaut, dans les colonnes suivantes lorsqu'il n'y a pas de valeur correspondant au temps courant.

1 - La valeur par défaut pour ABSENT est −1. Cela convient le plus souvent, la plupart des données expérimentales restant positives. Ce n'est pas toujours le cas cependant, et on peut modifier cette valeur par défaut, avant de compiler le programme, par la fonction *Limits* de Nestor.

2 - La construction d'un fichier multi-expériences en mode "normal" nécessite en général l'usage d'un tableur permettant de manipuler des colonnes. Il faut ensuite sauvegarder ce fichier en mode texte (ASCCII). Les colonnes peuvent être séparées par des espaces ou des tabulations. N'oubliez pas qu'il ne doit contenir aucune donnée non numérique, comme des titres de colonnes par exemples, que certains logiciels incorporent automatiquement. Si c'est le cas, il faudra les supprimer à l'aide d'un simple éditeur.

Ouvrez le fichier *ABPmultexp.exp* avec n'importe quel éditeur (éditeur *Sa* ou autre). Vous constatez que les dernières lignes, correspondant aux temps de 9.05 à 10.1, des deux dernières colonnes (troisième expérience) ont été mises à −1.

Vérifiez que les cases *obs* des six variables expérimentales sont correctement cochées dans l'onglet *Variables*.

Dans l'onglet *Données exp.*, assurez-vous que **le bouton *Multi* n'est pas enfoncé**. Vous constatez que *n_exp* est à 1, bien que vous l'ayez initialisé à 3 dans `Identification`. C'est un défaut venant du fait qu'en réalité cette variable est gérée à plusieurs niveaux, et automatiquement dans le mode "multi" (voir plus loin), il faut y remédier manuellement. Affectez 3 à *n_exp* et activez la mémorisation de la dernière session (menu *Fichier* de la fenêtre principale) de façon à ce que la valeur 3 revienne lors d'une réouverture du programme.

Faites lire maintenant ce fichier. Il peut être nécessaire d'élargir la fenêtre *Infos* pour voir toutes les variables expérimentales. Constatez que les points mis à −1 dans le fichier sont affichés avec un tiret -.

Cochez la case *Actif* et simulez.

Si vous n'avez pas mis *n_exp* à 3, vous constaterez que les calculs correspondant aux deuxième et troisième expériences ne sont pas effectués.

Le premier défaut de la simulation avec ces paramètres vient des décalages : la simulation de la première expérience est systématiquement environ à 0.04 en dessus, on mettra donc son offset à −0.04 ; celle de la seconde est vers 0.1 en dessous, on mettra donc son offset à 0.1 ; celle de la troisième est vers 0.16 en dessus et on mettra également son offset à −0.16.

Ce recalage étant fait, déclarez les offsets ajustables et lancez une optimisation.

Rappelons que les paramètres ajustables de doivent jamais être nuls au départ.

Vous devez trouver les paramètres ajustés suivants :

$k_1 = 1\ min^{-1}$
$k_2 = 0.55\ min^{-1}$
$\varepsilon^A_1 = 2.3\times10^4\ mol^{-1}.L.cm^{-1}$
$\varepsilon^B_1 = 1.6\times10^4\ mol^{-1}.L.cm^{-1}$
$\varepsilon^P_1 = 6.0\times10^3\ mol^{-1}.L.cm^{-1}$
$\varepsilon^A_2 = 1.2\times10^4\ mol^{-1}.L.cm^{-1}$
$\varepsilon^B_2 = 1.4\times10^4\ mol^{-1}.L.cm^{-1}$
$\varepsilon^P_2 = 2.6\times10^4\ mol^{-1}.L.cm^{-1}$
$offset\_1 = -0.05$

offset_2 = 0.1
offset_3 = −0.15

2.3 Fichiers expérimentaux multi-expériences, mode *multi*

Les fichiers en mode *multi* présentent les avantages suivants :

- ils sont plus faciles à construire car les expériences y figurent les unes à la suite des autres et un simple éditeur suffit en général pour cela

- chaque expérience a sa propre colonne de temps et peut donc être échantillonnée de façon différente, sans qu'il soit nécessaire d'ajouter des *données absentes* (−1 par défaut)

- ils permettent de traiter simultanément un nombre quelconque d'expériences, comportant chacune un nombre quelconque de variables expérimentales.

Nous vous recommandons ici de consulter la documentation en ligne de *Sa* (rubrique *Ajustement global et multi-expériences*, ou *chapitre XIII du Manuel*, p. 72-77). Nous ne donnons ici qu'une description sommaire.

A - Télécharger *ABPmultexp_multi3a.exp*

Contrairement aux fichiers expérimentaux en mode standard, les fichiers *multi* doivent contenir quelques indications non numériques, et être lus avec le bouton *Multi* (onglet *Données exp.*) enfoncé. Par exemple, le fichier *ABPmultexp_multi3a.exp*, strictement équivalent au fichier standard précédent *ABPmultexp.exp*, a la structure suivante :

```
multi 3
obs 2 : 3 4
0.00001.10040.54061
0.050001.08730.56037
0.100001.07190.55960
 ..................
 ..................
10.000  0.26348  1.2401
10.050  0.25887  1.2515
10.100  0.25647  1.2505
obs 2 : 3 4
0.0000  1.3857   0.86275
0.05000 1.3702   0.87579
0.10000 1.3526   0.88449
 ..................
 ..................
10.000  0.46202  1.6575
10.050  0.46193  1.6503
10.100  0.47076  1.6624
obs 2 : 3 4
0.0000  1.3341   1.1042
0.05000 1.3120   1.1073
0.10000 1.3111   1.1211
 ..................
 ..................
```

(vous pouvez le voir en l'ouvrant avec un éditeur)

Les éléments en rouge sont des mots clé. Tous les éléments d'une ligne non numérique doivent être séparés par un espace, y compris le signe " : " .

Le fichier doit commencer par le mot clé *multi*, suivi du nombre d'expériences.

Ensuite, chaque expérience doit commencer par le mot clé *obs*, suivi du nombre de variables observées de cette expérience (il peut être différent d'une expérience à l'autre), puis du mot clé " : " suivi de la liste des numéros de ces variables. Attention : il s'agit du numéro qu'elles auraient s'il y avait une seule expérience, ce sont donc des nombres inférieurs à `nv_mod`.

Suivent les valeurs numériques de l'expérience concernée : une première colonne de temps et les colonnes correspondant aux variables observées.

A la lecture d'un tel fichier, qui doit s'effectuer bouton *Multi* enfoncé :

- le nombre suivant le mot clé *multi* est affecté automatiquement à `n_exp`.

- les cases *obs* des différentes variables observées suivant les mots clé " : " sont automatiquement cochées, mais il s'agit cette fois de leur numéro réel, premier indice de `ca`, comme dans le mode standard.

- les valeurs numériques sont affectées convenablement, exactement comme dans le mode standard : une colonne temps commune à toutes les expériences est automatiquement générée et des *données absentes* (−1 par défaut) sont automatiquement mises où il le faut.

Ainsi, une fois le fichier *ABPmultexp_multi3a.exp* lu, les onglets *Variables* et *Données exp.* se présentent exactement comme dans le mode standard précédent. Le mode *multi* n'est qu'une facilité de gestion des multi-expériences, il ne change pas le mode de fonctionnement de *Sa*.

En conséquence, les fichiers *.sac* correspondants, qui doivent être cohérents avec le nombre d'expériences et de variables observées de chacune d'elles, et, en particulier, donner les concentrations initiales de chaque expérience, restent à votre charge. Le plus simple est de prévoir au départ un fichier *.sac* correspondant au nombre maximum d'expériences que l'on sera amené à traiter. Cela ne posera aucun problème ensuite de travailler avec un nombre inférieur.

Si des données sont **absentes à l'intérieur d'une même expérience**, par exemple si pour un temps donné une seule variable observée a été réellement mesurée alors que deux sont normalement attendues, vous devez vous-même, bien entendu, mettre une valeur *donnée absente* (−1 par défaut) à la place de la donnée manquante.

Naturellement, l'ajustement sur le fichier *ABPmultexp_multi3a.exp* sera strictement le même qu'en mode standard.

B - Télécharger *ABPmultexp_multi3b.exp*

Ouvrez avec un éditeur le fichier *ABPmultexp_multi3b.exp*. Il contient également 3 expériences. Mais, d'une part, la première n'est constituée que d'une seule variable observée : l'absorbance à λ_1. Tout cela est indiqué par les mots clé *multi*, *obs* et " : ". D'autre part, il y a des "trous" dans ces données et, en particulier, il n'y a aucune donnée correspondant au temps zéro.

Faites lire maintenant ce fichier, en mode *Multi*, partant de la situation précédente (fichier expérimental *ABPmultexp_multi3a.exp* préalablement lu). Ayant détecté l'absence de valeurs au temps zéro, le programme vous demande si vous voulez insérer automatiquement une ligne de temps zéro (avec des *données absentes*). En effet, une telle ligne est nécessaire si les concentrations initiales données dans le fichier .sac sont celles du temps zéro, comme c'est le cas le plus souvent. Répondez Oui.

La lecture se fait et vous constatez que des *données absentes* ont été mises à tous les endroits nécessaires. Constatez également, dans l'onglet *Variables*, que les cases *obs* cochées correspondent bien à la nouvelle situation.

Faites une simulation pour "voir" graphiquement cette nouvelle situation... peu réaliste, il est vrai. Les concentrations initiales correspondant à ces données sont toujours les mêmes. Vous pouvez donc conserver le même fichier *.sac* et les mêmes paramètres de départ.

Faites un ajustement... ça fonctionne toujours !

C - Télécharger *ABPmultexp_multi2.exp*

Faites lire maintenant le fichier *ABPmultexp_multi2.exp*, qui ne contient cette fois que deux expériences. Un premier message vous avertit que `n_exp` est changé. La question concernant la ligne de temps zéro vous est ensuite posée comme ci-dessus. Cette fois-ci, la première expérience comporte une variable observée, et la seconde en comporte deux. De sorte qu'au total il n'y a plus que 3 variables observées.

Remettez les paramètres de départ des fois précédentes.

Attention : si vous faites relire un fichier *.sac*, cela coche les cases *obs* comme indiqué dans celui-ci, écrasant l'effet de la lecture du fichier expérimental en mode *multi*. Il faut alors soit rectifier manuellement, soit faire relire le fichier *multi*.

Attention cette fois de ne pas faire ajuster l'offset de la troisième expérience. Cela provoquerait un *calcul avorté* de l'optimisation. En effet, l'optimisateur détecterait que ce paramètre n'a aucun effet sur la fonction d'erreur, et ne peut donc pas l'ajuster.

Lancez une optimisation. Le résultat sera peut-être décevant au premier coup. C'est que les conditions d'optimisation sont devenues plus difficiles, il n'y a plus beaucoup

de données. Si c'est le cas, relancez l'optimisation, en brouillant l'ordre initial des paramètres. Vous devez finir par obtenir les mêmes résultats que précédemment.

Toutefois, on peut se demander, dans cette situation encore plus que dans les précédentes, si la solution trouvée est stable et unique. Vous pouvez le tester en choisissant des paramètres de départ différents, comme nous l'avons vu en 1.5. Vous constaterez que, bien que l'optimisation soit parfois plus laborieuse maintenant, la solution reste stable : ces données très parcellaires sont encore suffisantes pour caler ce modèle.

L'ajustement global multi-expériences est un outil très puissant. Le mode *multi* apporte une aide appréciable, mais, en contrepartie, il exige d'être particulièrement attentif : on doit veiller en permanence à ce que toutes les données figurant dans les onglets *Paramètres*, *Variables* et *Données exp.* soient entièrement cohérentes.