Didier Auroux

Laboratoire Jean-Alexandre Dieudonné

Université de Nice Sophia Antipolis

auroux@unice.fr

Assimilation de données : estimation de paramètres et différentiation automatique

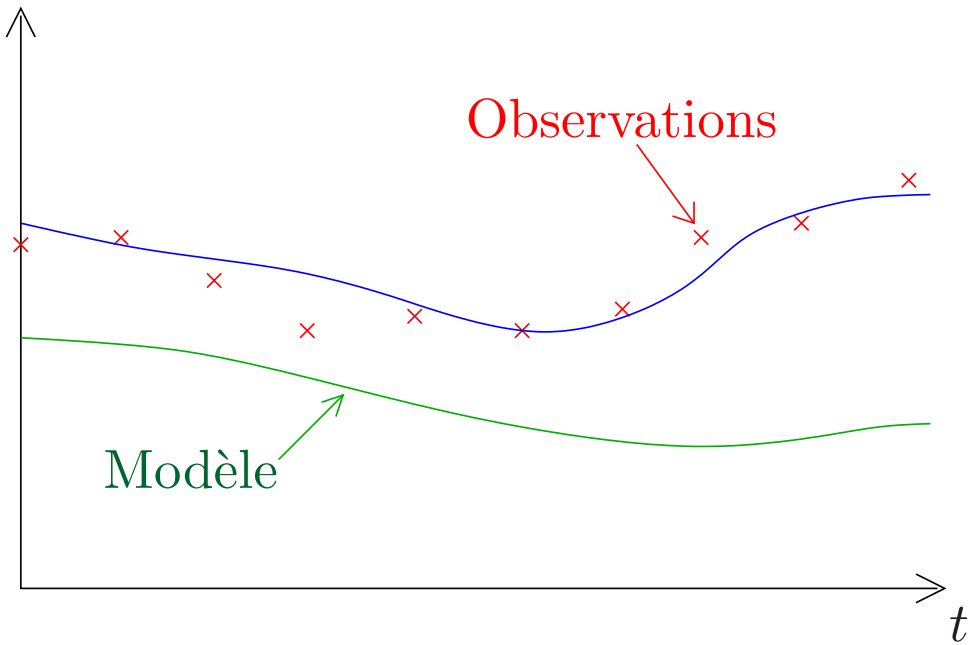
Réunion ANR IODISSEE, Nice

12 Juin 2012

Plan

1. Qu'est-ce que l'assimilation de données ?
2. Méthode variationnelle 4D-VAR (adjoint)
3. Estimation de paramètres
4. Différentiation automatique

Assimilation de données



combinaison

modèle + observations

$\Downarrow$

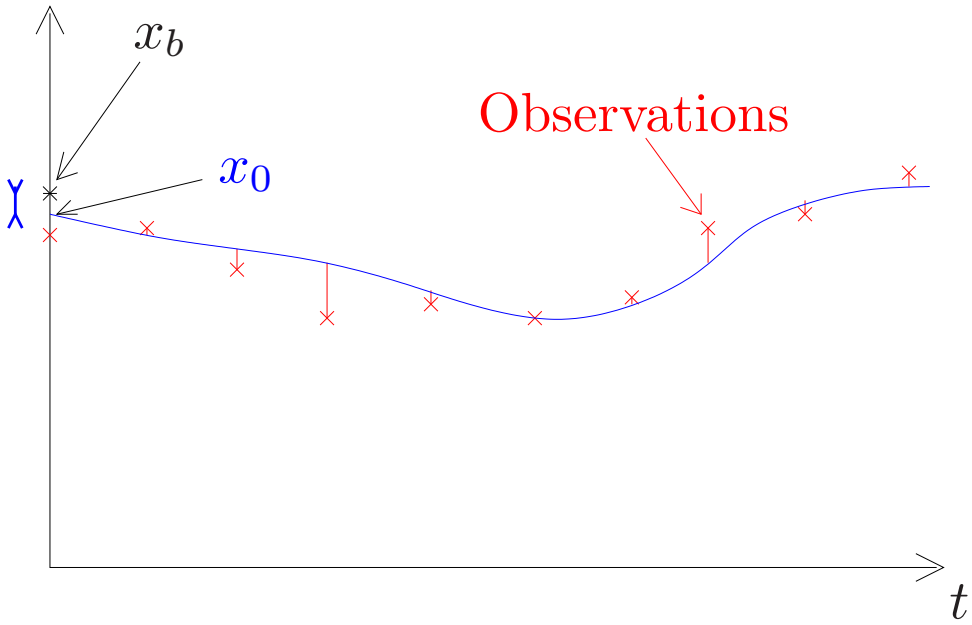
estimation de la condition initiale

état de l'écoulement

- Équations non-linéaires
- Non reproductibilité (unicité d'une trajectoire)
- Condition initiale et conditions aux limites ? ?

Comment identifier ces conditions ?

4D-VAR



$$\begin{cases} \dfrac{dx}{dt} = F(x), \\ \\ x(0) = x_0, \end{cases}$$

$(t_i)_{0 \leq i \leq n}$, les instants où des observations $x_{obs}(t_i)$ sont disponibles, H_i les opérateurs d'observation et R_i leur matrice de covariance d'erreur.

$$\begin{aligned} J(x_0) &= \frac{1}{2}(x_0 - x_b)^T B^{-1}(x_0 - x_b) \\ &+ \frac{1}{2}\sum_{i=0}^{n} \left(x_{obs}(t_i) - H_i(x(t_i))\right)^T R_i^{-1} \left(x_{obs}(t_i) - H_i(x(t_i))\right) \end{aligned}$$

4D-VAR

Difficultés :

La dépendance de J par rapport au contrôle x_0 est désormais implicite ($x(t_i)$).

Comment dépendent les termes $x(t_i)$ en fonction de la condition initiale x_0 ?

Comment calculer le gradient de J ?

Méthode des différences finies : généralement exclue pour des raisons de dimension et coût de calcul (le calcul de J en un point nécessite déjà la résolution du modèle).

Formulation lagrangienne

On réécrit le problème de minimisation de la fonction coût J sous la contrainte que x soit une solution de l'équation du modèle, en introduisant un multiplicateur de Lagrange p :

$$\mathcal{L}(x_0, x, p) = J(x_0) + \int_0^T \left\langle p, \frac{dx}{dt} - F(x) \right\rangle dt$$

Dans un cadre linéaire (modèle linéaire) : Si (x_0^*, x^*, p^*) est un point-selle de $\mathcal{L}$, alors x_0^* est solution du problème de minimisation de départ.

Point-selle (x_0^*, x^*, p^*) :

$$\mathcal{L}(x_0^*, x^*, p) \leq \mathcal{L}(x_0^*, x^*, p^*) \leq \mathcal{L}(x_0, x, p^*), \quad \forall x_0, x, p.$$

Maximum par rapport à p

$$\mathcal{L}(x_0, x, p) = J(x_0) + \int_0^T \left\langle p, \frac{dx}{dt} - F(x) \right\rangle dt$$

$$\frac{\partial \mathcal{L}}{\partial p} = 0 \quad \Longleftrightarrow \quad \frac{dx^*}{dt} = F(x^*)$$

On obtient le modèle direct, i.e. x^* est une trajectoire du modèle.

Minimum par rapport à x

$$\mathcal{L}(x_0, x, p) = J(x_0) + \int_0^T \left\langle p, \frac{dx}{dt} - F(x) \right\rangle dt$$

$$= \frac{1}{2}(x_0 - x_b)^T B^{-1}(x_0 - x_b) + \frac{1}{2}\sum_{i=0}^{n} \left(x_{obs}(t_i) - H_i x(t_i)\right)^T R_i^{-1} \left(x_{obs}(t_i) - H_i x(t_i)\right)$$

$$+ \int_0^T \left[\left\langle -\frac{dp}{dt}, x \right\rangle + \langle -p, F(x)\rangle\right] dt + \langle p(T), x(T)\rangle - \langle p(0), x_0\rangle$$

$$\frac{\partial \mathcal{L}}{\partial x} = 0 \quad \Longleftrightarrow \quad -\frac{dp^*}{dt} = \left[\frac{\partial F}{\partial x}(x^*)\right]^T p^* - \sum_{i=0}^{n} H_i^T R_i^{-1} \left(H_i x^*(t_i) - x_{obs}(t_i)\right) \delta_{t_i}(t)$$

Équation adjointe (p : état adjoint), rétrograde en temps, avec la condition finale $p^*(T) = 0$.

Minimum par rapport à x_0

En utilisant la dernière formulation :

$$\frac{\partial \mathcal{L}}{\partial x_0} = 0 \quad \Longleftrightarrow \quad B^{-1}(x_0^* - x_b) - p^*(0) = 0$$

Lorsque la contrainte de modèle est satisfaite, $\mathcal{L}(x_0, x, p) = J(x_0)$, donc

$$\frac{\partial \mathcal{L}}{\partial x_0} = \nabla J(x_0),$$

donc

$$\nabla J(x_0) = B^{-1}(x_0^* - x_b) - p^*(0).$$

Cela donne en une seule résolution du modèle adjoint le gradient de la fonction coût par rapport à la condition initiale.

Système d'optimalité

$$\text{Modèle direct : } \begin{cases} \dfrac{dx}{dt} = F(x) \\ x(0) = x_0 \end{cases}$$

$$\text{Modèle adjoint : } \begin{cases} -\dfrac{dp}{dt} = \left[\dfrac{\partial F}{\partial x}\right]^T p - \displaystyle\sum_{i=0}^{n} H_i^T R_i^{-1}(H_i(x(t_i)) - x_{obs}(t_i)) \\ p(T) = 0 \end{cases}$$

À l'optimum, le gradient de J est nul : $x_0 = x_b + Bp(0)$.

Algorithme 4D-VAR

$$\begin{cases} x_{n-1}(0) & \overset{\text{modèle direct}}{\longrightarrow} & x_{n-1}(t) & \overset{\text{modèle adjoint}}{\longrightarrow} & p_{n-1}(t) \\ & & & & \downarrow \\ x_n(0) & \overset{\text{algorithme de descente}}{\longleftarrow} & \nabla J_{n-1} & = & \nabla J(x_{n-1}(0)) \end{cases}$$

Chaque itération dans l'algorithme de minimisation de J ne coûte *que* 2 résolutions de systèmes d'EDP dans $\mathbb{R}^n$, l'une directe (modèle direct), l'autre rétrograde (modèle adjoint).

Difficultés de mise en œuvre

Discrétisation et contrôle :

$$\begin{array}{ccc} \text{Modèle direct continu} & \longrightarrow & \text{Modèle direct discret} \\ \downarrow & & \downarrow \\ \text{Adjoint continu} & \overset{FAUX}{\longrightarrow} & \text{Adjoint du modèle discret} \\ \downarrow & & \downarrow \\ \text{Système d'optimalité} & & \text{Système d'optimalité discret} \end{array}$$

Le système d'optimalité discret résout le problème direct discret, et le gradient par l'adjoint est bien le gradient discret.

Dérivation du code adjoint :

- Code direct compliqué → dérivation automatique
- Linéaire tangent : dérivé du code direct en mode direct
- Adjoint : dérivé du code direct en mode adjoint

Estimation de paramètres

On suppose dans cette partie que le modèle dépend de certains paramètres u, a priori inconnus.

L'équation du modèle est désormais la suivante :

$$\begin{cases} \dfrac{dx}{dt} = F(x, u), \\ x(0) = x_0. \end{cases}$$

On souhaite désormais identifier le jeu de paramètres u (en plus de la condition initiale x_0) qui minimise l'écart aux données :

$$\begin{aligned} J(x_0, u) &= \frac{1}{2}(x_0 - x_b)^T B^{-1}(x_0 - x_b) + \frac{1}{2}(u - u_b)^T Q^{-1}(u - u_b) \\ &+ \frac{1}{2}\sum_{i=0}^{n} \left(x_{obs}(t_i) - H_i(x(t_i))\right)^T R_i^{-1} \left(x_{obs}(t_i) - H_i(x(t_i))\right) \end{aligned}$$

Remarque : on peut supposer que x_0 est donné (ou connu) et ne chercher le minimum de J que par rapport à u.

Estimation de paramètres

L'algorithme 4D-VAR fonctionne de la même façon, en ajoutant u comme varible de contrôle (en plus de x_0) :

$$\mathcal{L}\left(x_0, u; x, p\right) = J(x_0, u) + \int_0^T \left\langle p, \frac{dx}{dt} - F(x, u) \right\rangle dt$$

Maximum par rapport à p :

$$\frac{\partial \mathcal{L}}{\partial p} = 0 \quad \Longleftrightarrow \quad \frac{dx^*}{dt} = F(x^*, u^*)$$

On obtient le modèle direct, i.e. x^* est une trajectoire du modèle avec les paramètres u^*.

Estimation de paramètres

Minimum par rapport à x :

$$\mathcal{L}(x_0, u, x, p) = J(x_0) + \int_0^T \left\langle p, \frac{dx}{dt} - F(x,u) \right\rangle dt$$

$$= \frac{1}{2}(x_0 - x_b)^T B^{-1}(x_0 - x_b) + \frac{1}{2}(u - u_b)^T Q^{-1}(u - u_b)$$

$$+ \frac{1}{2}\sum_{i=0}^{n} \left(x_{obs}(t_i) - H_i x(t_i)\right)^T R_i^{-1} \left(x_{obs}(t_i) - H_i x(t_i)\right)$$

$$+ \int_0^T \left[\left\langle -\frac{dp}{dt}, x \right\rangle + \langle -p, F(x,u)\rangle\right] dt + \langle p(T), x(T)\rangle - \langle p(0), x_0\rangle$$

$$\frac{\partial \mathcal{L}}{\partial x} = 0 \quad \Longleftrightarrow \quad -\frac{dp^*}{dt} = \left[\frac{\partial F}{\partial x}(x^*, u^*)\right]^T p^* - \sum_{i=0}^{n} H_i^T R_i^{-1}\left(H_i x^*(t_i) - x_{obs}(t_i)\right)\delta_{t_i}(t)$$

Remarque : On obtient la même équation adjointe (p : état adjoint), rétrograde en temps, avec la condition finale $p^*(T) = 0$.

Estimation de paramètres

En utilisant la dernière formulation du Lagrangien :

$$\frac{\partial \mathcal{L}}{\partial x_0} = B^{-1}(x_0^* - x_b) - p^*(0) = 0,$$

$$\frac{\partial \mathcal{L}}{\partial u} = Q^{-1}(u^* - u_b) - \left[\frac{\partial F}{\partial u}(x^*, u^*)\right]^T p^* = 0.$$

Lorsque la contrainte de modèle est satisfaite, $\mathcal{L}(x_0, u, x, p) = J(x_0, u)$, donc

$$\frac{\partial \mathcal{L}}{\partial x_0} = \frac{\partial J}{\partial x_0}, \qquad \frac{\partial \mathcal{L}}{\partial u} = \frac{\partial J}{\partial u}.$$

Estimation de paramètres

On obtient donc :

$$\frac{\partial J}{\partial x_0}(x_0, u) = B^{-1}(x_0 - x_b) - p(0),$$

$$\frac{\partial J}{\partial u}(x_0, u) = Q^{-1}(u - u_b) - \left[\frac{\partial F}{\partial u}(x, u)\right]^T p.$$

Cela donne toujours en une seule résolution du modèle adjoint le gradient de la fonction coût par rapport à la condition initiale et par rapport aux paramètres.

À l'optimum :

$$x_0^* = x_b + B\,p(0) \qquad u^* = u_b + Q\left[\frac{\partial F}{\partial u}(x, u)\right]^T p \quad \text{(à chaque instant)}$$

Méthode De L'Adjoint

Codage de l'adjoint

Généralement, on travaille localement sur le code direct, ligne par ligne si possible. Les règles sont :

- L'adjoint d'une séquence d'instructions est la séquence inverse des instructions adjointes.
- L'entrée d'une routine devient la sortie de la routine adjointe et réciproquement.
- Le codage de l'adjoint se fait au niveau le plus bas des routines, sur le plus petit bloc d'instructions qui décrive un opérateur linéaire avec des cœfficients connus. L'adjoint est alors la transposée de cet opérateur.
- Chaque variable modifiée fait partie des entrées du code adjoint, sauf la toute première fois qu'elle est définie.
- Chaque variable d'entrée fait partie des sorties du code adjoint sauf si c'est la toute dernière fois qu'elle est utilisée dans le code direct.
- La commande adjointe de la fin d'utilisation d'une variable est sa définition à 0.

Exemples

Utilisation d'une nouvelle variable :

Dans le code direct, on passe de (a) à (a, b). Dans le code adjoint, on passera donc de (a, b) à (a). La variable b deviendra indéfinie, et aucune instruction particulière n'est nécessaire. Si b est utilisée ultérieurement, il faudra la ré-initialiser.

Exemples

Fin d'utilisation d'une variable :

On passe de (a, b) à (a) dans le code direct, généralement de façon implicite après la dernière utilisation de b. Cela peut s'écrire mathématiquement avec un opérateur linéaire de projection :

$$(a) = (1 \quad 0) \begin{pmatrix} a \\ b \end{pmatrix} \tag{1}$$

et en transposant, on obtient

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} (a). \tag{2}$$

ce qui se traduit par

$$\mathtt{b} = \mathtt{0}. \tag{3}$$

Si on oublie cette instruction, la variable b sera mal initialisée $\rightarrow$ erreurs si elle est utilisée plus tard dans le code.

Exemples

Réassignement d'une variable :

Si le code direct comprend la ligne

$$\texttt{a} = \texttt{b}, \tag{4}$$

et que l'espace des entrées est (b) et l'espace de sortie (a), l'opérateur linéaire est simplement

$$(a) = (1)(b) \tag{5}$$

donc l'adjoint est

$$(b) = (1)(a) \tag{6}$$

donc

$$\texttt{b} = \texttt{a}. \tag{7}$$

Exemples

Mais si b est utilisée plus tard dans le code, l'espace des sorties est (a, b) et donc l'opérateur linéaire est

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} (b) \tag{8}$$

et l'adjoint est donc

$$(b) = (1 \quad 1) \begin{pmatrix} a \\ b \end{pmatrix} \tag{9}$$

et le code adjoint est

$$\mathtt{b} = \mathtt{b} + \mathtt{a}. \tag{10}$$

Exemples

Fin de d'utilisation d'une variable :

La meilleure façon de se rappeler quand une variable est utilisée pour la dernière fois est de toujours supposer qu'une variable est utilisée plus tard, et d'assigner la valeur 0 aux variables adjointes quand elles sont définies.

Si le code direct est

$$\mathtt{a} = \mathtt{s} * \mathtt{b} + \mathtt{t} * \mathtt{c}, \tag{11}$$

où s et t sont des constantes, le code adjoint est :

```
a = 0  ! when a is first defined in the adjoint code
b = 0  ! when b is first defined in the adjoint code
c = 0  ! when c is first defined in the adjoint code
.....
b = b + s * a
c = c + t * a.
```
(12)

Exemples

Si l'une des variables a, b, c est une entrée d'une routine adjointe, sa valeur initiale est définie en dehors de la routine et elle doit garder sa valeur d'entrée.

Il n'y a aucun problème d'indéfinition de a dans une instruction du type

$$\texttt{a} = \texttt{s} * \texttt{a} + \texttt{t} * \texttt{c}, \tag{13}$$

et l'adjoint est

$$\begin{array}{l} \texttt{c} = \texttt{0} \ \ \texttt{! when c is first defined in the adjoint code} \\ \ldots\ldots \\ \texttt{a} = \texttt{s} * \texttt{a} \ \ \texttt{! not a} = \texttt{a} + \texttt{s} * \texttt{a} \\ \texttt{c} = \texttt{c} + \texttt{t} * \texttt{a} \end{array} \tag{14}$$

car a est à la fois une entrée et une sortie.

Exemples

Tests conditionnels :

Une instruction du type

```
if (a > 0) then
  a = 2 * a
endif
```
(15)

définit une fonction non linéaire de a, ce qui est *interdit* dans la phase de construction de l'adjoint.

Le problème est donc souvent la linéarisation du code, et non l'adjoint.

Exemples

Exemple concret général :

$$x = az + by^2 + 3f(z) + c \tag{16}$$

où x, y et z sont les variables actives, a, b et c sont des variables passives, et f est une fonction (dérivable).
Le modèle linéaire tangent est

$$\delta x = a\delta z + 2by\delta y + 3\delta z f'(z). \tag{17}$$

Il faudra donc stocker les variables directes dès lors qu'elles interviennent de façon non linéaire.
L'écriture sous forme matricielle est

$$\begin{pmatrix} \delta x \\ \delta y \\ \delta z \end{pmatrix} = \begin{pmatrix} 0 & 2by & a + 3f'(z) \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \delta x \\ \delta y \\ \delta z \end{pmatrix} \tag{18}$$

Exemples

Le système linéaire adjoint est

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 2by & 1 & 0 \\ a + 3f'(z) & 0 & 1 \end{pmatrix} \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{pmatrix} \tag{19}$$

et l'adjoint est, par transposition,

$$\begin{cases} \tilde{z} = \tilde{z} + (a + 3f'(z))\,\tilde{x}, \\ \tilde{y} = \tilde{y} + 2by\tilde{x}, \\ \tilde{x} = 0. \end{cases} \tag{20}$$

Exemples

Difficultés réelles :

- Non dérivabilité d'une fonction : soit on simplifie le code direct pour le rendre dérivable, soit on donne une valeur arbitraire à la dérivée.

Exemple : une fonction d'interpolation sur une grille est dérivable partout sauf sur la grille. Malgré tout, il est facile d'attribuer une valeur à la dérivée.

- Non linéarités intrinsèques du code direct : les variables directes sont requises dans les codes tangent et adjoint. Il existe plusieurs solutions pour obtenir ces valeurs :

1. stocker toutes les valeurs en mémoire ;
2. écrire toutes les valeurs sur le disque ;
3. recalculer les valeurs au fur et à mesure ;
4. écrire une partie des valeurs sur le disque et recalculer les valeurs intermédiaires au fur et à mesure ;
5. idem en interpolant les valeurs intermédiaires non stockées.

Résolution d'un système linéaire

On suppose que le code direct résout

$$x \in \mathbb{R}^n \mapsto y = J(x) \in \mathbb{R}.$$

Ici, x sont les paramètres d'entrée (vecteur), et y est la sortie (scalaire), le coût correspondant à x.

Code linéaire tangent : $x_d \in \mathbb{R}^n$ est la variable dérivée de x en entrée, et $y_d \in \mathbb{R}$ est la variable dérivée de y en sortie.

$$y = J(x) \quad \Rightarrow \quad y_d = J'(x)x_d.$$

(dérivation classique) Dans le code tangent, si on fournit une direction de dérivation x_d, on récupère la dérivée directionnelle y_d.

Résolution d'un système linéaire

Code adjoint : on utilise la conservation du produit scalaire

$$\langle u, Av \rangle = \langle A^T u, v \rangle$$

Soit $y_b \in \mathbb{R}$ (vecteur adjoint de y, correspondant à y_d). Alors

$$\langle y_d, y_b \rangle = \langle J'(x)x_d, y_b \rangle = \langle x_d, [J'(x)]^T y_b \rangle$$

et on définit naturellement $x_b = [J'(x)]^T y_b$ le vecteur adjoint de x correspondant à x_d.

Ainsi, si on fixe $y_b = 1$ dans le code adjoint, on obtient en sortie (échange entrées/sorties entre l'adjoint et le tangent/direct) $x_b = \nabla J(x) \in \mathbb{R}^n$.

Résolution d'un système linéaire

On considère un système linéaire $Au = b$ à résoudre, où A et b sont une matrice et un vecteur dépendant des paramètres d'entrée, et u est la solution du système linéaire, dépendant également des paramètres d'entrée.

$$Au = b$$

devient en mode tangent :

$$A_d u + A u_d = b_d$$

où A_d et b_d sont les variables dérivées correspondant à A et b (variables d'entrée) et u_d est la variable dérivée de sortie.

u_d est alors solution du système linéaire suivant :

$$Au_d = (b_d - A_d u)$$

Résolution d'un système linéaire

En mode adjoint, le produit scalaire des variables tangentes/adjointes est conservé entre l'entrée et la sortie :

$$\langle u_d, u_b \rangle = \langle (A_d; b_d), (A_b, b_b) \rangle = \langle A_d, A_b \rangle + \langle b_d, b_b \rangle$$

donc en remplaçant u_d par $A^{-1}(b_d - A_d u)$:

$$\langle A^{-1}(b_d - A_d u), u_b \rangle = \langle A_d, A_b \rangle + \langle b_d, b_b \rangle$$

En posant $A_d = 0$,

$$\langle A^{-1} b_d, u_b \rangle = \langle b_d, b_b \rangle$$

donc par identification,

$$b_b = A^{-T} u_b$$

ou encore b_b est solution du système linéaire

$$A^T b_b = u_b$$

Résolution d'un système linéaire

De même, en posant $b_d = 0$, on obtient

$$\langle A_d, A_b \rangle = -\langle A^{-1} A_d u, u_b \rangle = -\langle A_d u, A^{-T} u_b \rangle = -\langle A_d u, b_b \rangle = -\langle A_d, b_b \otimes u \rangle$$

En effet,

$$\langle Au, b \rangle = \sum_i \sum_j A_{ij} u_j b_i = \sum_{i,j} A_{ij} [b \otimes u]_{ij}$$

Donc par identification,

$$A_b = -b_b \otimes u$$