Différentiation automatique
La méthodologie graphique de Diffedge

John Masse, Appedge, François Ollivier, CNRS

Les simulations numériques sont un outil de base pour la recherche académique et industrielle. Des outils informatiques, développés au fil du temps, constituent un patrimoine pour des communautés de chercheurs et d'ingénieurs, tant pour leur coût de développement, que pour la confiance acquise dans leurs résultats après des années de pratique.

Ces logiciels sont donc souvent réemployés pour résoudre des problèmes nouveaux, parmi ceux-ci : optimiser le comportement d'un système en minimisant ou maximisant un objectif, ou lui faire atteindre une valeur cible, ce qui revient mathématiquement à résoudre un système d'équations.

Il est alors nécessaire d'évaluer les dérivées des valeurs retournées, par rapport à certains paramètres, par exemple pour savoir comment la pression maximale dans la tuyère d'une fusée à l'allumage, calculée par un programme de dimensionnement, varie en fonction des débits d'ergols liquides. Les possibilités d'applications sont vastes. Outre la météorologie ou les industries aéronautique et spatiale, on peut citer l'ingénierie financière depuis la crise de 2008, ou l'intelligence artificielle. Mentionnons aussi l'identification paramétrique, délicate en biologie où les mesures sont pauvres.

Nous décrivons quelques techniques, leurs limitations et quelques solutions disponibles pour améliorer la fiabilité des résultats et la souplesse d'utilisation.

1. Différences finies et différentiation symbolique.

La méthode la plus immédiate est d'évaluer la dérivée par différences finies, ce qui revient pour une sortie discrète de pas ε, à approcher $\mathrm{d}f/\mathrm{d}x$ par $(f(x+\varepsilon)-f(x-\varepsilon))/2\varepsilon$. Cette approche suppose de choisir une valeur de la variation ε satisfaisante : assez petite pour que l'influence des dérivées d'ordre supérieur soit négligeable, assez grande pour que les erreurs d'arrondi ne dégradent pas la précision. Faute de disposer d'une méthode alternative pour évaluer la qualité du résultat, le choix est incertain.

Si le programme est une séquence de fonctions élémentaires +, −, × ou ÷, ou même de fonctions plus complexes, exp, log, cos, sin, etc., il est possible de le transformer en une formule mathématique, que l'on va pouvoir dériver formellement, puis retraduire en un programme. Les sorties y_1, ..., y_m du programme s'obtiennent à partir des arguments x_1, ..., x_n, en calculant de proche en proche des valeurs intermédiaires $w_i=f_i(w_k,w_l)$, à partir de valeurs déjà connues. Deux stratégies peuvent être retenues : l'une consiste à calculer dans le même ordre les $\mathrm{d}w_i/\mathrm{d}x_j$; l'autre, à calculer dans l'ordre inverse les $\mathrm{d}y_i/\mathrm{d}w_j$. Pour la première, le nombre de calculs est proportionnel au nombre d'arguments par rapport auxquels on dérive, pour la seconde au nombre de sorties. Entre ces deux extrêmes, il existe une grande variété de choix et trouver une solution optimale est connu pour être

un problème «NP-complet», une classe réputée difficile[1]. En pratique, on dispose de logiciels efficaces, utilisant l'une ou l'autre stratégie, et susceptibles de dériver des programmes en C, en Fortran ou en d'autres langages.

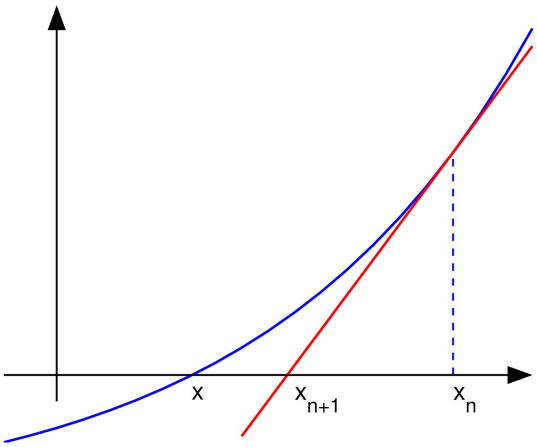
2. Au delà de la théorie élémentaire.

La plupart des programmes comportent aussi des boucles et des clauses conditionnelles. Les fonctions qu'ils approximent peuvent être, par exemple, solutions d'un système d'équations algébriques ou différentielles, deux cas qui couvrent un large spectre d'applications.

Pour les équations algébriques, la méthode de Newton est classique, et nécessite d'itérer une boucle informatique, jusqu'à ce que la valeur de la fonction soit assez proche de zéro. Une solution consiste, si la boucle correspond à une fonction f, à dériver la fonction f^i, où i est le nombre exact d'itération. Mais dans le cas extrême où la valeur initiale par défaut y_0, est proche du résultat, la boucle retourne cette constante y_0, qui ne dépend pas des arguments. Les dérivées seront donc nulles ! On commence à avoir une bonne approximation en passant au moins 2 fois dans la boucle. (Voir encadré 1). Des méthodes empiriques consistent à accroître artificiellement le nombre d'itération, ce qui augmente la complexité et peut avoir des conséquences imprévisibles si la boucle sert à tout autre chose.

Encadré 1. Méthode de Newton – Raphson

Découverte indépendamment par Isaac Newton (1642-1727) et Joseph Raphson (ca 1648-ca 1715), elle méthode consiste à résoudre l'équation $f(x)=0$ en approchant la courbe par sa tangente en $x=x_0$. On obtient ainsi une nouvelle valeur x_1 et on itère le processus jusqu'à atteindre la précision souhaitée. Lorsque la valeur initiale x_0 est assez proche de la racine, $|x-x_1| \approx |x-x_0|^2$. Alors, le nombre de décimales correctes obtenues double à chaque itération.



Kurt Hensel (1861-1941) a étendu cette méthode à d'autres situations. Si on connaît un nombre solution d'une équation modulo p^n (le reste de leurs division par p^n est le même), où p est premier, on peut en déduire sa valeur modulo p^{2n}, ce qui est très utilisé pour la factorisation de polynômes. On peut aussi calculer une solution en séries d'une équation algébrique en passant d'une approximation à l'ordre n à une approximation à l'ordre $2n$.

En 2001, Alexandre Sedoglavic a étendu la méthode de Newton au calcul des solutions en séries des systèmes d'équations différentielles. C'est la meilleure méthode connue pour de très grands ordres. Dans la gamme utile en pratique, la méthode relaxée de Joris van der Hoeven est néanmoins plus rapide, en différant astucieusement certaines opérations pour les effectuer plus vite, en bloc, au moment opportun pour mutualiser certaines opérations.

Signalons qu'avec David Harvey, celui-ci a récemment produit le meilleur algorithme pour multiplier de très grands nombres entiers… à partir de 20 milliards de milliards de milliards.

1 Voir *La combinatoire, hier et aujourd'hui*, Entretien avec Pierre Duchet, par Édouard Thomas
http://www.tangente-mag.com/article.php?id=4268

Une solution formellement plus élégante et plus précise permet d'exprimer les dérivées d'une racine à partir de la racine elle-même et d'une formule classique d'algèbre différentielle : si x est solution de $P(x,\alpha)=0$, alors $\partial P/\partial x\, dx + \partial P/\partial \alpha\, d\alpha = 0$ et $dx/d\alpha = -\ (\partial P/\partial \alpha)/(\partial P/\partial x)$.
Le cas d'un système d'équations différentielles est plus complexe. Un intégrateur numérique calcule la valeur des fonctions en une succession de points, dont la distance, le «pas d'intégration», doit être plus réduite si le système devient «raide», c'est-à-dire si la valeur de la dérivée peut changer brutalement. La dérivation automatique standard est alors à la peine, si le pas varie fortement en fonction du paramètre par rapport auquel on dérive. Une meilleure solution consiste à exprimer les dérivées comme solutions d'un nouveau système différentiel. (Voir encadré 2.)

Encadré 2. Folklore mathématique

Dans un manuscrit posthume, rédigé vers 1840, C.G.J. Jacobi (1804-1851) donne sans démonstration les équations satisfaites par les dérivées des solutions d'un système différentiel par rapport à des paramètres. L'un des plus grands mathématicien du XIX^{e} siècle, Jacobi était inspiré par ses échanges avec son frère Moritz, physicien, inventeur du premier moteur électrique fonctionnel, et appuya l'entrée de la recherche expérimentale à l'université.

Cette formule réapparaît dans un article de Joseph Ritt (1893-1951), publié en 1919, dans le cas particulier d'une équation d'ordre 1 : $f(x', x, \alpha) = 0$, $x(0)= c(\alpha)$. La dérivée $dx/d\alpha$ est solution de $\partial f/\partial x'\ (dx/d\alpha)' + \partial f/\partial x\ dx/d\alpha + \partial f/\partial \alpha = 0$, avec la condition initiale $dx(0) = dc/d\alpha$. Il en attribue l'idée à l'astronome Forest Ray Moulton (1872-1952), rencontré sur les champs de tir d'Aberdeen, où Oswald Veblen (1880-1960) avait réunit un groupe de mathématiciens pour produire des tables de tir durant la première guerre mondiale. Ritt avait préalablement travaillé au Naval Observatory de Washington, pour soutenir sa famille, tout en poursuivant ses études. Il a fondé l'algèbre différentielle et développé des algorithmes de calcul reposant sur la notion d'ensemble caractéristique.

Sous l'impulsion de Wu Wenjun (1919-2017), mathématicien chinois qui découvrit les possibilités de l'informatique en étant affecté dans une usine d'ordinateurs durant la révolution culturelle, les méthodes de Ritt devinrent des outils de base du calcul symbolique et de la démonstration automatique. Dans le cas des équations algébriques, le mathématicien chinois Zhū Shìjié (1270-1330) avait développé une technique semblable aux ensembles caractéristique dans son *Miroir de jade des quatre inconnues* (1303). À cette époque, on calculait avec des bâtonnets de jade placés sur un échiquier.

Enfin, de nombreux systèmes incluent des phénomènes discontinus : chocs, rebonds, fermeture d'un circuit électrique… Supposons que l'on veuille intégrer la fonction $f(t) = 0$, si $t < \alpha$ et 1, si $t \geq \alpha$. Son inégrale est 0, si $t < \alpha$ et $t-\alpha$, si $t \geq \alpha$ et sa dérivée par rapport à α est -1, si $t \geq \alpha$. Mais on ne dispose d'aucun moyen général pour dériver par rapport à un paramètre α qui apparaît dans une condition logique, et la différentiation automatique risque d'échouer dans un tel cas, si elle se contente de différentier les expressions correspondant à chacune des branches.

Ces limitations sont marginales, mais peu prévisibles. L'idéal serait de connaître le rôle de chaque partie d'un programme, ce qui est mis à profit par quelques logiciel, mais suppose un code très structuré, avec des normes de documentation précises permettant une analyse automatique.

3. Méthodologie graphique de Diffedge en Matlab Simulink

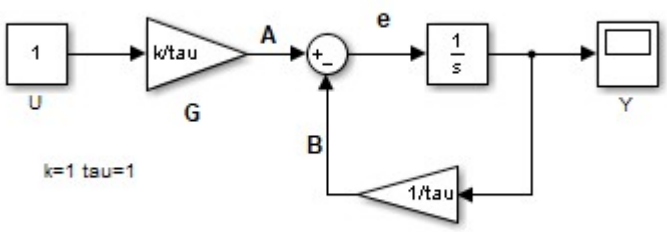
Matlab est un logiciel dédié au calcul numérique, qui compte près de 4 millions d'utilisateurs. Simulink, son interface graphique de schémas-blocs, permet de simuler des représentations mathématiques hybrides : représentation d'état, fonctions de transferts, associant une entrée à une sortie pour un signal continu ou discret, etc. Diffedge tire profit du flux d'information causal (représenté par une flèche orientée) entre les blocs pour construire, via un «flux dérivé», le modèle calculant les dérivées par rapport à un ou plusieurs paramètres du modèle. Ainsi le résultat produit par Diffedge est un nouveau modèle Simulink.

Le modèle dérivé conserve la connaissance physique que l'on a sur chacun des blocs et permet d'obtenir la dérivée en tout point du modèle en offrant la possibilité de continuer à profiter pleinement des possibilités de connexion de Simulink avec le monde du temps réel, comme par exemple avec les plateformes de prototypage Arduino ou les nano-cartes Raspberry. En ce sens, Diffedge renouvelle l'optimisation et l'identification paramétrique en ligne des procédés : maintenance prédictive, détection de défaillance, où embarquer le gradient d'un modèle est souvent nécessaire et compliqué.

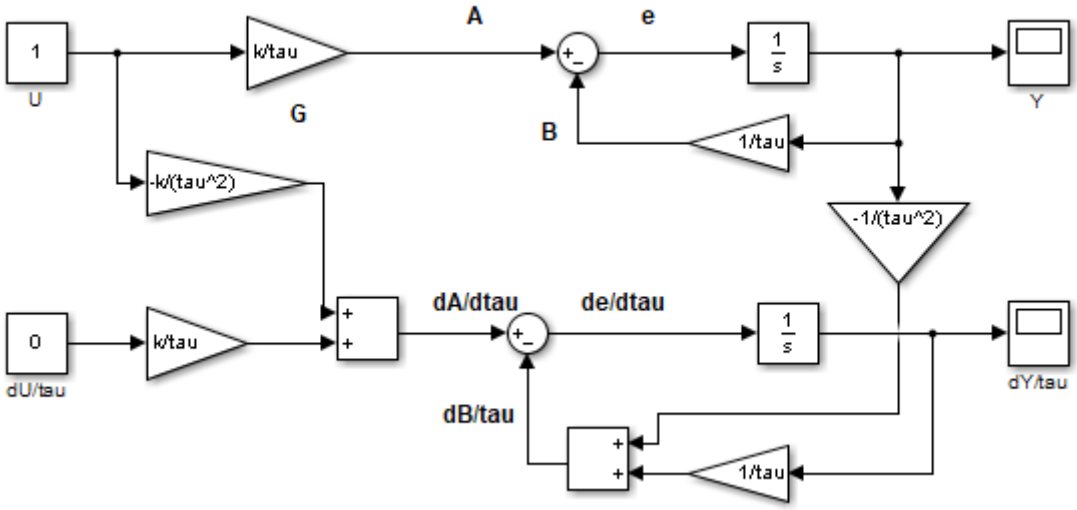
Sans Diffedge, le calcul est fastidieux, même dans le cas simple où l'on peut évaluer analytiquement le schéma bloc. Dans le calcul symbolique de Heaviside (1850-1925), l'opérateur s représente d/dt et l'on s'autorise des simplifications algébriques : un modèle qui s'écrit $\dot{y}=\frac{-y+k*u}{\tau}$, donne la fonction de transfert $Y=\frac{k}{1+\tau s}U$. Il faut ensuite calculer la dérivée $\frac{dY}{d\tau}=\frac{-ks}{(1+\tau s)^2}U$, puis enfin la réécrire dans Simulink. Et l'on perd la représentation physique du procédé.

Dans le cas ou l'on souhaite obtenir ou manipuler rapidement les équations d un modèle Simulink, la boite à outil « BlockImporter » de Maple est une solution séduisante. La méthode originale de Diffedge est basée sur des règles de différentiation graphique applicables automatiquement, que nous illustrons sur le modèle précédent.

Celui-ci est décrit par la figure ci-contre. Pour calculer la dérivée de la sortie Y par rapport à τ, le principe est le suivant. La première règle consiste à dupliquer le modèle originel autant fois qu'il y a de paramètres par rapport auxquels dériver. Ce modèle dit « augmenté », voir figure ci-dessous, est parcouru par le flux dérivé dU/dτ. On sait que U ne dépend pas de τ, donc dU/dτ =0. Ensuite, pour le gain entre entre les point A et U où G= k/τ, puisque le flux est causal, on peut appliquer la règle de dérivation des fonctions composées $\frac{dA}{d\tau}=\frac{dG}{d\tau}U+G\frac{dU}{d\tau}=\frac{-k}{\tau^2}U+\frac{k}{\tau}\frac{dU}{d\tau}$ qui utilise simultanément les flux du modèle originel (U) et augmenté (dU/dτ). On notera que l'on retrouve les blocs contenant ces opérations devant le point dA/dτ de la figure. Lorsqu'un bloc linéaire par rapport ses entrées ne dépend pas explicitement du paramètre à dériver, on le laisse tel quel dans le modèle augmenté. Par exemple pour le bloc soustraction $e=A-B$, on obtient $\frac{de}{d\tau}=\frac{dA}{d\tau}-\frac{dB}{d\tau}$. On fait de même pour tous les blocs jusqu'à la

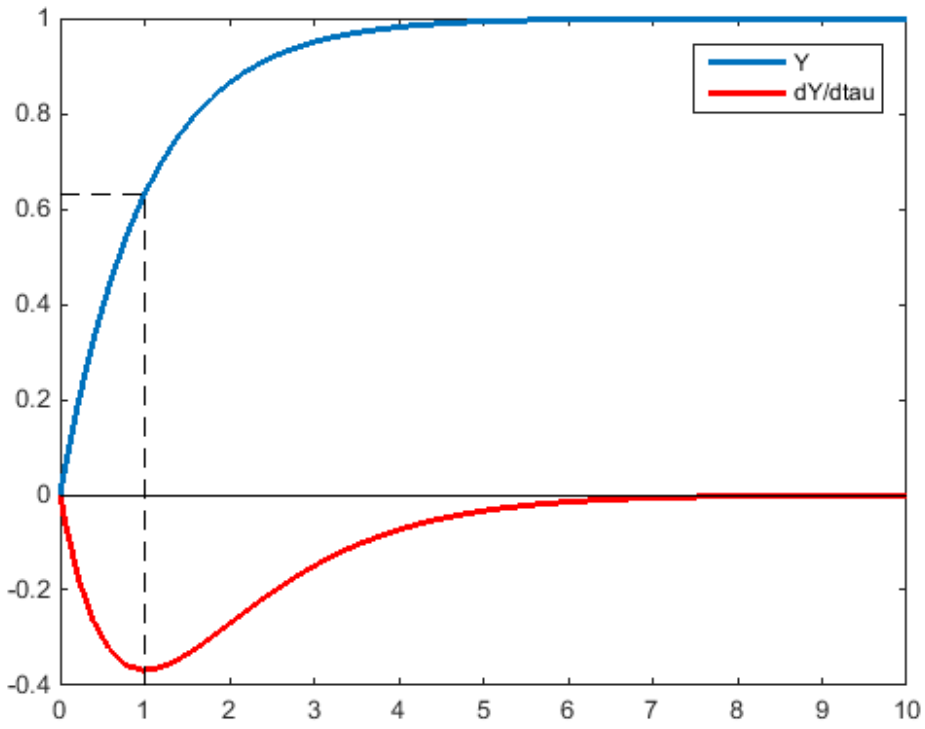


sortie *Y*. Avec le même principe, dans le cas de la dérivée par rapport à *k*, seul le premier bloc est à modifier. Le reste du modèle augmenté ne change pas !



On peut même itérer l'opération sur le modèle dérivé pour obtenir des dérivées d'ordre supérieur.

La connaissance de la dérivée d*Y*/dτ permet de définir le meilleur instant pour calculer la constante de temps τ, là ou |d*Y*/dτ| est maximale.



Références

Laurent Hascoet, «La Différentiation Algorithmique ou un éloge de la paresse», *Blog Le Monde, Binaire*, 9 mai 2016. https://www.lemonde.fr/blog/binaire/2016/05/09/la-differentiation-algorithmique-ou-comment-se-passer-des-ingenieurs/

Jean-Pierre Dussault, «La différentiation automatique et son utilisation en modélisation», *RAIRO-Oper. Res.* 42 (2008) 141–155. http://www.numdam.org/article/RO_2008__42_2_141_0.pdf

Documentation de Diffedge sur le site d'Appedge : https://www.appedge.com/francais/fr_developpement.html

J. Masse, C. Masse et F. Ollivier, Automatic differentiation of hybrid models Illustrated by Diffedge Graphic Methodology. (Survey) https://hal.archives-ouvertes.fr/hal-01536730