-

TRATTORIA 2015

COMPTE RENDU DE LA TABLE RONDE

CODES RAPIDES

	Nom	et Affiliation
Rédigé par	Malik CHAMI	Laboratoire d'Océanographie de Villefranche Université Pierre et Marie Curie, Villefranche sur mer
Et	Céline Cornet	Laboratoire d'Optique Atmosphérique Université de Lille Villeneuve d'Ascq

1. Description de la table ronde

Synthèse de la présentation préparée par les animateurs et organisation de la discussion

La table ronde a débuté par un bref rappel des différents codes et méthodes de transfert radiatif pour le système océan-atmosphère :

- modèles utilisés en couleur de l'océan incluant une surface océanique et une couche atmosphérique en ciel clair. Ces modèles sont assez performants en temps de calcul mais ne prennent pas en compte les propriétés optiques des hydrosols.
- modèles utilisés en couleur de l'océan comportant une couche océanique et une couche atmosphérique. Ces modèles sont plus complets et précis mais il y a souvent une dégradation du temps de calcul. La version vectorielle de ces codes n'est pas disponible à la communauté.
- modèle atmosphérique permettant le calcul des luminances en présence de différents constituants atmosphérique (gaz, aérosols et nuages). Dans l'infra-rouge et les micro-ondes, il existe des modèles paramétrés rapides (ex : RTTOV) mais dès que la diffusion doit être prise en compte (spectre visible), le temps de calcul des modèles utilisés devient long.

Nous avons ensuite découpé la discussion en 3 axes. Le premier axe concernait la rapidité des codes en lien avec les méthodes numériques/techniques de résolution de l'équation de transfert radiatif. Le second axe concernait la rapidité des codes selon les applications et besoins des utilisateurs. Enfin, nous avons terminé par une discussion sur la mise à disposition et la valorisation des codes rapides validés et existants à la communauté.

2. Participants

Nombre, et dans la mesure du possible, affiliations et thèmes d'expertise.
Nombre : 30 personnes
Affiliation / Laboratoire : LOV, LOA, LOCEAN, CNES, Thales, Noveltis, Météo-France, LOG, Hygeos, ONERA, LAMP, Cap Gemini, LAPLACE (univ. Toulouse), Alyotech, Icare, CS-SI
Thèmes d'expertise : Atmosphère, océan, surfaces continentales…

3. Compte-rendu des discussions

Axe 1 : rapidité en lien avec les méthodes numériques/techniques de résolution de l'équation de transfert radiatif

1-1 Qu'est ce qu'un code rapide et comment comparer des codes ?

En guise d'introduction, nous avons recueilli quelques exemples d'applications nécessitant des codes rapides avec des ordres de grandeur associés :

- Dans le cas de certaines applications, la simulation sur une bande large de 10-12µm ou 3-5µm d'images de 500X500 pixels doit pouvoir s'effectuer sur un PC standard en quelques minutes avec une bonne précision.
- La simulation d'une image 3MI (avec polarisation) 512 x 512 pixels à 670 nm avec prise en compte de la diffusion et de la BRDF prend de 7 à 14 heures sur un CPU.
- L'instrument IASI mesure 3millions de radiances par seconde, il est donc nécessaire de simuler avec RTTOV environ 67000 radiances / secondes. Le temps de calcul du code doit être inférieur à 216ms pour 4 pixels IASI.

La discussion s'est ensuite orientée vers la définition de codes « rapides » :

- pour des applications opérationnelles, ils doivent être ultra-rapides mais ils nécessitent généralement un compromis sur la précision des luminances simulées.
- pour des applications recherches, ils doivent être plus précis (ex. prise en compte de la diffusion) donc souvent plus lents. Les calculs doivent cependant toujours pouvoir s'effectuer dans des temps raisonnables.

Pour être en mesure de comparer et de répertorier les codes existants avec une indication de rapidité et de précision, il semble nécessaire de définir des cas tests « simples » (e.g., diffusion moléculaire uniquement) et représentatifs avec des entrées communes. Vu la grande variété de types de serveurs de calculs existants, le plus simple serait de pouvoir effectuer les calculs sur un serveur commun équipé des 2 environnements CPU et GPU, voire Xeon Phi / MIC (Many Integrated Core architecture). Le pôle de données Icare a été identifié à titre d'exemple comme pouvant être une plateforme possible pour ces exercices d'intercomparaison de codes.

Les résultats seraient comparés aux résultats d'un calcul de référence avec une précision optimale.

Plusieurs critères à comparer sont apparus, en plus de la **rapidité** et la **précision** : la **mémoire** utilisée apparaît aussi importante à répertorier ainsi que le **temps horloge** et pas seulement le **temps CPU**.

Nous avons aussi noté l'importance de refaire les mêmes simulations, par exemple tous les 2 ans, afin de cataloguer l'augmentation de la rapidité des codes lié à l'évolution des technologies.

Au delà des comparaisons en rapidité, cette base de codes documentés permettrait aussi aux utilisateurs de sélectionner le « bon » code en fonction de l'application que ces utilisateurs veulent en faire. Une base de codes documentés servirait aussi à établir une passerelle entre code « recherche » et code « opérationnel ».

1-2 Augmentation de la rapidité des codes grâce aux méthodes d'optimisation numériques ?

On rappelle que l'on parle ici d'optimisation à précision équivalente sur la sortie d'un calcul de transfert radiatif.

Différentes pistes d'améliorations suivant les codes ont été citées :

- Pour RTTOV, plusieurs méthodes d'optimisation sont utilisées, certaines méthodes avec perte de précision d'autre sans : utilisation de coefficients de régressions (perte de précision), approximation de la diffusion (perte de précision), parallélisation avec openMP et MPI (sans perte de précision), utilisation des composantes principales pour IASI (perte de précision).
- Pour les modèles permettant le calcul en atmosphère tri-dimensionnelle (codes 3D), il existe des pistes à explorer comme une approche stochastique avec réécriture de l'équation de transfert radiatif (ETR) en prenant en compte la moyenne et les effets sous-maille. Pour les codes utilisant Monte-Carlo, des méthodes permettent de s'affranchir du suivi des particules fictives à travers les mailles du modèles ce qui devrait permettre d'accélérer ces modèles 3D.
- Les modèles utilisant une méthode analytique pour résoudre l'ETR (ordonnées discrètes, ordres successifs de diffusion), des méthodes pour corriger les biais introduits par la troncature des fonctions de phase devraient permettre d'accélérer le temps de calcul de ces codes.
- Dans le cas de modèles utilisant les ordres successifs de diffusion, il apparaît possible de revisiter le code pour qu'il soit utilisable dans le cas de nuages sans temps de calcul trop important.
- Il faut aussi essayer d'utiliser au mieux la physique en travaillant sur la diffusion primaire, le régime asymptotique.
- Quoiqu'il en soit, il est important de garder à l'esprit le domaine de validité et d'utilisation d'un code.

Avant de vouloir accélérer un code, il est important de faire un « profiling » informatique du code afin d'identifier quelles parties du code peuvent être optimisées d'un point de vue informatique pour accélérer le temps de calcul. Des méthodes de profiling pourraient être indiquées sur le serveur dédié aux codes rapides. Il est apparu important également de bien définir l'interface d'un code « recherche » au moment de son développement pour le rendre « scriptable » et « intégrable » dans un autre code à des fins d'opérationnalité ultérieure.

1-3. Augmentation de la rapidité des codes grâce aux moyens techniques ?

En 2008 (Trattoria 1), il avait été abordé le transfert de certains calculs de transfert radiatif effectués traditionnellement par les CPU vers les GPU (Graphical Processing Units).

Des travaux ont été menés avec différents codes :

- Une version du code 3D SHDOM a été portée sur GPU par Alyotech. Le travail n'est pas trivial. Le gain de temps est de l'ordre d'un facteur 6-7.
- Un code de Monte-Carlo sphérique (SMARTG) a aussi été porté sur GPU par la société Hygeos. Le facteur gagné est de l'ordre de 100-200.
- RTTOV-7 : une expérience sur GPU a été faite au Space Science and Engineering Center (SSEC) à l'université de Wisconsin en 2010 : un gain d'un facteur 180 par rapport à une mono-processeur a été constaté.
- Matisse (Onera) est en cours de portage, trop tôt pour donner des détails à l'heure actuel.

Le portage sur GPU permet donc un gain de temps important mais nécessite un travail de reformulation du code non négligeable. Un des inconvénients concerne les problèmes de mémoire qui est limitée sur GPU. Il faut donc penser au gain théorique mais aussi au gain pratique. Selon les méthodes de calcul, le gain en performance est plus ou moins important après portage sur GPU. D'autres architectures comme le Xeon-Phi doivent également être considérées (Voir annexe).

Les méthodes analytiques apparaissent transférables avec l'aide d'informaticiens spécialisés mais il faut faire attention à ce que le code n'échappe pas au scientifique et que celui-ci soit capable de le modifier et de l'améliorer.

Une question se pose dans la communauté : comment s'y prendre pour transférer un code vers les nouvelles architectures de type GPU, où trouver les infos ou l'informaticien compétent ?
La communauté GPU grossit, on peut donc trouver des informations sur internet et sur les sites des vendeurs de cartes graphiques. Il serait intéressant qu'un guide simple qui résumerait la procédure pour faciliter le portage d'un code vers les nouvelles architectures soit édité, sur la base du retour d'expérience des codes qui ont déjà connu un tel portage, pour accroître le potentiel des codes « recherches » pour les rendre utilisables à des fins opérationnelles.

Axe 2: rapidité en lien avec les applications et besoins des utilisateurs

Dans cette deuxième partie, nous avons essayé d'identifier les besoins en terme de codes rapides selon les applications qui en sont faites.
Dans le futur, il apparaît intéressant d'avoir **un code rapide couplé et complet nuages/aérosols/hydrosols** mais ce n'est pour l'instant pas une priorité.
Il existe déjà Matisse qui est un code construit en concaténant les banques de données, sol, atmosphère, nuages. La Terre est considérée comme sphérique. Le code Matisse est disponible et évolutif selon les besoins de la communauté.

Le besoin pour **un code Sphérique** apparaît lorsque l'on travaille aux hautes latitudes, avec des géostationnaires ou encore en astrophysique. Il existe :
- SMARTG qui est un code Monte Carlo GPU précis et rapide (1sec pour 10^8 photons ; quelques minutes pour 0.5% de précision avec plusieurs angles). Ce code tient compte du couplage océan/atmosphère, de la sphéricité et de la polarisation. Il devrait être mis à disposition à la communauté par le CNES dans le futur.
- VLIDORT, code qui permet de corriger des effets de sphéricité de la Terre.

Il apparaît à l'heure actuelle illusoire de disposer de **codes 3D** opérationnel. Il est cependant toujours utile de faire baisser leur temps de calcul afin de pouvoir multiplier les études de cas et de sensibilités.

Il existe aussi un besoin en océanographie de **codes vectoriels** rapides qui soient précis et disponibles à la communauté. Dans le cas des besoins dans le domaine « atmosphère », de tels codes vectoriels sont déjà disponibles même si dans le régime diffusif, ils ne sont pas encore performants.

Il existe aussi des besoins de développer des codes rapides et précis permettant de simuler les capteurs actifs actuels et futurs : **Lidar et Radar Doppler.** Cela ne semble a priori pas une priorité à l'heure actuelle

Il existe le besoin **de codes hyperspectraux** couvrant de l'UV à l'IR (voire les micro-ondes) utiles pour des instruments tels que GOSAT qui couvre du visible à l'IR thermique ou OCO2 (du visible au NIR). Actuellement, l'extension de 4A jusqu'à l'UV-Vis est en cours : ce code (pseudo-sphérique) encapsule VLIDORT (vectoriel). Pour l'instant ce n'est pas encore assez rapide mais c'est en cours d'accélération.

Dans le cadre de l'**assimilation des données**, il est impératif de disposer de codes rapides aussi bien pour le calcul direct que pour les jacobiens: il existe déjà RTTOV qui évoluera en fonction des besoins des nouveaux modèles de prévision météorologique et des nouvelles techniques d'assimilation. Il existe aussi CRTM de la NOAA ou OSS (code payant américain).

Pour **l'inversion des données**, dans l'infrarouge, les codes RTTOV disponible pour la communauté ou FORLI développé à l'ULB (non disponible) sont utilisés pour les inversions IASI.

Dans le domaine visible, il est nécessaire de prendre en compte la diffusion, les codes deviennent trop lents pour que des méthodes d'inversion variationnelles soient utilisées dans un contexte opérationnel. **Une recommandation est donc d'augmenter la rapidité des codes de diffusion (ex : Ordres successifs de diffusion en facilitant le portage sur GPU ou autres)**

Il apparaît aussi nécessaire de disposer d'un code assez rapide couvrant l'ensemble du domaine spectral afin de pouvoir mener des études en amont sur la synergie des instruments de type MSG (Meteosat) par exemple.

Axe 3: Mise à disposition des codes rapides validés et existants à la communauté.

Dans cette partie, nous avons repris les points importants des parties précédentes qui apparaissent dans la partie bilan et recommandations. Quelques pistes ont été discutées pour la diffusion des codes à la communauté : créer un portail wikipedia dédié et utiliser Github pour la dépose des codes sources, ou utilisation du pôle de données atmosphère (Icare) à l'instar de ce qui est fait actuellement pour la mise à disposition des codes du projet ARTDECO (TOSCA-CNES). Le pôle de données pourrait être une plateforme pertinente pour (i) donner accès aux codes de transfert radiatif, (ii) mise à disposition des données ancillaires qui permettent d'utiliser ces codes, (iii) faciliter les exercices d'intercomparaison de codes pour établir leur rapidité et précision.

4. Bilan et recommandations

Bilan sur une dizaine de lignes, et quelques recommandations (pour le CNES, mais aussi éventuellement pour la communauté).
Possibilité de distinguer les recommandations prioritaires (ou court terme) des autres.

Recommandation 1 : Intercomparaisons de codes

Il est apparu assez rapidement au cours de cette table ronde qu'il y avait un besoin d'inter-comparaisons de différents codes à partir de la définition de quelques cas tests représentatifs et ceux à partir d'un serveur commun à partir duquel les calculs pourraient être faits. Ce serveur commun devrait être équipé des environnements CPU et GPU ainsi que de l'architecture Xeon-Phi.

Les résultats seraient comparés aux résultats d'un calcul de référence avec une précision optimale. Plusieurs critères à comparer sont apparus, en plus de la rapidité et la **précision**, la **mémoire** utilisée apparaît aussi importante à répertorier ainsi que le **temps horloge** et pas seulement le **temps CPU**.

Nous avons aussi noté l'importance de refaire les mêmes simulations, par exemple tous les 2 ans, afin de cataloguer l'augmentation de la rapidité des codes lié à l'évolution des technologies.

Il semble important que les résultats des intercomparaisons, la documentation de ces codes, leurs applications et la mise à disposition de données ancillaires utilisées en entrée de ces codes puissent être mis sur un site public et facilement accessibles : wikipédia ? une structure nationale qui pourrait être le pôle atmosphère ?

Recommandation 2 : Portages des codes vers les nouvelles architectures/technologies

Afin d'aider les chercheurs qui veulent porter leurs codes vers les nouvelles technologies (GPU, Intel Xeon-Phi,...) pour accroitre de manière significative leur rapidité, plusieurs pistes ont été évoquées :

- écrire une feuille de route (ou un guide d'une procédure) à suivre
- avoir le soutien d'informaticiens qui peuvent être dans une super structure. Cependant, il y a besoin de former des ingénieurs et des scientifiques dans les laboratoires pour que l'expertise de leurs codes soit conservée.

Recommandation 3 : Codes à accélérer

Après avoir balayé différents codes de transfert radiatif, il est apparu important de développer un code rapide :

- Afin d'améliorer la synergie de différents instruments (par exemple sur le futur MetOP-SG), il apparaît important de développer un modèle utilisant le même moteur allant de l'uv-visible (ex : pour Sentinel) à l'infrarouge thermique (ex IASI-NG) et aux micro-ondes.
- Pour aller vers le développement d'algorithmes opérationnels à partir de capteurs visibles (ex : 3MI) utilisant des méthodes variationnelles pour l'inversion, il apparaît indispensable d'accélérer les codes 1D en diffusion. Des codes opérationnels permettraient de s'affranchir des tables de pré-calculs et d'améliorer ainsi la précision des paramètres géophysiques à estimer par inversion du signal satellitaire.

5. Annexes

5-1. Comparaison des technologies GPU et Xeon-phi

Les technologies GPU et Xeon-phi (également appelé MIC) sont des coprocesseurs visant à accélérer grandement la vitesse de certains calculs. L'un, le GPU est développé par les entreprises Nvidia ou ATI (ex : carte Kepler de Nvidia) et est dérivé des processeurs dédiés à l'affichage. Le second, le MIC a été lancé par Intel fin 2013 et cible le marché du calcul intensif. Ces coprocesseurs possèdent plusieurs caractéristiques similaires :

- Ce sont tous deux des cartes que l'on ajoute dans une machine via le bus PCI-express ;
- Chaque carte contient sa propre mémoire RAM non partagée avec celle du CPU ;
- Ces deux technologies ont une architecture massivement parallèle (plusieurs dizaines de cœurs) ;
- Ces coprocesseurs ont une faible consommation électrique, sont compatibles avec les OS classiques et peuvent être 5 à 10 fois plus puissants qu'un CPU (à génération équivalente).

Malgré tous ces points communs les architectures GPU/MIC restent fondamentalement différentes. Le tableau ci-dessous montre les principales divergences:

	GPU (Graphic Processing Unit)	**MIC (Many Integrated Cores)**
Nombres de cœurs	Grand nombre de cœurs « streaming processor » (1000 ou plus)	~ 60 cœurs (mais peut exécuter 240 threads ou plus)
Bande passante mémoire	~ 320 GB/s	~ 320 GB/s
Fréquence de base	~ 745 MHz	~ 1.1 GHz
Langages compatibles	Requière l'utilisation d'un langage spécifique (CUDA, openCL ou openACC)	Compatible avec les langages compilés courants (C/C++ et Fortran)
Parallélisation	Utilise des threads natifs GPU	Peut utiliser des API classiques comme openMP et pthread
Vectorisation SIMD (Single Instruction Multiple Data)	Plusieurs threads traitent des données adjacentes en mémoires	Le paradigme SIMD est exploité au sein d'un seul thread
Compatibilité CPU	Non compatible	Peut être compatible moyennant une nouvelle compilation
Portage du code	Il faut réécrire le code dans l'un des langages spécifiques au GPU (CUDA, openCL …)	Il faut effectuer des adaptations mais le code reste dans des langages courants (C/C++, Fortran)
Performances	Plus performant que le MIC	Moins performant (un déficit

	dans la plupart des cas	de vitesse d'au maximum 30% en mode natif par rapport au GPU)
Prix	Entrée de gamme ~ 2000€	Entrée de gamme ~ 2000€

Le graphique qui suit illustre de la légère supériorité du GPU dans l'exécution de certains calculs :

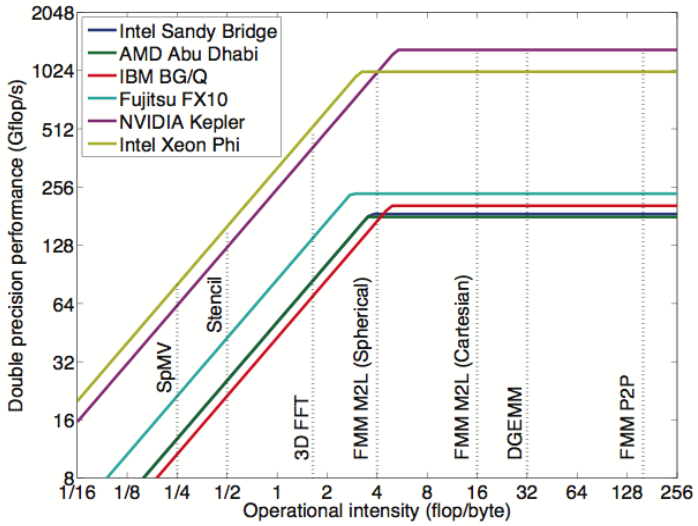

Figure 1 : modèle de performances le plus élevé pour les architectures modernes et indication de l'intensité opérationnelle d'algorithmes courants, notamment FFT et DGEMM (multiplication matricielle dense). La pente indique la limite de bande passante mémoire et l'horizontale indique la limite de calcul. (Lorena A. Barba and Rio Yokota , "How Will the Fast Multipole Method Fare in the Exascale Era?", SIAM News, Volume 46, Number 6, July/August 2013)

En plus des résultats obtenus sur des algorithmes courants, Thales a mené pour le CNES et en collaboration avec le laboratoire Vision Lab de l'université d'Anvers, durant l'année 2014, une étude/comparaison entre CPU, MIC et GPU sur des algorithmes de traitement d'image. Des facteurs d'accélérations ont pu être mesurés et comparés :

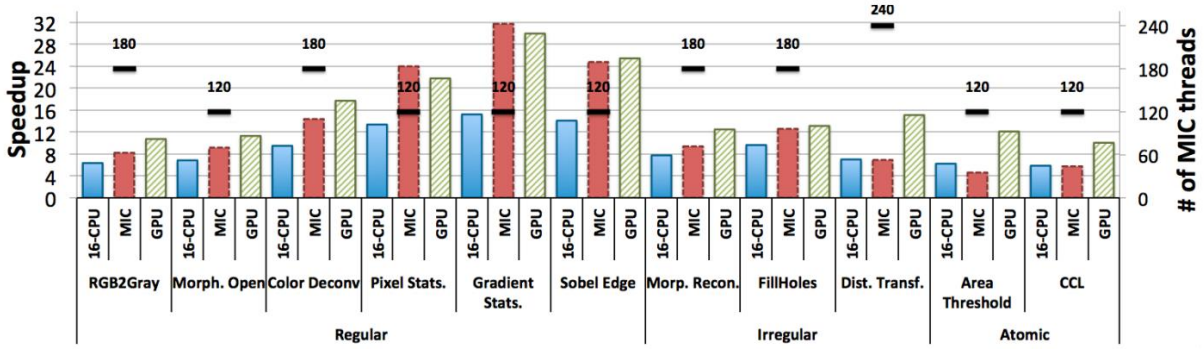

Figure 2: comparaison d'accélérations pour CPU (E5-2680), MIC (SE10) et GPU (K20) pour des opérations de traitement d'image spécifiques. Les opérations sont groupées en trois catégories : accès aux données normal, accès aux données irrégulier et opérations dépendant fortement des instructions atomiques (Teodoro, T. Kurc, J. Kong, L. Cooper, and J.Saltz, « Comparative Performance Analysis of Intel Xeon Phi, GPU and CPU, » arXiv :1311.0378 [cs], Nov. 2013.)

Ainsi le GPU montre les meilleurs résultats en termes de capacité de calcul et de speedup même si parfois le MIC fait mieux que rivaliser. Le coprocesseur choisi à la fin de l'étude fut un MIC. Ce choix se justifie par les performances plus que correctes du MIC, par une plus grande facilité dans le portage du code (pas de réécriture mais juste des adaptations) et par la compatibilité du code porté avec une architecture CPU classique. L'étude a laissé place au développement et les premiers résultats font part d'un gain entre *2 et *17 par rapport à la version CPU d'origine.

Les résultats et les chiffres donnés précédemment ne sont absolument pas figés puisque les architectures GPU et MIC évoluent très rapidement. Les deux technologies sont récentes (~10 ans pour le GPU et moins de 2 ans pour le MIC) et ont encore une belle marge de progression devant elles (en particulier pour le MIC). Le Xeon-phi V2 est annoncé pour mi 2015 et une nouvelle carte GPU Nvidia devrait voir le jour avant la fin de l'année. Des améliorations seront donc apportées dans les deux technologies :

Pour le Xeon-phi : _ Une version socket (suppression du goulot PCI-express)
_ Une puissante brute (*4) et bande passante améliorée
_ Intel annonce son intention de fusionner le Many-core et le Multi-core

Pour le GPU : _ Nouveau bus de communication (amélioration du goulot PCI-express)
_ Une puissante brute (*4) et bande passante améliorée

Des informations supplémentaires sur les technologies GPU/MIC sont présentes dans l'article suivant : O. Melet « GPGPU and MIC in accelerated cluster for remote sensed image processing software ».

•