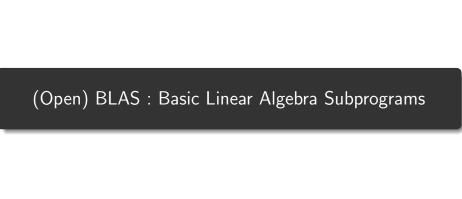
High Performance Computing et calcul parallèle

D. Cornu, UTINAM / Observatoire de Besancon

Master 2 - P2N & PICS

Automne 2019

Crédits et remerciement à Benoit Semelin, LERMA (OBSPM)



Introduction OpenBLAS

OpenBLAS est une **bilbiothèque Open Source** qui permet d'implémenter des opérations d'algèbre linéaire optimisées.

Elle inclut directement des opérations bas niveau (BLAS) et des opérations haut niveau (LAPACK).

Exemples:

Level 1:

Vecteurs or scalaires :

$$\mathsf{xAXPY}(...) \to y = \alpha x + y$$

Level 2:

Vecteurs × Matrices : xGEMV (...)

$$\rightarrow y = \alpha A x + \beta y$$

Level 3:

 $\mathsf{Matrices} \times \mathsf{Matrices}$:

$$\to C = \alpha o p(A) o p(B) + \beta C$$

Introduction OpenBLAS

OpenBLAS est une **bilbiothèque Open Source** qui permet d'implémenter des opérations d'algèbre linéaire optimisées.

Elle inclut directement des opérations bas niveau (BLAS) et des opérations haut niveau (LAPACK).

Exemples:

Level 1:Level 2:Level 3:Vecteurs or scalaires :Vecteurs \times Matrices :Matrices \times Matrices : \times AXPY(...) $\rightarrow y = \alpha x + y$ \times GEMV (...) \times GEMM(...) $\rightarrow y = \alpha Ax + \beta y$ $\rightarrow C = \alpha op(A)op(B) + \beta C$

OpenBLAS est automatiquement optimisée au moment de l'installation pour tirer pleinement parti du materiel présent. Utilisation de jeux d'instructions CPU, adaptation aux quantités de cache, ...

La bilbiothèque est également **automatiquement parallélisée** via OpenMP ou PThreads. Le nombre de coeurs utlisé est choisit automatique en fonction des niveaux de caches et de la surcharge induit par la parallélisation!

2/5

Installation

La dernière version de OpenBLAS peut être trouvée à : https://github.com/xianyi/OpenBLAS/

Vous pouvez cloner ou télécharger la bibliothèque puis l'installer avec la commande :

```
make USE_OPENMP=1
sudo make install
```

Attention : La compilation est très longue et gourmande en ressource !

Pour utiliser la librairie vous devez ajouter le path de son installation dans votre fichier .bashrc, ou bien préciser son emplacement au moment de la compilation :

```
gfortran -03 test.f90 -o test -I /opt/OpenBLAS/include/
-L/opt/OpenBLAS/lib -lopenblas -fopenmp
```

Le nombre de coeurs est alors définit via OpenMP avec export OMP_NUM_THREADS=N

Exemple classique : xGEMM

La routine **xGEMM** permet de réaliser une multiplication Matrice - Matrice avec une accumulation.

C'est l'opération la plus classique avec ce type de bibliothèque. Elle est notamment capable de remplacer très efficacement la fonction MATMUL pré-intégrée à Fortran

```
subroutine dgemm ( character TRANSA,
character TRANSB,
integer M, integer N, integer K,
double precision ALPHA,
double precision, dimension(lda,*) A, integer LDA,
double precision, dimension(ldb,*) B, integer LDB,
double precision BETA,
double precision, dimension(ldc,*) C, integer LDC)
```

La définition des parametres peut être trouvée a :

http://www.netlib.org/lapack/

Exercice:

Ecrivez un programme qui déclare trois matrices (A, B, et C) de dimension M=N=K. Faire la multiplication matricielle $C = A \times B$ avec DGEMM et avec MATMUL.

Comparez le temps de calcul pour un N asser grand sur un seul coeur. Vous pourrez utiliser les commandes de mesure de temps à l'intérieur du code vues dans l'exercice "loop_workshare" du cours OpenMP. Augmentez le

nombre de coeurs et observer le comportement de OpenBLAS en fonction de N en regardant la charge CPU de votre machine. Que constatez vous?