

Librairies pour le calcul numérique

000

Contexte

■ Matlab (/Octave, Scilab)

- Très bonnes fonctions calcul numérique

Mais ...

- Programmation non numérique laborieuse
 - Pas de structures de données (hash, map, graph, etc)
- Lent
- Non interactif

=> Uniquement restreint à réaliser des prototypes

001

Contexte

■ C++

- Vrai langage de programmation
 - Structures données avancées
- Rapide
- Interactif (Qt, OpenGL, ...)
- Standard en entreprise

Mais ...

- Pas de librairie numérique dans le standard (Matrices, vecteurs, ...)

=> Beaucoup de temps passé pour refaire
ce que fait un script Matlab
+ bugs

002

Librairies numériques

Il est possible de réécrire vos propres structures

Inversion matricielle, valeurs propres, matrices creuses, ...

Mais à éviter pour des vrais applications (sauf cas spécifique)

Préférer des librairies existantes

Testées, plus robustes et rapide qu'un code personnel

Une bonne librairie de calcul numérique, plusieurs mois/années
de développement

003

Calcul numérique

A l'origine
un langage est développé spécifiquement pour le calcul numérique

Fortran

(*FORmula TRANslating system*)

004

Fortran

Fortran : un des plus anciens langages (1954)
(C 1970)

A beaucoup évolué Fortran 66
Fortran 77
Fortran 90
Fortran 2008
Fortran 2015...

Autrefois très limité

Aujourd'hui langage objet avancé

- Surcharge d'opérateur
- Gestion mémoire dynamique
- Héritage
- polymorphisme
- structure données abstraites

005

Fortran

Langage très agréable pour calcul numérique

Calcul vectoriel en standard
Array slicing
Matrice en standard
Parallélisme aisé

```
INTEGER N = ...

REAL, DIMENSION(N,N) :: A
INTEGER, DIMENSION(N) :: v, w
DO i = 1, N
    w(i) = SUM ( A(i, : ) * v )
END DO
```

Ressemble à Matlab, + simple que C, C++, Java, ...

006

Avantage/limitation Fortran

- + Agréable pour le calcul numérique
- + Portable
- + Standard le plus robuste et répandu
- + Très rapide (souvent + rapide que C ou C++)
- + Mise en place sur les supercalculateurs

- Uniquement centré sur le calcul numérique
- Gestion texte difficile
- Pas de *logiciel* en Fortran avec GUI, interaction
- Pas de graphique

007

Fortan et Lapack

Librairie numérique de référence

- Structure de donnée matrice

BLAS

Basic Linear Algebra Subprograms

(opérations somme, multiplication matricielles)

- Algorithmes d'algèbre linéaire

Lapack

Linear Algebra **PACK**age

(inversion matrices, factorisation)

Lapack

Lapack à été pendant longtemps le standard "de facto" du calcul numérique

Matlab était une *interface* au dessus de Lapack

Mais ...

Blas et Lapack sont bas niveau. Peu agréable à utiliser

~1800 fichiers

shdw3.f	shdw4z	shwep1z	shwptr1	shwptf2z	shwptf3	shwptf4	shwptf5	shwptf6	shwptf7	shwptf8	shwptf9	shwptf10
shdw4.f	shdw5z	shwep2z	shwptr2	shwptf2z	shwptf3z	shwptf4z	shwptf5z	shwptf6z	shwptf7z	shwptf8z	shwptf9z	shwptf10z
shdw5.f	shdw6z	shwep3z	shwptr3	shwptf3z	shwptf4z	shwptf5z	shwptf6z	shwptf7z	shwptf8z	shwptf9z	shwptf10z	shwptf11z
shdw6.f	shdw7z	shwep4z	shwptr4	shwptf4z	shwptf5z	shwptf6z	shwptf7z	shwptf8z	shwptf9z	shwptf10z	shwptf11z	shwptf12z
shdw7.f	shdw8z	shwep5z	shwptr5	shwptf5z	shwptf6z	shwptf7z	shwptf8z	shwptf9z	shwptf10z	shwptf11z	shwptf12z	shwptf13z
shdw8.f	shdw9z	shwep6z	shwptr6	shwptf6z	shwptf7z	shwptf8z	shwptf9z	shwptf10z	shwptf11z	shwptf12z	shwptf13z	shwptf14z
shdw9.f	shdw10z	shwep7z	shwptr7	shwptf7z	shwptf8z	shwptf9z	shwptf10z	shwptf11z	shwptf12z	shwptf13z	shwptf14z	shwptf15z
shdw10.f	shdw11z	shwep8z	shwptr8	shwptf8z	shwptf9z	shwptf10z	shwptf11z	shwptf12z	shwptf13z	shwptf14z	shwptf15z	shwptf16z
shdw11.f	shdw12z	shwep9z	shwptr9	shwptf9z	shwptf10z	shwptf11z	shwptf12z	shwptf13z	shwptf14z	shwptf15z	shwptf16z	shwptf17z
shdw12.f	shdw13z	shwep10z	shwptr10	shwptf10z	shwptf11z	shwptf12z	shwptf13z	shwptf14z	shwptf15z	shwptf16z	shwptf17z	shwptf18z
shdw13.f	shdw14z	shwep11z	shwptr11	shwptf11z	shwptf12z	shwptf13z	shwptf14z	shwptf15z	shwptf16z	shwptf17z	shwptf18z	shwptf19z
shdw14.f	shdw15z	shwep12z	shwptr12	shwptf12z	shwptf13z	shwptf14z	shwptf15z	shwptf16z	shwptf17z	shwptf18z	shwptf19z	shwptf20z
shdw15.f	shdw16z	shwep13z	shwptr13	shwptf13z	shwptf14z	shwptf15z	shwptf16z	shwptf17z	shwptf18z	shwptf19z	shwptf20z	shwptf21z
shdw16.f	shdw17z	shwep14z	shwptr14	shwptf14z	shwptf15z	shwptf16z	shwptf17z	shwptf18z	shwptf19z	shwptf20z	shwptf21z	shwptf22z
shdw17.f	shdw18z	shwep15z	shwptr15	shwptf15z	shwptf16z	shwptf17z	shwptf18z	shwptf19z	shwptf20z	shwptf21z	shwptf22z	shwptf23z
shdw18.f	shdw19z	shwep16z	shwptr16	shwptf16z	shwptf17z	shwptf18z	shwptf19z	shwptf20z	shwptf21z	shwptf22z	shwptf23z	shwptf24z
shdw19.f	shdw20z	shwep17z	shwptr17	shwptf17z	shwptf18z	shwptf19z	shwptf20z	shwptf21z	shwptf22z	shwptf23z	shwptf24z	shwptf25z
shdw20.f	shdw21z	shwep18z	shwptr18	shwptf18z	shwptf19z	shwptf20z	shwptf21z	shwptf22z	shwptf23z	shwptf24z	shwptf25z	shwptf26z
shdw21.f	shdw22z	shwep19z	shwptr19	shwptf19z	shwptf20z	shwptf21z	shwptf22z	shwptf23z	shwptf24z	shwptf25z	shwptf26z	shwptf27z
shdw22.f	shdw23z	shwep20z	shwptr20	shwptf20z	shwptf21z	shwptf22z	shwptf23z	shwptf24z	shwptf25z	shwptf26z	shwptf27z	shwptf28z
shdw23.f	shdw24z	shwep21z	shwptr21	shwptf21z	shwptf22z	shwptf23z	shwptf24z	shwptf25z	shwptf26z	shwptf27z	shwptf28z	shwptf29z
shdw24.f	shdw25z	shwep22z	shwptr22	shwptf22z	shwptf23z	shwptf24z	shwptf25z	shwptf26z	shwptf27z	shwptf28z	shwptf29z	shwptf30z
shdw25.f	shdw26z	shwep23z	shwptr23	shwptf23z	shwptf24z	shwptf25z	shwptf26z	shwptf27z	shwptf28z	shwptf29z	shwptf30z	shwptf31z
shdw26.f	shdw27z	shwep24z	shwptr24	shwptf24z	shwptf25z	shwptf26z	shwptf27z	shwptf28z	shwptf29z	shwptf30z	shwptf31z	shwptf32z
shdw27.f	shdw28z	shwep25z	shwptr25	shwptf25z	shwptf26z	shwptf27z	shwptf28z	shwptf29z	shwptf30z	shwptf31z	shwptf32z	shwptf33z
shdw28.f	shdw29z	shwep26z	shwptr26	shwptf26z	shwptf27z	shwptf28z	shwptf29z	shwptf30z	shwptf31z	shwptf32z	shwptf33z	shwptf34z
shdw29.f	shdw30z	shwep27z	shwptr27	shwptf27z	shwptf28z	shwptf29z	shwptf30z	shwptf31z	shwptf32z	shwptf33z	shwptf34z	shwptf35z
shdw30.f												

```

DO I = 1, N-Q
  IF (I.EQ. 1) THEN
    DO J = 1, M
      PHANTOM(1) = ZERO
    END DO
    CALL ZUNBDS( P, N-P, Q, PHANTOM(1), 1, PHANTOM(P+1), 1,
      X11, LDx11, X21, LDx21, WORK(1ORBDS),
      LORBDGS, CHILINFO )
    CALL ZSCL( P, NEGONE, PHANTOM(1), 1 )
    CALL ZLARFGP( P, PHANTOM(1), PHANTOM(2), 1, TAUPI(1) )
    CALL ZLARFGP( N-P, PHANTOM(P+1), PHANTOM(P+2), 1, TAUPI(2) )
    THETA(1) = ATAN2( DBLE( PHANTOM(1) ), DBLE( PHANTOM(P+1) ) )
    C = COS( THETA(1) )
    S = SIN( THETA(1) )
    PHANTOM(1) = ONE
    PHANTOM(P+1) = ONE
    CALL ZLARF( 'L', P, Q, PHANTOM(1), 1, DCONJG(TAUPI(1)), X11,
      LDx11, WORK(LARF) )
    CALL ZLARF( 'L', N-P, Q, PHANTOM(P+1), 1, DCONJG(TAUPI(2)),
      X21, LDx21, WORK(LARF) )
  ELSE
    CALL ZUNBDS( P=1+I, N-P=1+I, Q=1+I, X11(I,1-1), 1,
      X21(I,1-1), 1, X11(I,I), LDx11, X21(I,I),
      LDx21, WORK(1ORBDS), LORBDGS, CHILINFO )
    CALL ZSCL( P=1+I, NEGONE, X11(I,1-1), 1 )
    CALL ZLARFGP( P=1+I, X21(I,1-1), X21(I+1,1-1), 1, TAUPI(I) )
    CALL ZLARFGP( N-P=1+I, X21(I,1-1), X21(I+1,1-1), 1,
      TAUPI(I) )
    THETA(1) = ATAN2( DBLE( X11(I,1-1) ), DBLE( X21(I,1-1) ) )
    C = COS( THETA(1) )
    S = SIN( THETA(1) )
    X11(I,1-1) = ONE
    X21(I,1-1) = ONE
  
```

Autres librairies

- Souvent une surcouche au dessus de Lapack
- Souvent spécialisé sur un domaine
 - Conteneur matriciel
 - Décomposition matrice pleine
 - Décomposition matrice creuse
 - Fitting non linéaire
 - ODE / PDE

Les grands acteurs: Vue d'ensemble

GPU <i>Pour la puissance de calcul</i> + Le plus performant - Codage long et difficile - Portabilité	Python <i>Remplace de plus en plus Fortran + Lapack</i> + Numpy + Interfaçage aisé avec graphique (Qt) et GPU - Par défaut plus lent que Fortran
Librairies C + Interfaçage direct sur un logiciel complet (Qt, drivers, OS, interaction, etc) - Apprentissage plus long - Pas de standard - Portabilité	Librairies C++ <i>Pour développer des logiciels</i>
Fortran <i>Pour le standard</i> + Lapack + Standard + Robuste et vérifié - Limité au calcul	Matlab / Octave / Scilab <i>Pour les prototypes</i> + Très simple d'utilisation - Limitées aux prototypes - Lenteur du code

Exemple de librairies

GSL: GNU Scientific Library

+ Librairie open source complète

- En C (utilisation fastidieuse)
- Moins robuste, vérifiée que Lapack
(lib issu de l'open source écrite par des informaticiens)

- Complex Numbers:
- Polynomials:
- Special Functions:
- Vectors and Matrices:
- Permutations:
- Linear Algebra:
- Eigensystems:
- Fast Fourier Transforms:
- Numerical Integration:
- Statistics:
- Monte Carlo Integration:
- Ordinary Differential Equations:
- Interpolation:
- Numerical Differentiation:
- Series Acceleration:
- Wavelet Transforms:
- Multidimensional Root-Finding:
- Least-Squares Fitting:
- Basis Splines:

```
int
gsl_schur_solve_equation_z(double ca, const gsl_matrix *A, gsl_complex *z,
                          double d1, double d2,
                          const gsl_vector_complex *b,
                          gsl_vector_complex *x, double *s, double *xnorm,
                          double smin)
{
    size_t N = A->nsize;
    double scale = 1.0;
    double bnorm;

    if (N == 1)
    {
        double cr, /* denominator */
              ci,
              cnorm; /* |c| */
        gsl_complex bval, c, xval, tmp;

        /* we have a 1-by-1 (complex) scalar system to solve */

        /* c = ca * a - z * d1 */
        cr = ca * gsl_matrix_get(A, 0, 0) - GSL_REAL(*z) * d1;
        ci = -GSL_IMAG(*z) * d1;
        cnorm = fabs(cr) + fabs(ci);

        if (cnorm < smin)
        {
            /* set c = smin * i */
            cr = smin;
            ci = 0.0;
            cnorm = smin;
        }
    }
}
```

012

Exemple de librairies

- Blitz++ Vecteur/Matrice dense.
Librairie très rapide
(utilisation métaprogrammation/template)
- Boost Conteneurs pour matrice
- Dlib Optimisation non linéaire, fitting
Simple d'utilisation
- IML++,
SparseLib++ Inversion matrices creuses
- ...

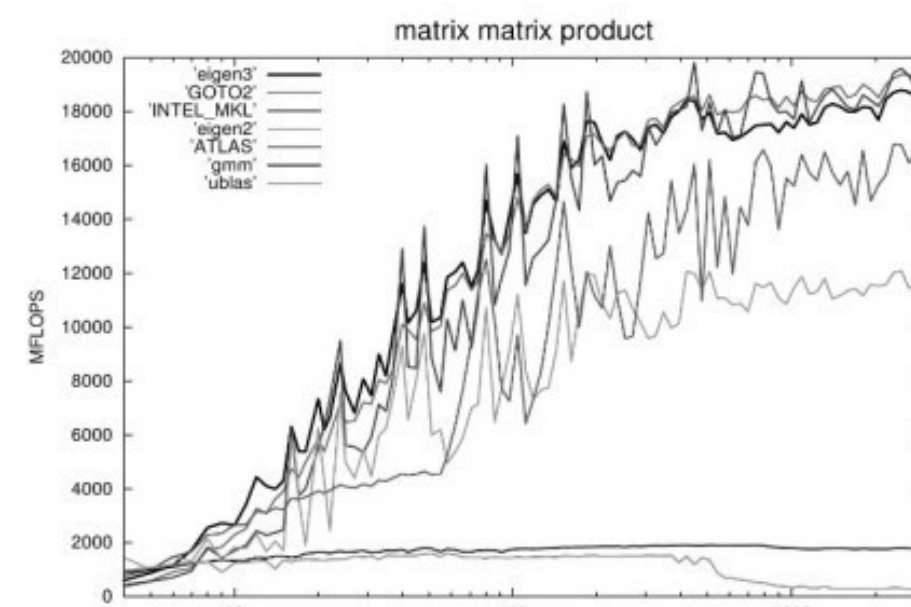
013

Eigen

■ Eigen

librairie C++ d'algèbre linéaire récente

- Conteneur matrice, vecteur, matrice creuses
Algorithme de résolution de systèmes et décomposition matricielles
- Repose sur les templates / métaprogrammation
Peut être plus rapide qu'un code C/Fortran équivalent
- Utilisation instructions vectorielles, SSE



014

Eigen: Utilisation

Support vecteur nD

```
Eigen::Array2d p0(1.0, 1.1);
Eigen::Array2d p1(0.4, -1.2);

std::cout << p0 + p1 << std::endl;
```

```
Eigen::ArrayXd p0(4);
p0[0]=1.0; p0[1]=2.2; p0[2]=4.7; p0[3]=-1.2;
p0=8.1*p0;

std::cout << p0 << std::endl;
```

015

Eigen: Utilisation

Support matrices

```
Eigen::Matrix2f M;  
M<<1.2,4.5,  
  -0.1,2.2;  
  
M(1,0)=-4.2;  
M=2.0*M;  
  
std::cout<<M<<std::endl;
```

```
2.4  9  
-8.4 4.4
```

016

Eigen: Utilisation

Support inversion de matrices

```
Eigen::Matrix3f A;  
A << 1,2,3, 4,5,6, 7,8,10;  
  
std::cout<<A.inverse()<<std::endl;
```

017

Eigen: Utilisation

Support résolution de système linéaire $Ax=b$

```
Eigen::Matrix3f A;  
Eigen::Vector3f b;  
A << 1,2,3, 4,5,6, 7,8,10;  
b << 3, 3, 4;  
std::cout << "Here is the matrix A:\n" << A << std::endl;  
std::cout << "Here is the vector b:\n" << b << std::endl;  
Eigen::Vector3f x = A.colPivHouseholderQr().solve(b);  
std::cout << "The solution is:\n" << x << std::endl;
```

Choix des approches

Decomposition	Method	Requirements on the matrix	Speed	Accuracy
PartialPivLU	partialPivLu()	Invertible	++	+
FullPivLU	fullPivLu()	None	-	+++
HouseholderQR	householderQr()	None	++	+
ColPivHouseholderQR	colPivHouseholderQr()	None	+	++
FullPivHouseholderQR	fullPivHouseholderQr()	None	-	+++
LLT	llt()	Positive definite	+++	+
LDLT	ldlt()	Positive or negative semidefinite	+++	++

018

Eigen: Utilisation

<http://eigen.tuxfamily.org/>

Support pour:

Décompositions: LU, QR, valeurs singulières, Cholesky, Shur
Matrices creuses
Transformation géométriques

019