

Confronto tra risolutori per sistemi lineari sparsi

Metodi del calcolo scientifico

Matamoros Ricardo 807450

r.matamorosaragon@campus.unimib.it

18 giugno 2019

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

L'obiettivo di questo progetto è quello di confrontare Matlab con diverse librerie open source, le quali forniscono L'Implementazione del metodo di Cholesky per risolvere sistemi lineari di matrici sparse definite positive. Il confronto prevede l'utilizzo di due sistemi operativi, i quali sono stati configurati per l'utilizzo delle librerie analizzate.

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

Linux Mint 19.1 Cinnamon

version Cinnamon: 4.0.8

version: Lenovo ideapad
Y700-15ISK

width: 64 bits

capabilities: smbios-2.8

dmi-2.8 smp vsyscall32

cpu

product: Intel(R) Core(TM)

i7-6700HQ CPU @

2.60GHz

vendor: Intel Corp.

memory

description: System

Memory size: 8GiB

memory storage : 80GB

Windows 10

version: Lenovo ideapad

Y700-15ISK

width: 64 bits

capabilities: smbios-2.8

dmi-2.8 smp vsyscall32

cpu

product: Intel(R) Core(TM)

i7-6700HQ CPU @

2.60GHz

vendor: Intel Corp.

memory

description: System

Memory size: 8GiB

memory storage : 420 GB

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

MATLAB è una piattaforma di programmazione progettata per il calcolo basato su matrice, permette di implementare efficientemente algoritmi della matematica computazionale.

MATLAB è ben documentato, supportato e ha una grande comunità.

MATLAB non è open-source e specialmente non è gratuito: il costo della licenza varia molto a seconda dell'uso. MATLAB è facile da imparare e da utilizzare, ma è anche un intero ambiente di programmazione, ed è questo che lo rende meno portabile rispetto ad altre scelte.

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

MATLAB integra la funzionalità per la risoluzione di sistemi lineari sparsi tramite la fattorizzazione di Cholesky. La quale viene eseguita attraverso l'istruzione $x = A \setminus B$ (operatore `mldivide`), MATLAB sceglie internamente il risolutore ottimale basato su matrici simmetriche, riducendo al minimo il tempo di calcolo.

Per le matrici sparse, la fattorizzazione di Cholesky viene utilizzata quando la matrice è quadrata, non è triangolare né permutata, è Hermitiana ed è diagonale definita positiva.

Scikit-Sparse (Python)

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

Scikit-Sparse è una libreria di script Scipy.Sparse open source per la manipolazione di matrici sparse che estende la libreria Scipy.

Scikit-Sparse è [documentato](#) anche se il livello di descrizione delle funzioni fornite non sia molto accurata. Sulla maggior parte delle distribuzioni Linux vengono offerti i file già compilati nella repository della distro. Per Windows, la compilazione di SuiteSparse / Cholmod richiede l'uso di qualche gestore delle dipendenze (Ubuntu Windows installer).

Scikit-Sparse (Python)

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

Scikit-Sparse consente di effettuare la Fattorizzazioni di Cholesky sia come LDL^T che LL^T .
Offre anche un'interfaccia comoda ed efficiente per l'utilizzo di questa decomposizione per risolvere i problemi del tipo $Ax = b$.

Eigen (C++)

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

Eigen è una libreria C ++ open source per l'algebra lineare.

Eigen presenta un'ampia [documentazione](#), cicli di rilascio stabili ed è utilizzato da un ampia quantità di progetti (ad esempio Google's Tensorflow).

Eigen, essendo una libreria di solo file .header, è abbastanza facile da includere in un progetto, e non richiede dipendenze aggiuntive.

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

Tutti i test sono stati condotti calcolando l'equazione $Ax = b$, dove b è stata scelta per contenere la soluzione esatta composta come $x_e = [1 \ 1 \ 1 \ 1 \ 1 \ \dots]$.

In altre parole, $b = A * x_e$.

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

Tutti le matrici in input sono definite positive e simmetriche, provengono dalla [SuiteSparse Matrix Collection](#). In particolare, sono state utilizzate le seguenti matrici:

- ex15
- cfd1
- shallow_water1
- cfd2
- parabolic_fem
- apache2
- StocF-1465
- Flan_1565
- G3_circuit

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

Per ogni libreria e piattaforma, sono stati misurati tre parametri per il confronto:

- Errore relativo
- Tempo per la risoluzione del sistema
- Memoria utilizzata

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

```
mat =load(strcat(path, name_matrix));  
A=mat.Problem.A;  
n=size(A,1);  
xe=ones(n,1);  
b=A*xe;  
x=A\b;  
error = norm(x-xe)/norm(xe);
```

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

```
from sksparse.cholmod import cholesky
import numpy as np
from scipy.io import mmread

def chol(A):
    factor = cholesky(A)
    xe=[1]*A.shape[0]
    b=A.dot(xe)
    x =factor(b)
    error=np.linalg.norm(x-xe)/np.linalg.norm(xe)

A = mmread(path+matrix).tocsc()
chol(A)
```

Eigen

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

```
#include <Eigen/Sparse>
#include <Eigen/SparseCholesky>
typedef Eigen::SparseMatrix<double, Eigen::ColMajor> sparsa;
int main(int argc, char** argv){
    sparsa A;
    //open matrix
    loadMarket(A, matrix);
    int n=A.cols();
    Eigen::VectorXd b(n),x(n), xe(n);
    xe=VectorXd::Constant(A.rows(), 1);
    //Cholesky factorization of A
    Eigen::SimplicialCholesky<sparsa,Eigen::Lower> chol(A);
    b=A.selfadjointView<Eigen::Lower>() * xe;
    x =chol.solve(b);
    double relative_error = (x-xe).norm()/(xe).norm();
}
```

Introduzione

Sistemi Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

- Le matrici StocF-1465 e Flan_1565 falliscono in eigen.
- Le matrici StocF-1465 e Flan_1565 non erano computabili in linux (vincolo di memoria).
- Per rilevare la memoria sono stati utilizzati diversi strumenti.
- Le matrici sono state ordinate in modo crescente rispetto alla dimensione.

Introduzione

Sistemi Operativi

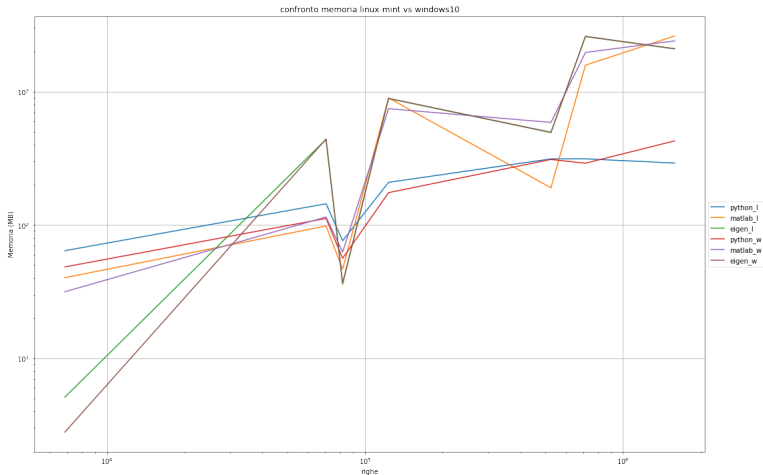
Librerie

Confronti

Listato Codice

Risultati

Conclusioni



Memoria per StocF-1465 & Flan_1565

Introduzione

Sistemi
Operativi

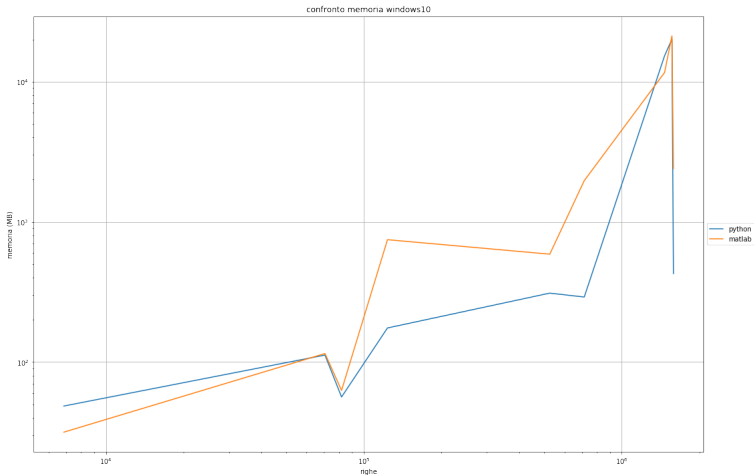
Librerie

Confronti

Listato Codice

Risultati

Conclusioni



Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

- Per le matrici di dimensione inferiore a 10K Eigen si comporta in modo migliore rispetto a Matlab e Scikit-sparse sia in Linux che Windows.
- La libreria più stabile è scikit-sparse.
- A livello di memoria è preferibile Scikit-sparse con 292 MB per G3_circuit.

Introduzione

Sistemi
Operativi

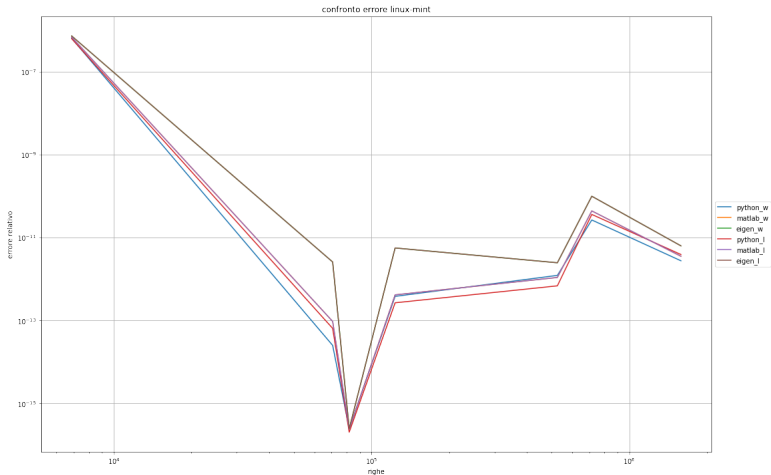
Librerie

Confronti

Listato Codice

Risultati

Conclusioni



Errore Relativo per StocF-1465 & Flan_1565

Introduzione

Sistemi
Operativi

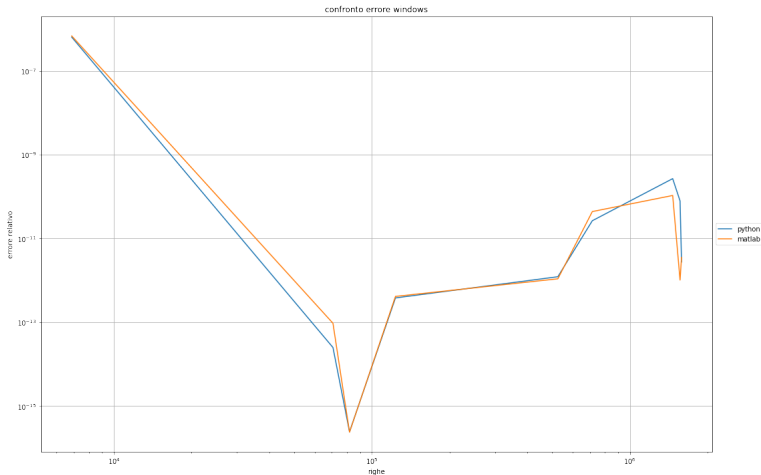
Librerie

Confronti

Listato Codice

Risultati

Conclusioni



Considerazioni sull'Errore relativo

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

- Complessivamente le librerie si comportano allo stesso modo sia in Linux che Windows.
- In Linux Mint Eigen presenta un errore relativo superiore di circa 1 ordine di grandezza.
- l'errore relativo più grande corrisponde alla matrice ex15, dovuto al mal condizionamento della matrice.

Introduzione

Sistemi
Operativi

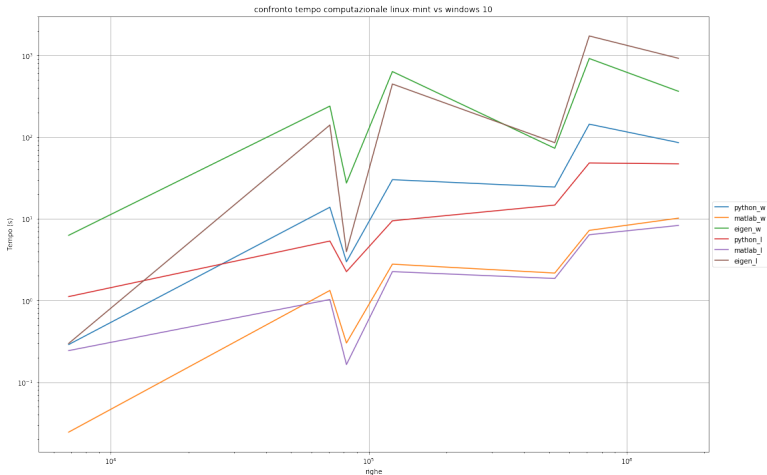
Librerie

Confronti

Listato Codice

Risultati

Conclusioni



Tempo Richiesto per StocF-1465 & Flan_1565

Introduzione

Sistemi Operativi

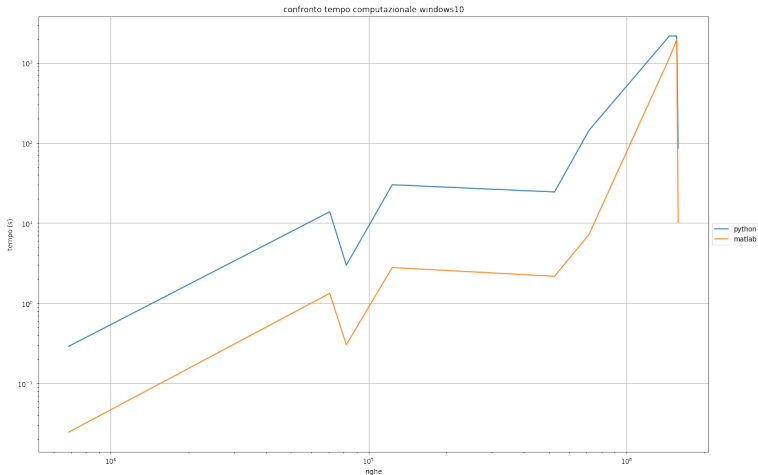
Librerie

Confronti

Listato Codice

Risultati

Conclusioni



Considerazioni sul tempo richiesto

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

- Per questa misura è evidente che Matlab è molto più performante rispetto a Python e Eigen che presenta il peggiore risultato.
- Per esempio per Apache2 Eigen va oltre le 10 ore di tempo, Scikit-Sparse circa 2 ore, invece Matlab circa 8 secondi.

Introduzione

Sistemi Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

- Dopo l'analisi delle misure effettuata nelle diapositive precedenti, MATLAB sembra essere lo strumento più veloce per risolvere sistemi lineari sparsi con la fattorizzazione di Cholesky.
- Tuttavia, su matrici di grandi dimensioni richiede una quantità di memoria elevata per funzionare correttamente.
- Produce anche errori relativi molto bassi, soprattutto per le matrici di grandi dimensioni riporta i risultati migliori.

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

- Scikit-sparse presenta i risultati migliori rispetto alla quantità di memoria richiesta per la fattorizzazione di Cholesky.
- Scikit-Sparse potrebbe essere una buona scelta per la risoluzione di matrici che richiedono tempi ragionevoli di esecuzione, inoltre è open source.
- L'errore relativo ottenuto è leggermente migliore di quello di Matlab rispetto alle matrici di piccole dimensione.

Introduzione

Sistemi
Operativi

Librerie

Confronti

Listato Codice

Risultati

Conclusioni

- Infine Eigen è stato il meno performante in termini di tempo, memoria e errore relativo.
- Eigen è stato utilizzato solo con le funzioni fornite di default, ma ha il vantaggio che è ampiamente supportato, è in futuro potrebbero essere rilasciati miglioramenti o estensioni a librerie esterne come BLAS / Lapack.

GRAZIE PER L'ATTENZIONE



Matamoros Ricardo 807450