

**CSC 435**  
**Project 2 – Your Very Own Threaded Baby BLAS**  
**Due 11:59 p.m., March 22, 2007**

In this project you will expand on Project 1 and write highly optimized shared memory multiprocessor C/C++ functions to do the most fundamental operations in matrix algebra:

- inner (dot) products of two vectors
- tensor product of two vectors (column vector times a row vector)
- matrix/vector multiplication (rotates the vector)
- matrix/matrix multiplication (square matrices only)

These single precision functions will be called from Fortran – and it is up to you to determine how to call these most effectively to minimize execution time. I will tell you that the calls on the FORTRAN side will look like.

```
dot(num_threads, N, vec1, vec2) — dot is a single precision function
call vvm(num_threads, N, vec1, vec2, rmat)
call mvv(num_threads, N, vec, mat, vresults)
call mmm(num_threads, N, mat1, mat2, rmat)
```

In addition, the code should be able to run on Pentium III and newer processors ( I will test on PIII and PIV processors). I am also expecting each of you to utilize a makefile for both the compilation of the code as well as creating the library. I will provide more information on this later.

When done, each of you should tar the directory(ies) containing your source code, and makefile. The source code should be well commented so I will know what improvements you made and you should also annotate your makefile so I will know what compilation flags you think work best. I will "make" your library, link it in with my own code, and run speed trials.

Grading will be as follows.

- A – Fastest code with correct results
- B – Code with correct results
- C – Incorrect Results
- D – Failure of compilation, makefile, linking
- F – Missing any components

Plagiarizing from the BLAS/Atlas libraries will not be tolerated and will wind you up in front of the honor council.

If you want, you can turn in the entire single processor (from project 1) and multiprocessor libraries as one file – if you can figure out how to overload the functions so FORTRAN will recognize them.