Subject: Specifying a particular LAPACK implementation using the standard IDL7 LAPACK DLM?
Posted by Matt Francis on Mon, 07 Jan 2013 03:32:51 GMT
View Forum Message <> Reply to Message

I have a task that has a bottleneck inverting large matrices. My understanding is that the LAPACK DLM that comes with IDL7 that is loaded when LA_INVERT is called for the first time is a serial implementation? I am running my code on an 8 core machine so would expect a reasonable speed up in the matrix inversion routine could utilise multiple cores to do the operation in parallel.

There are multi threaded LAPACK implementations, such as ATLAS, that I could use. If possible though, I'd like to avoid creating a new DLM from scratch. It would seem to me to be conceivable that if I could point IDL to the ATLAS (or whatever) multi-threaded LAPACK implementation instead of wherever it currently points to that this might neatly achieve what I want to do.

So, does anyone know if this is possible and if so how to do it? If not, does anyone have any suggestions for the simplest way to achieve what I want, which is parallel (dense) matrix inversion in IDL?

Subject: Re: Specifying a particular LAPACK implementation using the standard IDL7 LAPACK DLM?
Posted by Michael Galloy on Tue, 08 Jan 2013 19:53:18 GMT
View Forum Message <> Reply to Message

On 1/6/13 8:32 PM, Bogdanovist wrote:
> I have a task that has a bottleneck inverting large matrices. My
> understanding is that the LAPACK DLM that comes with IDL7 that is
> loaded when LA_INVERT is called for the first time is a serial
> implementation? I am running my code on an 8 core machine so would
> expect a reasonable speed up in the matrix inversion routine could
> utilise multiple cores to do the operation in parallel.
>
> There are multi threaded LAPACK implementations, such as ATLAS, that
> I could use. If possible though, I'd like to avoid creating a new DLM
> from scratch. It would seem to me to be conceivable that if I could
> point IDL to the ATLAS (or whatever) multi-threaded LAPACK
> implementation instead of wherever it currently points to that this
> might neatly achieve what I want to do.
>
> So, does anyone know if this is possible and if so how to do it? If
> not, does anyone have any suggestions for the simplest way to achieve
> what I want, which is parallel (dense) matrix inversion in IDL?

What platform are you targeting?

Looking at idl_lapack.so with nm on my platform (Mac OS X, IDL 8.2.1)
seems to indicate that the LAPACK routines are in the .so, so my (albeit

limited) understanding is that it would not be simple to swap out (i.e.,
by setting LD_LIBRARY_PATH).

$ nm idl_lapack.so
...
000000000023d5bb t _dgemm_
000000000023dfcb t _dgemv_
000000000009d9be t _dgeqp3_
000000000009e204 t _dgeqr2_
000000000009e41d t _dgeqrf_
000000000023e714 t _dger_
000000000009e9ad t _dgerfs_
000000000009f4ba t _dgerq2_
000000000009f6e1 t _dgerqf_

Some other possibilities...

My personal library (at https://github.com/mgalloy/mglib) has a
directory src/dist_tools/bindings containing routines for automatically
creating bindings for a library given "simple" (i.e., not using
typedefs) function prototypes. LAPACK would be fairly easy to create
bindings for (the hardest part would probably be to build the package
you want to use ATLAS, PLAPACK, ScaLAPACK, etc.). The library is free to
use, a small consulting contract could be done if you would like it done
for you.

The next version of GPULib (see www.txcorp.com) will contain a GPU
implementation of LAPACK, using the MAGMA library. This is effectively a
highly parallel option, but only works on CUDA graphics cards. It would
also work best if other operations besides the matrix inversion could be
done on the GPU to minimize memory transfer. This option costs money.

Mike
--
Michael Galloy
www.michaelgalloy.com
Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)
Research Mathematician
Tech-X Corporation