
Subject: GPULib 1.8 released

Posted by [Michael Galloy](#) on Mon, 02 Feb 2015 19:53:22 GMT

[View Forum Message](#) <> [Reply to Message](#)

GPULib 1.8 has been released with updates to the underlying libraries as well as many other features in many areas of the library. For information about purchasing, see:

<http://www.txcorp.com/home/gpulib>

It has been updated to use the most recent versions of IDL and CUDA, IDL 8.4 and CUDA 6.5. The new features are:

* Support for integer data types. I have been wanting to support integer types in GPULib for awhile and now GPULib supports all the numeric types provided by IDL! We can finally do:

```
dx = gpulndgen(10)
```

* Added `GPUREPMAT` routine. This is a handy routine to create a new array by repeating a 2-dimensional array in a grid.

* Added `GPUCREATEKERNEL` routine to create the source code of a simple kernel. This is a code generation routine that can be loaded with `GPULOADMODULE`/`GPULOADFUNCTION` and executed with `GPUEXECUTEFUNCTION`.

* Added `GPUFINITE` routine similar to IDL's library routine.

* Added linear algebra routines `GPULUDC`, `GPULUSOL`, and `GPULEAST\_SQUARES`. This fills out more of the GPU equivalent of the convenience routines provided by IDL so that the LAPACK interface of MAGMA is not required to perform linear algebra computations.

* Added support for `RHO` and `THETA` keywords in `GPURADON`.

* Added `GPUMEAN` routine. This routine has `DIMENSION` and `NAN` keywords with the same functionality as IDL's library routine.

Mike

--

Michael Galloy

www.michaelgalloy.com

Modern IDL: A Guide to IDL Programming (<http://modernidl.idldev.com>)

Research Mathematician

Tech-X Corporation

Subject: Re: GPULib 1.8 released

Posted by [markb77](#) on Tue, 03 Feb 2015 13:27:38 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Monday, February 2, 2015 at 8:53:26 PM UTC+1, Mike Galloy wrote:

> GPULib 1.8 has been released with updates to the underlying libraries as
> well as many other features in many areas of the library. For
> information about purchasing, see:
>
> <http://www.txcorp.com/home/gpulib>
>
> It has been updated to use the most recent versions of IDL and CUDA, IDL
> 8.4 and CUDA 6.5. The new features are:
>
> * Support for integer data types. I have been wanting to support
> integer types in GPULib for awhile and now GPULib supports all the
> numeric types provided by IDL! We can finally do:
>
> dx = gpulndgen(10)
>
> * Added `GPUREPMAT` routine. This is a handy routine to create a new
> array by repeating a 2-dimensional array in a grid.
>
> * Added `GPUCREATEKERNEL` routine to create the source code of a
> simple kernel. This is a code generation routine that can be loaded with
> `GPULOADMODULE`/`GPULOADFUNCTION` and executed with
> `GPUEXECUTEFUNCTION`.
>
> * Added `GPUFINITE` routine similar to IDL's library routine.
>
> * Added linear algebra routines `GPULUDC`, `GPULUSOL`, and
> `GPULEAST\_SQUARES`. This fills out more of the GPU equivalent of the
> convenience routines provided by IDL so that the LAPACK interface of
> MAGMA is not required to perform linear algebra computations.
>
> * Added support for `RHO` and `THETA` keywords in `GPURADON`.
>
> * Added `GPUMEAN` routine. This routine has `DIMENSION` and `NAN`
> keywords with the same functionality as IDL's library routine.
>
> Mike
> --
> Michael Galloy
> www.michaelgalloy.com
> Modern IDL: A Guide to IDL Programming (<http://modernidl.idldev.com>)
> Research Mathematician
> Tech-X Corporation

hi Mike,

Cool - thanks for the update. When will the documentation for version 1.8 be available online?

Mark

Subject: Re: GPULib 1.8 released

Posted by [Michael Galloy](#) on Tue, 03 Feb 2015 17:39:21 GMT

[View Forum Message](#) <> [Reply to Message](#)

On 2/3/15, 6:27 AM, superchromix wrote:

> hi Mike,

>

> Cool - thanks for the update. When will the documentation for version 1.8 be available online?

>

> Mark

>

Sorry, the documentation is on the web server, but the link to it wasn't made. I'll get it fixed shortly, thanks for the tip!

Mike

--

Michael Galloy

www.michaelgalloy.com

Modern IDL: A Guide to IDL Programming (<http://modernidl.idldev.com>)

Research Mathematician

Tech-X Corporation

Subject: Re: GPULib 1.8 released

Posted by [Michael Galloy](#) on Tue, 03 Feb 2015 18:12:48 GMT

[View Forum Message](#) <> [Reply to Message](#)

On 2/3/15, 10:39 AM, Michael Galloy wrote:

> On 2/3/15, 6:27 AM, superchromix wrote:

>> hi Mike,

>>

>> Cool - thanks for the update. When will the documentation for version
>> 1.8 be available online?

>>

>> Mark

>>

>

> Sorry, the documentation is on the web server, but the link to it wasn't
> made. I'll get it fixed shortly, thanks for the tip!

>
> Mike

Should be fixed now!

Mike

--

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

Subject: Re: GPULib 1.8 released
Posted by [markb77](#) on Thu, 05 Feb 2015 10:41:59 GMT
[View Forum Message](#) <> [Reply to Message](#)

hi Mike,

I notice that there is a new function called GPULEASTSQUARES. Is this an implementation of the GPU-based least squares fitting which you have mentioned in the past? It would be great if you could include a usage example in the documentation.

best
Mark

Subject: Re: GPULib 1.8 released
Posted by [Michael Galloy](#) on Thu, 05 Feb 2015 20:35:48 GMT
[View Forum Message](#) <> [Reply to Message](#)

On 2/5/15, 3:41 AM, superchromix wrote:

>
> hi Mike,
>
> I notice that there is a new function called GPULEASTSQUARES. Is
> this an implementation of the GPU-based least squares fitting which
> you have mentioned in the past? It would be great if you could
> include a usage example in the documentation.
>
> best Mark
>

The GPULib distribution includes a unit test for GPULEAST_SQUARES shown below (ignore the ASSERT statements, that is just checking to make sure things are as the test expects along the way):

```

function gpuleast_squares_ut::test_float
  compile_opt strictarr

  assert, !gpu.mode ne 0, 'GPULEAST_SQUARES not available in emulation
mode', /skip
  assert, !gpu.mode eq 0L || gpulmgr(/full), 'GPULib not licensed for
LAPACK calculations', /skip
  assert, !gpu.mode eq 0L || gpuMagmaPresent(), 'MAGMA not present', /skip

  a = [[1., 2., 1.], [4., 10., 15.], [3., 7., 1.]]
  dims = size(a, /dimensions)
  m = dims[0]
  n = dims[1]

  b = [2., 3., 7.]

  standard = la_least_squares(a, b)

  da = gpuputarr(a, ERROR=err)
  assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)

  db = gpuputarr(b, ERROR=err)
  assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)

  dresult = gpuleast_squares(da, db, ERROR=err)
  assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)

  result = gpugetarr(dresult, ERROR=err)
  assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)

  error = total(abs(standard - result), /preserve_type)
  assert, error lt self.tolerance * n * 10.0, 'incorrect result: error
= %g', error

  gpuFree, [da, db, dresult]

  return, 1
end

```

```

function gpuleast_squares_ut::test_double
  compile_opt strictarr

  assert, !gpu.mode ne 0, 'GPULEAST_SQUARES not available in emulation
mode', /skip
  assert, !gpu.mode eq 0L || gpulmgr(/full), 'GPULib not licensed for
LAPACK calculations', /skip

```

```
    assert, !gpu.mode eq 0L || gpuMagmaPresent(), 'MAGMA not present', /skip
    assert, !gpu.mode eq 0L || gpuDoubleCapable(), 'CUDA device not
double capable', /skip
```

```
    a = [[1.0D, 2.0D, 1.0D], [4.0D, 10.0D, 15.0D], [3.0D, 7.0D, 1.0D]]
    dims = size(a, /dimensions)
    m = dims[0]
    n = dims[1]
```

```
    b = [2.0D, 3.0D, 7.0D]
```

```
    standard = la_least_squares(a, b, /DOUBLE)
```

```
    da = gpuputarr(a, ERROR=err)
    assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
```

```
    db = gpuputarr(b, ERROR=err)
    assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
```

```
    dresult = gpuleast_squares(da, db, ERROR=err)
    assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
```

```
    result = gpugetarr(dresult, ERROR=err)
    assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
```

```
    error = total(abs(standard - result), /preserve_type)
    assert, error lt self.tolerance * n * 10.0, 'incorrect result: error
= %g', error
```

```
    gpuFree, [da, db, dresult]
```

```
    return, 1
end
```

Mike

--

Michael Galloy

www.michaelgalloy.com

Modern IDL: A Guide to IDL Programming (<http://modernidl.idldev.com>)

Research Mathematician

Tech-X Corporation

Subject: Re: GPULib 1.8 released

Posted by [markb77](#) on Fri, 06 Feb 2015 09:51:32 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Thursday, February 5, 2015 at 9:35:52 PM UTC+1, Mike Galloy wrote:

```

> On 2/5/15, 3:41 AM, superchromix wrote:
>>
>> hi Mike,
>>
>> I notice that there is a new function called GPULEASTSQUARES. Is
>> this an implementation of the GPU-based least squares fitting which
>> you have mentioned in the past? It would be great if you could
>> include a usage example in the documentation.
>>
>> best Mark
>>
>
> The GPULib distribution includes a unit test for GPULEAST_SQUARES shown
> below (ignore the ASSERT statements, that is just checking to make sure
> things are as the test expects along the way):
>
> function gpuleast_squares_ut::test_float
>   compile_opt strictarr
>
>   assert, !gpu.mode ne 0, 'GPULEAST_SQUARES not available in emulation
> mode', /skip
>   assert, !gpu.mode eq 0L || gpulmgr(/full), 'GPULib not licensed for
> LAPACK calculations', /skip
>   assert, !gpu.mode eq 0L || gpuMagmaPresent(), 'MAGMA not present', /skip
>
>   a = [[1., 2., 1.], [4., 10., 15.], [3., 7., 1.]]
>   dims = size(a, /dimensions)
>   m = dims[0]
>   n = dims[1]
>
>   b = [2., 3., 7.]
>
>   standard = la_least_squares(a, b)
>
>   da = gpuputarr(a, ERROR=err)
>   assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
>
>   db = gpuputarr(b, ERROR=err)
>   assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
>
>   dresult = gpuleast_squares(da, db, ERROR=err)
>   assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
>
>   result = gpugetarr(dresult, ERROR=err)
>   assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
>
>   error = total(abs(standard - result), /preserve_type)
>   assert, error lt self.tolerance * n * 10.0, 'incorrect result: error

```

```

> = %g', error
>
>   gpuFree, [da, db, dresult]
>
>   return, 1
> end
>
>
> function gpuleast_squares_ut::test_double
>   compile_opt strictarr
>
>   assert, !gpu.mode ne 0, 'GPULEAST_SQUARES not available in emulation
> mode', /skip
>   assert, !gpu.mode eq 0L || gpulmgr(/full), 'GPULib not licensed for
> LAPACK calculations', /skip
>   assert, !gpu.mode eq 0L || gpuMagmaPresent(), 'MAGMA not present', /skip
>   assert, !gpu.mode eq 0L || gpuDoubleCapable(), 'CUDA device not
> double capable', /skip
>
>   a = [[1.0D, 2.0D, 1.0D], [4.0D, 10.0D, 15.0D], [3.0D, 7.0D, 1.0D]]
>   dims = size(a, /dimensions)
>   m = dims[0]
>   n = dims[1]
>
>   b = [2.0D, 3.0D, 7.0D]
>
>   standard = la_least_squares(a, b, /DOUBLE)
>
>   da = gpuputarr(a, ERROR=err)
>   assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
>
>   db = gpuputarr(b, ERROR=err)
>   assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
>
>   dresult = gpuleast_squares(da, db, ERROR=err)
>   assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
>
>   result = gpugetarr(dresult, ERROR=err)
>   assert, err eq 0, 'CUDA error: %s', gpu_errormessage(err)
>
>   error = total(abs(standard - result), /preserve_type)
>   assert, error lt self.tolerance * n * 10.0, 'incorrect result: error
> = %g', error
>
>   gpuFree, [da, db, dresult]
>
>   return, 1
> end

```

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

I see - thanks. I guess I was looking for a non-linear least squares solver.
