

GPUmat Newsletter  
April 2009

---

With the April 2009 Newsletter, the *GP-you Group* presents the first beta release of the *Matlab* library *GPUmat*. The beta version is released under the Freeware license that can be found in the installation folder. In order to help us improving the library, please send any comment or suggestion to [gp-you@gp-you.org](mailto:gp-you@gp-you.org), or use the Forum space in our website <http://www.gp-you.org> Most of the information that follows can be found in the *GPUmat User Guide*, delivered together with the library or downloadable online. The following is a summary of *GPUmat* most important features:

- GPU computational power can be easily accessed from *Matlab* without any GPU knowledge.
- *Matlab* code is directly executed on the GPU. The execution is transparent to the user.
- *GPUmat* speeds up *Matlab* functions by using the GPU multi-processor architecture.
- Existing *Matlab* code can be ported and executed on GPUs with few modifications.
- GPU resources are accessed using *Matlab* scripting language. The fast code prototyping capability of the scripting language is combined with the fast code execution on the GPU.
- *GPUmat* can be used as a Source Development Kit to create new functions and extend the library functionality.

## The GP-you Group

The *GP-you Group* develops high-level software tools to easily access the parallel computing capabilities of the GPUs from different platforms (Windows, Linux and Mac). Thanks to the annual experience in software engineering and development (real time, embedded systems, numerical simulations), the GP-you Group proposes solutions that allow the end user to take full profit of the GPU capabilities without facing the costs related to the GPU code optimization and development: our solutions implement complex low level methods needed to tame the GPU computing power on the back-end while keeping a user-friendly and easy-to-use front-end.

---

## About GPUs

Although GPUs have been traditionally used only for computer graphics, a recent technique called GPGPU (General-purpose computing on graphics processing units) allows the GPUs to perform numerical computations usually handled by CPU. The advantage of using GPUs for general purpose computation is the performance speed up that can be achieved due to the parallel architecture of these devices.

One of the most promising GPGPU technologies is called CUDA SDK [1], developed by NVIDIA. For further information about CUDA, GPGPU and related topics please check [2] [3].

## System requirements

*GPUmat* was tested under Windows and Linux with Matlab ver. R2007a or newer installed. CUDA ver 2.1 should be installed on the system. Follow the instructions on NVIDIA's CUDA website [2] to download and install the software.

## GPUmat overview

*GPUmat* has been developed to speed up computational operations in *Matlab* by using the parallelism of the GPUs. The library is current under development. Table 1 shows a short summary of implemented functions and operators, while the full reference can be found in the *GPUmat User Guide* ([4]). More functions and packages (Financial, Finite Element Method) will be added to the library in the next releases.

---

Implemented functions	Example
Matlab operators (A*B, A-B, A.*B, A+B, etc.)	A = GPUsingle(rand(1000)); B = GPUsingle(rand(1000)); C = A + B;
Numerical functions (exp, sqrt, log, etc.)	A = GPUsingle(rand(1000)); B = GPUsingle(rand(1000)); C = exp(A); D = sqrt(C) + B;
Fast Fourier Transform	RE = GPUsingle(rand(1000)); IM = i*GPUsingle(rand(1000)); C = fft(RE + IM);

Table 1: Some *GPUmat* functions.

## Performance

The speed-up between the GPU and CPU for different examples is shown in Table 2. The hardware used for the tests is a PC Dual Core Intel 6600@2.4GHZ with a NVIDIA 8800GTX (128 stream processors) GPU. To calculate the speed-up the following code can be used:

```
N = 100:100:4000;
timecpu = zeros(1,length(N));
timegpu = zeros(1,length(N));

index=1;
for i=N
Ah = single(rand(i)); % CPU
A = GPUsingle(Ah); % GPU

%% Execution on GPU
```

---

```

tic;
A.*A;
GPUSync;
timegpu(index) = toc;

%% Execution on CPU
tic;
Ah.*Ah;
timecpu(index) = toc;

% increase index
index = index +1;
end

```

The above code calculates the two vectors *timecpu* and *timegpu* that can be used to evaluate the speed-up between the GPU and the CPU as follows:

```
speedup = timecpu./timegpu
```

Test	N.	CPU time (s)	GPU time (s)	Speed up
A.*B	1600x1600	0.0778	0.0026	29.44
A*B	1600x1600	2.81	0.24	11.66
fft2(A)	1600x1600	0.1710	0.0189	9.0463
exp(A)	1600x1600	0.2911	0.0029	99.72

Table 2: GPU time vs. CPU time for different tests on complex numbers.

## Future development

*GPUmat* currently allows the user to run the most important *Matlab* functions on the GPU, on both complex and real single precision floating point types. The *GP-you Group* plans to include in the library more features for the upcoming releases, such as Financial and Finite Element packages and the support to double precision floating point numbers. Please send your suggestions or any bug to [gp-you@gp-you.org](mailto:gp-you@gp-you.org).

# Bibliography

- [1] *NVIDIA Cuda Programming Guide*. NVIDIA Corporation.
- [2] Cuda. [http://www.nvidia.com/object/cuda\\_home.html#](http://www.nvidia.com/object/cuda_home.html#).
- [3] Gpgpu. <http://www.gpgpu.org>.
- [4] *GPUmat User Guide*. The GP-you Group.