GPU-accelerated libraries

CUBLAS & CURAND

Шевченко Александр, АРС

Постановка задачи

Требуется найти максимальное по модулю собственное число матрицы и соответствующий собственный вектор

$$\lambda \in eig(A) \Leftrightarrow \exists x : Ax = \lambda x$$

Истоки: оценки спектра оператора, вычислительная математика, …

Методика решения

- Упорядочим все собственные числа по убыванию*:

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_n|$$

- Разложим любой вектор по собственным векторам**:

$$x = C_1 a_1 + C_2 a_2 + \ldots + C_n a_n$$

$$A^k x = C_1 \lambda_1^k a_1 + C_2 \lambda_2^k a_2 + \ldots + C_n \lambda_2^k a_n$$

$$= \lambda_1^k \left(C_1 a_1 + C_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k a_2 + \ldots + C_n \left(\frac{\lambda_n}{\lambda_1} \right)^k a_n \right)$$

$$A^k x \approx C_1 a_1 \lambda_1^k$$

$$\lim_{k \to \infty} \left(\frac{\left\| A^k x \right\|}{C_1 \left| \lambda_1 \right|^k \left\| a_1 \right\|} \right) = 1$$

Методика решения

Алгоритм:

- Начальное приближение $x_0 : \|x_0\| = 1$
- $y_k = Ax_{k-1}$ $\qquad x_k = \dfrac{y_k}{\|y_k\|}$
- Итерации продолжать до тех пор, пока не наступит установление* вектора.

$$\|x_{k+1} - x_k\| < \varepsilon$$

- Собственное значение можно найти по формуле:

$$\lambda_1 \approx \frac{x_k^T A x_k}{x_k^T x_k}$$

Что нужно?

- Сгенерировать матрицу и начальный вектор
- Сделать функцию, которая будет делать итерации
  - Умножение матрицы на вектор
  - Линейные комбинации векторов
  - Вычисление нормы
  - Вычисление скалярного произведения
- Сделать функцию, которая после итераций вычислит собственное число

Простенькие CUDA ядра?!

Библиотеки

- Поставляются с SDK, почти бесплатны
- «Привычны»
- Основной принцип работы:
  - Инициализировать контекст

```
cublasCreate(&cublasHandle);
```

  - Провести требуемые вычисления

```
cublasSnrm2(cublasHandle, n, y, 1, &norm)
cublasSdot(cublasHandle, n, x, 1, y, 1, &num)
```

  - Удалить контекст

```
cublasDestroy(cublasHandle);
```

Функции CUBLAS

```
cublasStatus cublas<T>function(cublasHandle_t handle, …)
```

+ дополнительные «обслуживающие» функции

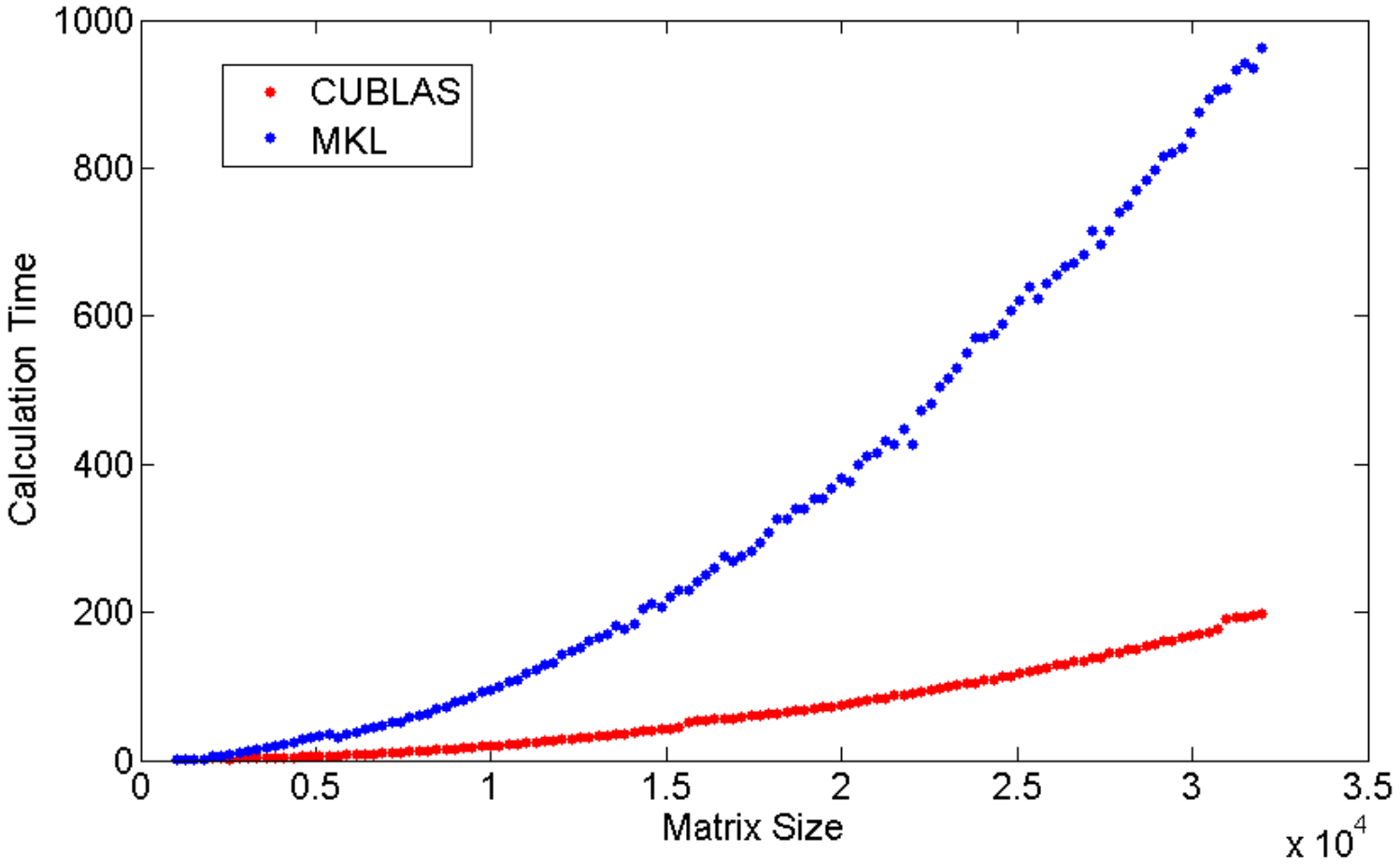
Название	Что делает
Level1: cublas<T>saxpy	y = alpha * y + x
Level2: cublas<T>gemv	y = alpha * op(A) * x + beta * y
Level3: cublas<T>gemm	C = aplha * op(A) * op(B) + beta * C

Измерение времени в Linux + CUDA

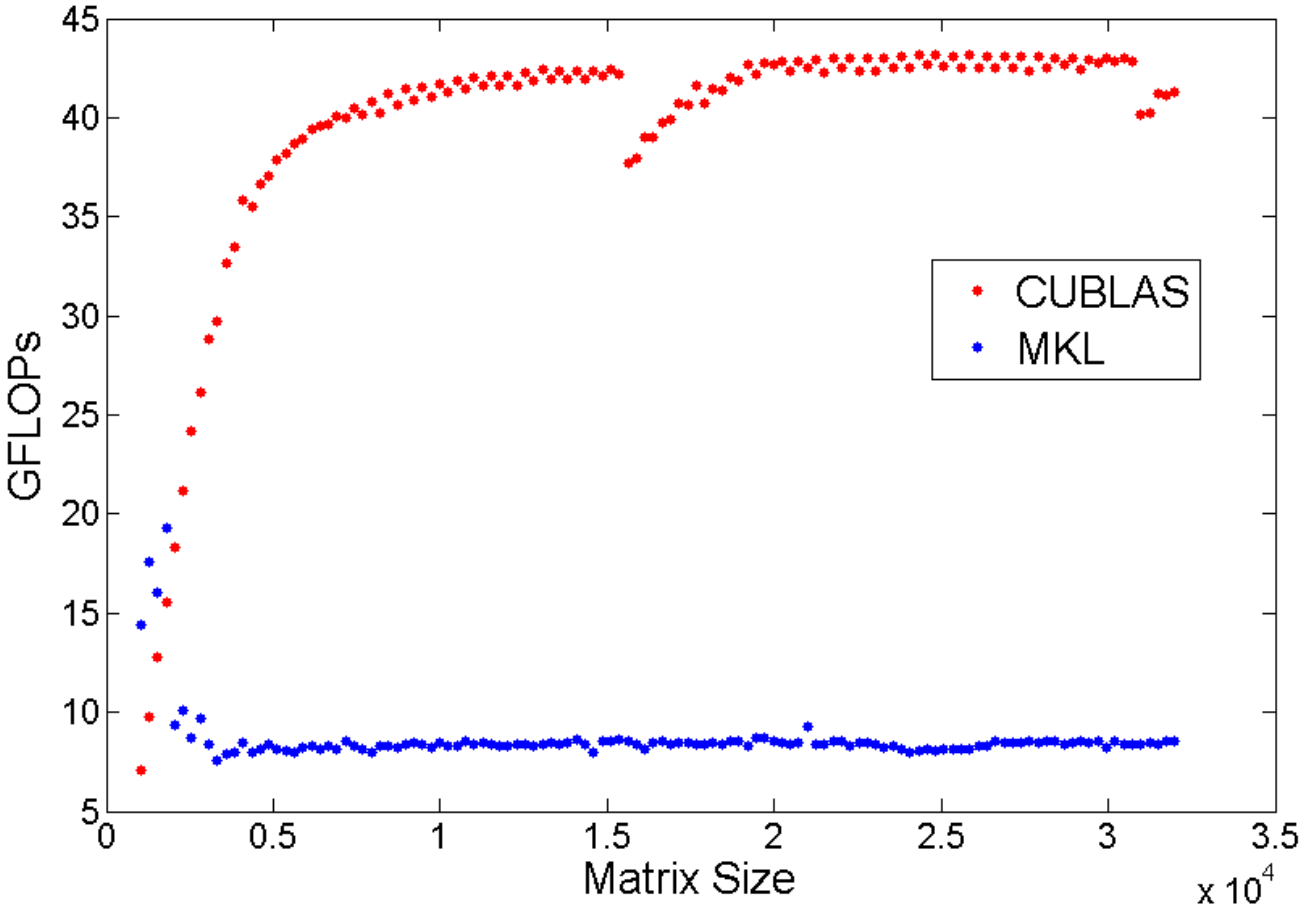
```
cudaEvent_t start, stop;
cudaEventCreate(&start);
cudaEventCreate(&stop);
cudaEventRecord(start, 0);
//Some calculation
//
//It may take a lot of time
//
//And lines of code =)
cudaEventRecord(stop, 0);
cudaEventSynchronize(stop);
float elapsedTime; //in milliseconds
cudaEventElapsedTime(&elapsedTime, start, stop);
cudaEventDestroy(start);
cudaEventDestroy(stop);
```

Демонстрация программы

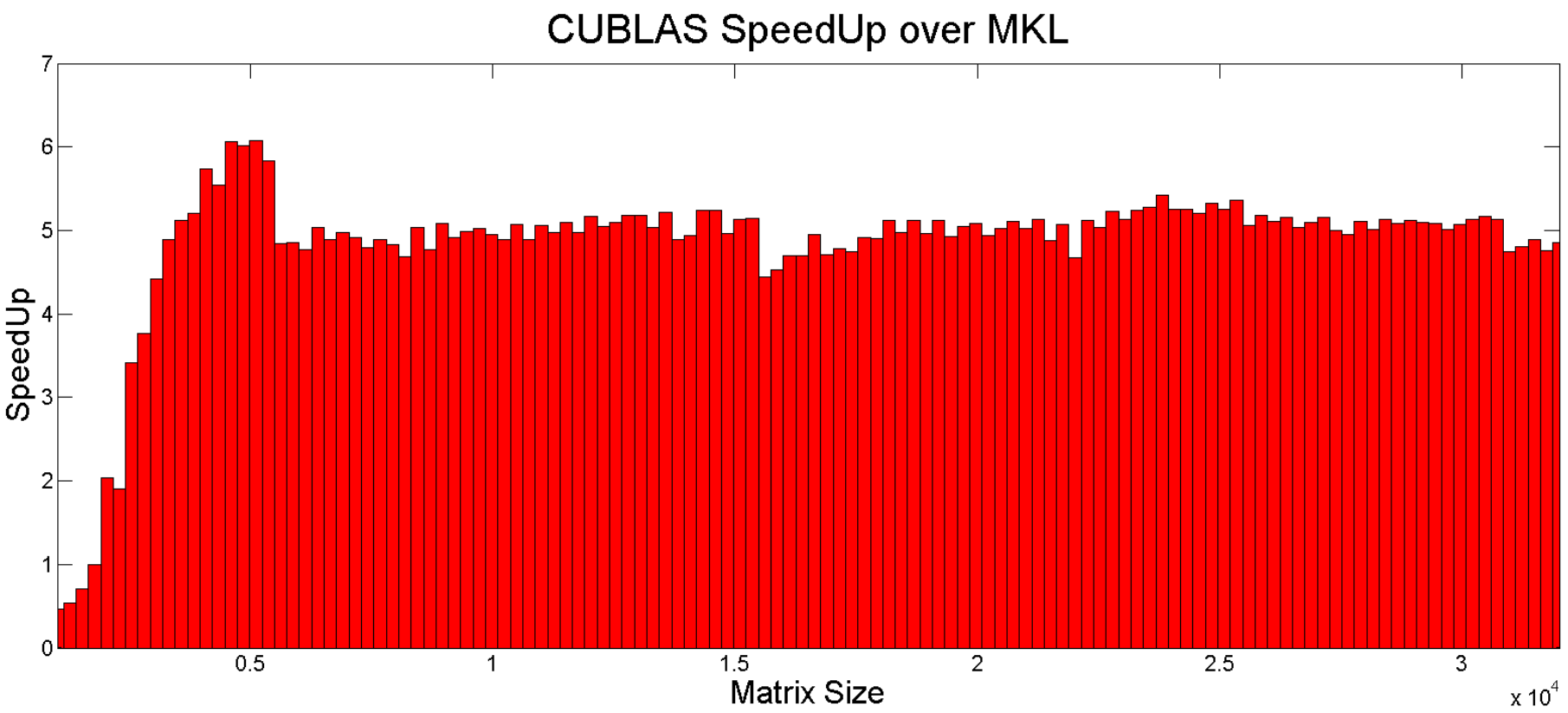
Результаты работы



Результаты работы



Результаты работы



Вопросы ?