Microsoft R OpenによるRの高速化
ーWindows 10、Python+numpy+matpoltlibも併せて評価ー

今回の内容：計算ベンチマークとグラフ描画比較

1. 以前openblasを適用したRのベンチマークを行った[1]。Intel MKLでどれくらい速くなるか試した
2. 100万個のデータにより正弦波を計算し、グラフ描画した
   また、Python+numpy+matpoltlibの描画速度を比較した

結果の概要

1. Windows上で、Linuxのopenblasの環境よりも若干高速になった
2. 100万個のsin演算も高速だった
3. グラフ描画は、Python+numpy+matpoltlibの方が圧倒的に高速

ベンチマークに使用したPCのスペック

1. CPU： Intel Core i5 6600K 3.5GHz 4コア
   デスクトップPCです

2. RAM: 32GB

3. ドライブ
   Cドライブ: SSD(480GB)、Eドライブ: HDD(3TB)

4. RとPythonの環境（OSは、Windows 10 Pro）
   R: RstudioからRを起動
   Python： 3系、anaconda

5. グラフィック： NVIDIA GTX 960

R-benchmark-25.Rのベンチマーク結果

	Core i5 6600K 3.5GHz 4コア RAM 32GB		
	Ubuntu 15.10	Windows 7 Pro	Windows 10 Pro
ライブラリ	OpenBlas-0.2.14 ソースからビルド	標準のR	Intel MKL (Microsoft R Open[2])
2800x2800 行列のクロス積 (b = a' * a)	*0.1556667*	12.08667	0.1600000
3000x3000 行列のコレスキー分解	*0.1126667*	4.783333	0.1066666
15 種類の全テストの合計時間	*4.0253333*	30.53	3.0233333

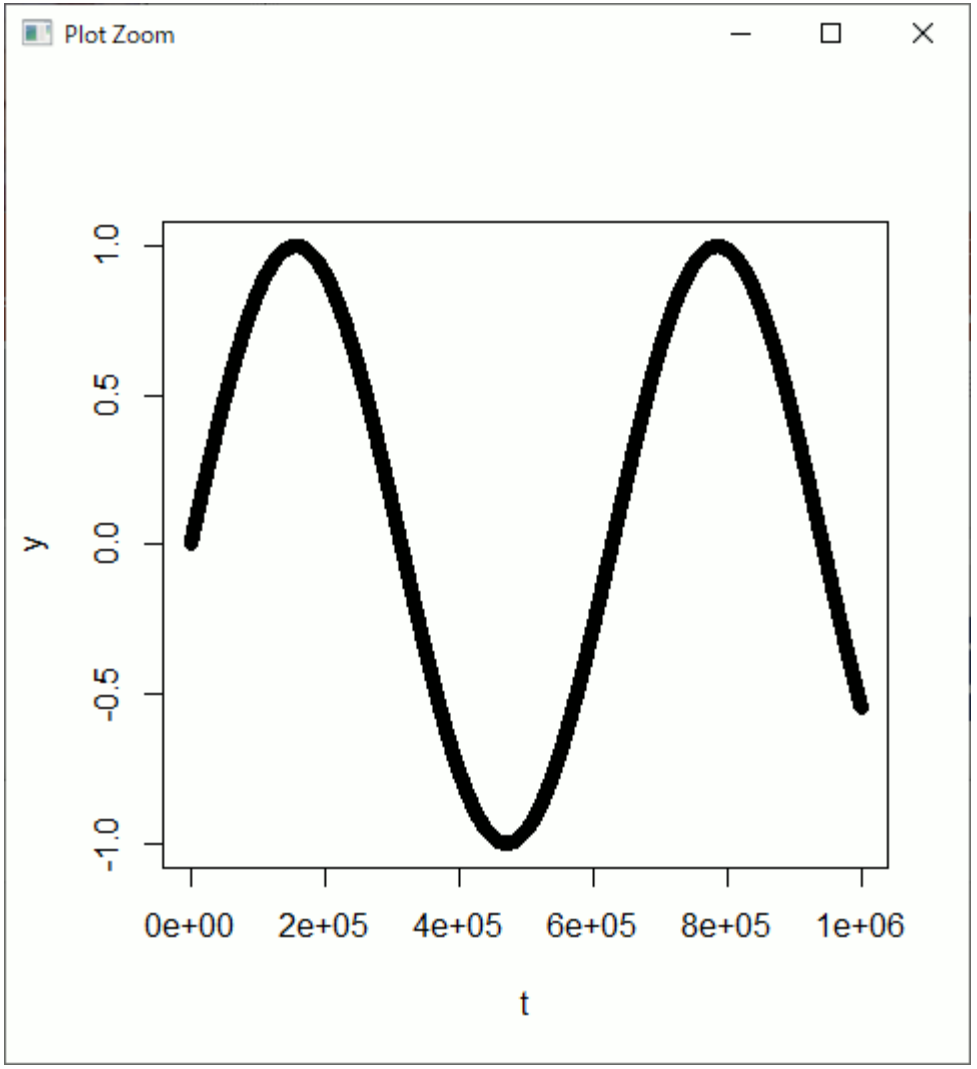
項目により、Ubuntuのopenblasの方が若干高速なものがあるが、全テストの合計時間ではIntel MKL（Microsoft R Open）の方が若干高速であった

R(Microsoft R Open)／100万個のデータによりグラフ描画

#Rのコマンドと出力結果です

```
> t <- c(0:1e6)
> system.time(sin(t/100000))
   ユーザ   システム      経過
     0.06       0.00       0.06
> y <- sin(t/100000)
> plot(t,y)
```

グラフが表示されるまで
58秒かかった。



計算は高速化されますが、プロットが遅いです

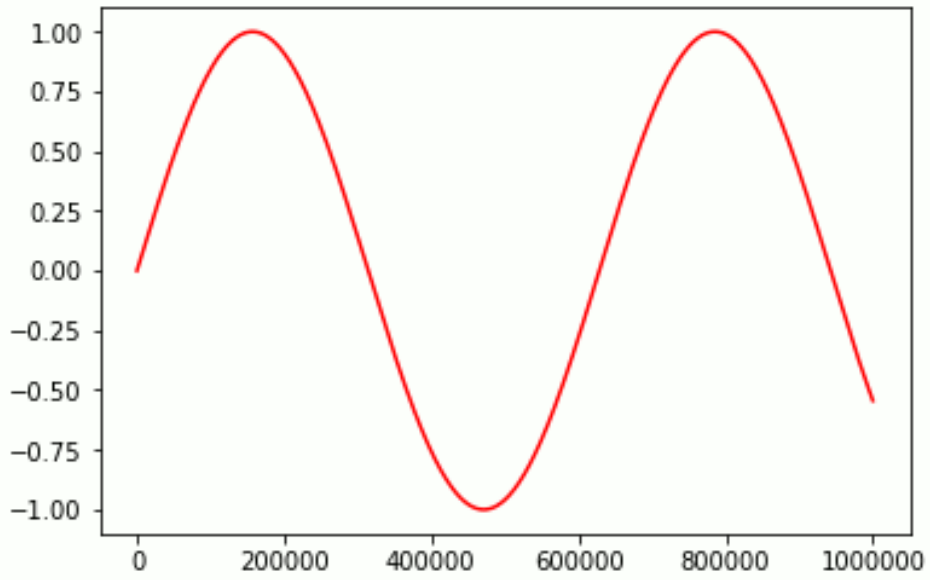
Python3(anaconda)／100万個のデータによりグラフ描画

```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np

In [2]: t = np.arange(1000000)

In [3]: plt.plot(t, np.sin(t/100000), "r")

Out[3]: [<matplotlib.lines.Line2D at 0x14ae1fe9c88>]
```



約1秒でグラフが表示されます（Rと比べて劇的に高速）

まとめ

1. Intel MKLが適用されたMicrosoft R Openを使用することにより、Rの計算を高速化することができます。
   但し、グラフの描画は低速です。

2. 同じ内容をPython3（Anaconda）のPython+numpy+matplotlibで試したところ、グラフ描画が劇的に速くなりました

参考資料

1. 無償BLAS-LAPACKライブラリによる浮動小数点演算ベンチマーク
   http://opencae.gifu-nct.ac.jp/pukiwiki/index.php?%C2%E8%A3%B4%A3%B5%B2%F3%CA%D9%B6%AF%B2%F1%A1%A7H280206

2. Microsoft R Open
   https://mran.microsoft.com/open