Computing Correlation

Ray Seyfarth

June 29, 2012

Outline

1. Correlation
2. Correlation in C
3. Implementation using SSE instructions
4. Implementation using AVX instructions

Correlation

- First the definition of correlation

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

- Next a formula more amenable to computation

$$r_{xy} = \frac{n \sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n \sum x_i^2 - (\sum x_i)^2}\sqrt{n \sum y_i^2 - (\sum y_i)^2}}$$

- This formula requires computing 5 sums while processing the 2 arrays

A simple C solution

```c
#include <math.h>
double corr ( double x[], double y[], long n )
{
    double sum_x, sum_y, sum_xx, sum_yy, sum_xy;
    long i;

    sum_x = sum_y = sum_xx = sum_yy = sum_xy = 0.0;
    for ( i = 0; i < n; i++ ) {
        sum_x += x[i];
        sum_y += y[i];
        sum_xx += x[i]*x[i];
        sum_yy += y[i]*y[i];
        sum_xy += x[i]*y[i];
    }
    return (n*sum_xy-sum_x*sum_y)/
           sqrt((n*sum_xx-sum_x*sum_x)*(n*sum_yy-sum_y*sum_y));
}
```

Simple C solution results

- gcc used all 16 XMM registers
- It unrolled the basic loop 4 times
- Is also handled non multiple of 4 array sizes
- Performing 1 million calls for arrays of size 10000 used 13.44 seconds for 5.9 GFLOPS
- Excellent for compiled code

SSE implementation

- The XMM registers were used to accumulate partial sums for the various sums
- 10 registers were used to hold 4 partial sums for each of the 5 required sums
- After the main loop the partial sums were added together
- Horizontal adds were used to add the 2 halves of registers
- Then the correlation was computed
- 1 million calls for arrays of size 10000 used 6.74 seconds or 11.8 GFLOPS
- This is about 3.5 double precision floating point results per CPU cycle
- Quite impressive

AVX implementation

- The YMM registers were used to accumulate partial sums for the various sums
- 10 registers were used to hold 8 partial sums for each of the 5 required sums
- After the main loop the partial sums were added together
- Horizontal adds were used to add the numbers in the 2 halves of registers
- Unfortunately this was not quite enough to add all 4 values
- A little more bit wrangling was required to add the partial sums
- Then the correlation was computed
- 1 million calls for arrays of size 10000 used 3.9 seconds or 20.5 GFLOPS
- This is about 6 double precision floating point results per CPU cycle
- Amazing for 1 core of a 4 core CPU