```cpp
// g++ -O3 -ffast-math -funsafe-math-optimizations -msse4.2
template <typename Real = double, ull degree = 200, ull I = 1>
struct Expv1 {
    static Real evaluate(Real x) {
        constexpr Real c = 1.0 / static_cast<Real>(I);
        x = 1.0 + c * x * Expv1<Real, degree, I + 1>::evaluate(x);
        return x;
    }
};

template <typename Real, ull degree>
struct Expv1<Real, degree, degree> {
    static Real evaluate(Real x) {
        constexpr Real c = 1.0 / static_cast<Real>(degree);
        x = 1.0 + c * x;
        return x;
    }
};

template <typename Real, ull degree = 30, ull i = 0>
struct Expv2 {
    static Real evaluate(Real x) {
        // constexpr Real c = 1.0 / static_cast<Real>(1ull << degree);
        x = Expv2<Real, degree, i + 1>::evaluate(x);
        return x * x;
```

Contoso

基本数学函数的设计与实现

218,52 90%

数学函数

1、$Sinx$

2、$\sqrt{x}$

3、x^a

4、$Log_e x$

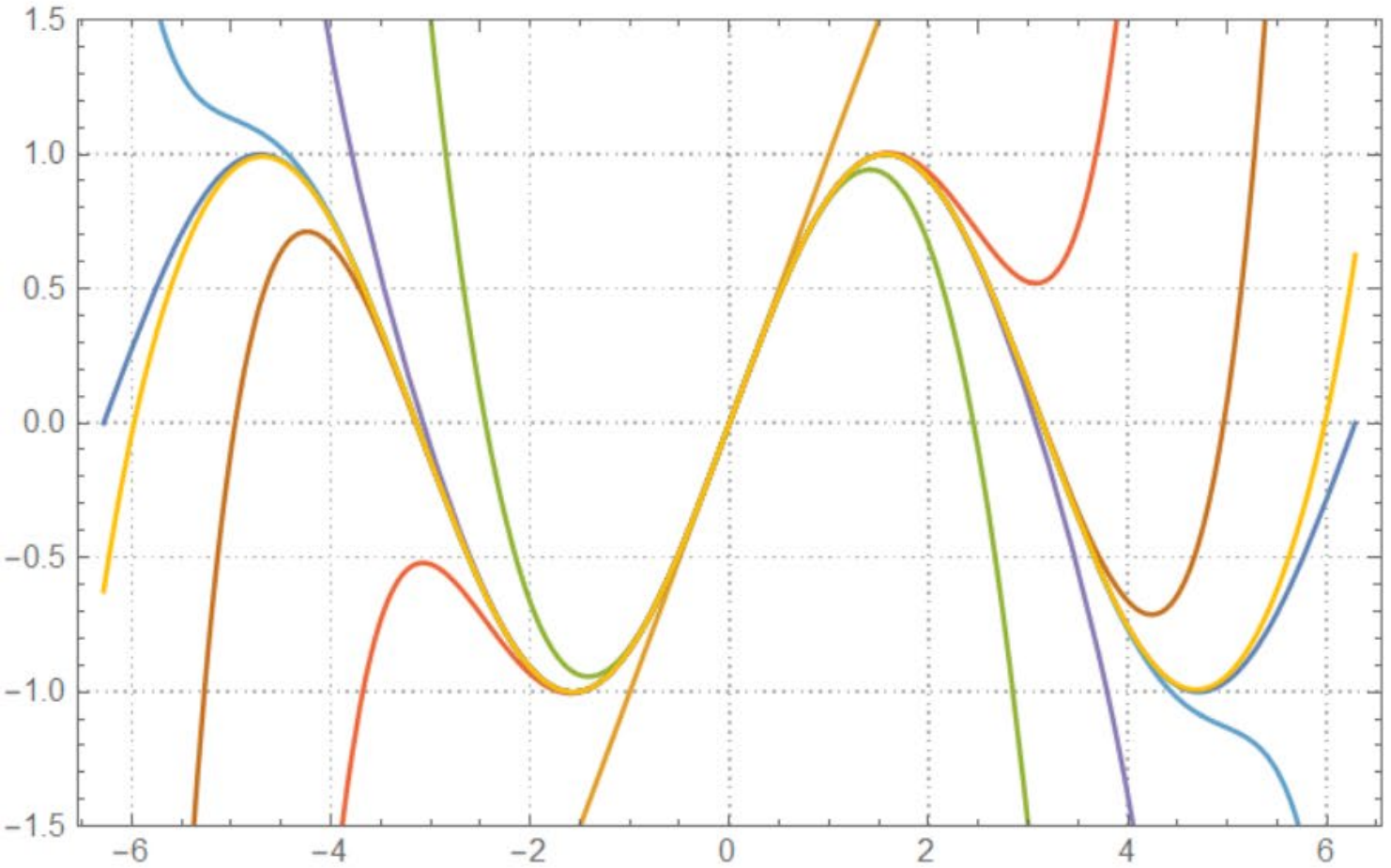
$Sinx$

$$f(x) = f(x_0) + \sum_{k=1}^{\infty} \frac{f^{(k)}(x_0)(x - x_0)^k}{k!}$$

$$\mathrm{Sin}(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} + O(x^{11})$$

$Sinx$

$$\mathrm{Sin}(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} + O(x^{11})$$



$\sin(x)$

x

$x - \frac{x^3}{6}$

$x - \frac{x^3}{6} + \frac{x^5}{120}$

$x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040}$

$x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040} + \frac{x^9}{362880}$

$x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040} + \frac{x^9}{362880} - \frac{x^{11}}{39916800}$

$x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040} + \frac{x^9}{362880} - \frac{x^{11}}{39916800} + \frac{x^{13}}{6227020800}$

$Sinx$

$$\mathrm{Sin}(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} + O(x^{11})$$

$$Z_0 = 0 \qquad Z_1 = x$$

$$Z_2 = Z_1 - \frac{x^3}{3!}$$

$$Z_3 = Z_2 + \frac{x^5}{5!}$$

...

$$Z_{k+1} = Z_k + (-1)^k \frac{x^{2k+1}}{(2k+1)!}$$

代码

Contoso

```
2020-11-08T20:39:01+08:00
Running /mnt/d/C++/Math/build/Sin
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.23, 0.05, 0.02
-----------------------------------------------------
Benchmark              Time             CPU   Iterations
-----------------------------------------------------
Simple_Sin       96689 ns        96689 ns         5987
```

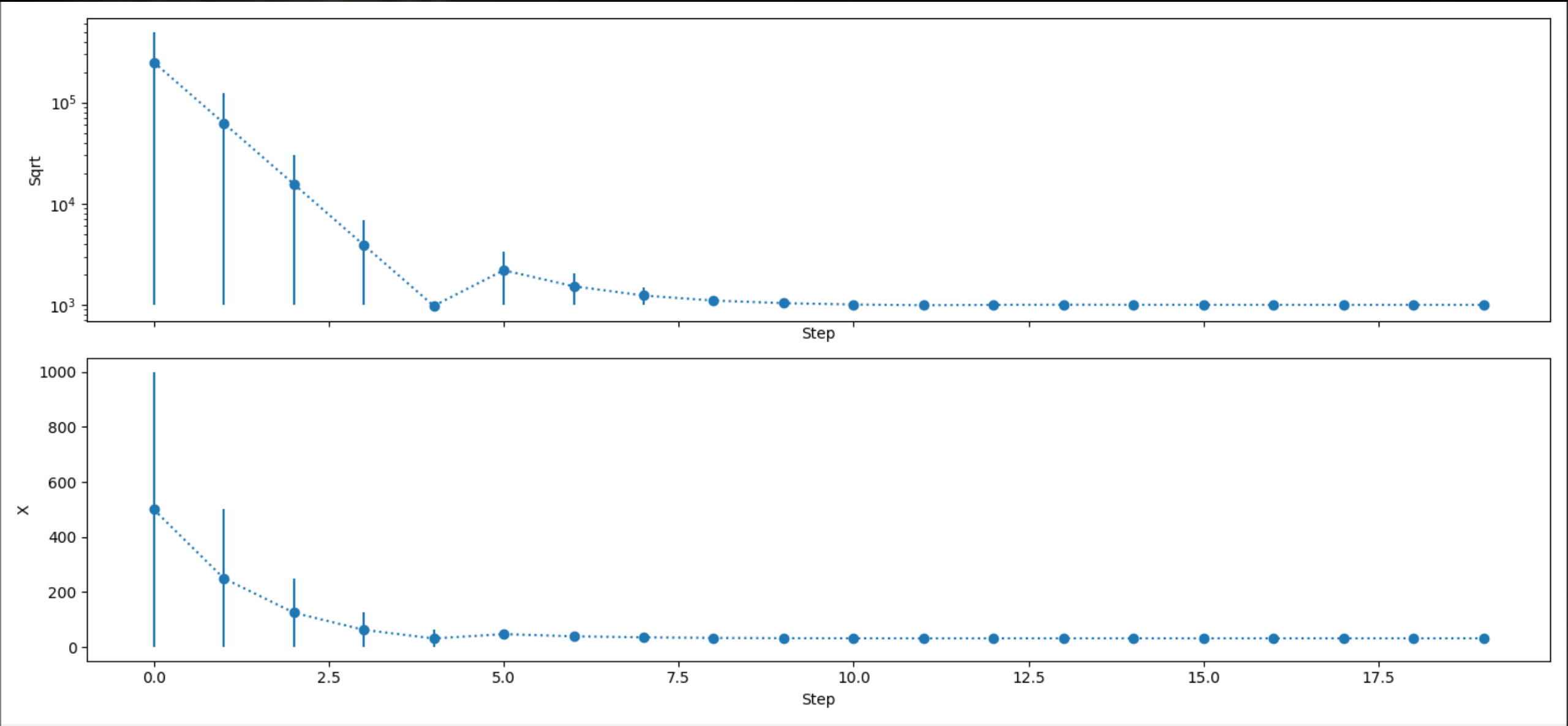
```cpp
double sin_simple(double A) {
    const int N = 50;
    A = std::fmod(A, 2 * Pi);
    double result = 0.0;
    double k = 1.0;
    double n;
    double x = A;
    double xx = A * A;
    for (int i = 1; i <= N; ++i) {
        if (i & 1)
            result += x / k;
        else
            result -= x / k;
        n = 2 * i - 1;
        k *= (n + 1) * (n + 2);
        x *= xx;
    }
    return result;
}
```

Contoso

$$\sqrt{x}$$

$$\sqrt{x+1} = 1 + \frac{x}{2} - \frac{x^2}{8} + \frac{x^3}{16} - \frac{5x^4}{128} + \frac{7x^5}{256} - \frac{21x^6}{1024} + \frac{33x^7}{2048} - \frac{429x^8}{32768} + O[x]^9$$

二分搜索



x=-0.0567198 y=521.443

代码

```
2020-11-08T21:43:18+08:00
Running /mnt/d/C++/Math/build/Sqrt
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.16, 0.05, 0.01
-----------------------------------------------------
Benchmark                Time             CPU   Iterations
-----------------------------------------------------
Bisection_Sqrt     292070 ns       292061 ns         2368
```

```cpp
double sqrt_bisection(double A) {
    assert(A > 0);
    double low = 0, high = A;
    double mid = (low + high) * 0.5;
    double square = mid * mid;
    while (std::abs(square - A) > epsilon)
        if (square > A)
            high = mid;
        else
            low = mid;
        mid = (low + high) * 0.5;
        square = mid * mid;
    }
    return mid;
}
```

Contoso

太慢了

还能更快吗?

牛顿迭代法

$$A = \sqrt{x} \times \sqrt{x}$$

$$x^2 - A = 0$$

令 $f(x) = x^2 - A = 0$

$$f(x) = f(x_0) + (x - x_0)f'(x_0) + \frac{1}{2}(x - x_0)^2 f''(x_0) + \frac{1}{6}f^{(3)}(x_0)(x - x_0)^3$$

$$+ \frac{1}{24}f^{(4)}(x_0)(x - x_0)^4 + \frac{1}{120}f^{(5)}(x_0)(x - x_0)^5 + O((x - x_0)^6)$$

$$f(x) = f(x_0) + (x - x_0)f'(x_0)$$

牛顿迭代法

$$f(x_0) + (x - x_0)f'(x_0) = 0$$

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

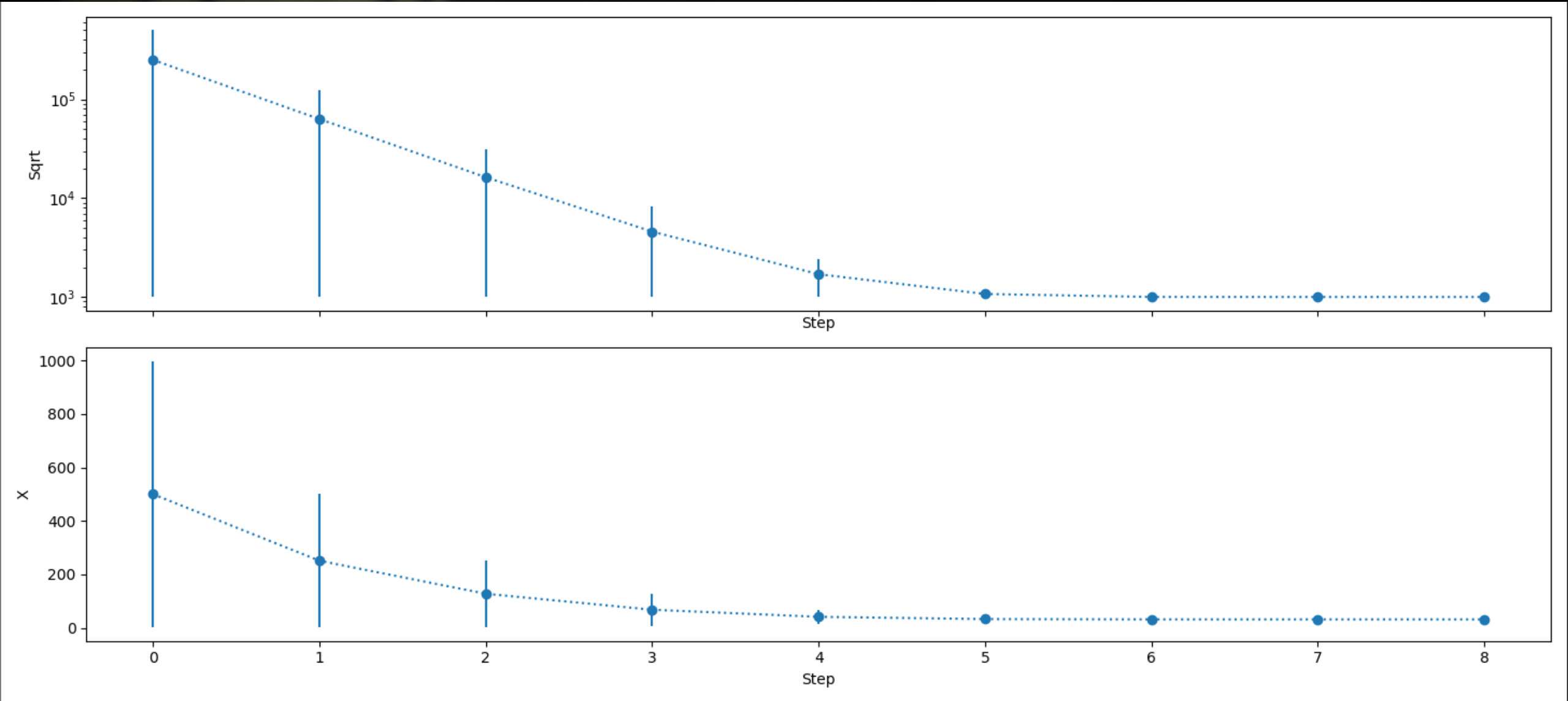
$$x_{n+1} = x_0 - \frac{f(x_n)}{f'(x_n)}$$

牛顿迭代法

$$f(x) = x^2 - A = 0$$
$$f'(x) = 2x$$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = \frac{1}{2}\left(\frac{A}{x_n} + x_n\right)$$

牛顿迭代法



x=0.00456572 y=505.761

代码

```
2020-11-08T21:43:18+08:00
Running /mnt/d/C++/Math/build/Sqrt
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.16, 0.05, 0.01
-------------------------------------------------------
Benchmark                Time             CPU   Iterations
-------------------------------------------------------
Bisection_Sqrt      292070 ns       292061 ns         2368
Newton_Sqrt          25624 ns        25624 ns        26402
```

```cpp
double sqrt_newton(double A) {
    assert(A > 0);
    double xn = A * 0.5;
    double square = xn * xn;
    while (std::abs(square - A) > epsilon)
        xn = (xn + A / xn) * 0.5;
        square = xn * xn;
    }
    return xn;
}
```

Contoso

快多了

但还能更快吗?


Contoso

Contoso


QUAKE™

代码

```
2020-11-08T21:43:18+08:00
Running /mnt/d/C++/Math/build/Sqrt
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.16, 0.05, 0.01
-----------------------------------------------------
Benchmark                Time             CPU   Iterations
-----------------------------------------------------
Bisection_Sqrt      292070 ns       292061 ns         2368
Newton_Sqrt          25624 ns        25624 ns        26402
Carmack_Sqrt          7327 ns         7327 ns        91397
```

```
double C_sqrt(double A) {
    const double ahalf = (A * 0.5);
    int64_t i = *((int64_t*) &A);
    i = (0x1FF7A3BEA91D9B00 + (i >> 1));
    A = *((double*) &i);
    A *= (0.5 + ((ahalf / A) / A));
    A *= (0.5 + ((ahalf / A) / A));
    A *= (0.5 + ((ahalf / A) / A));
    return A;
}
```

Contoso

$$x^n$$

$$x^n = x \times x \times x \times x \times \cdots \times x$$

代码

```
2020-11-08T23:44:20+08:00
Running /mnt/d/C++/Math/build/Pow
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.17, 0.05, 0.01
-----------------------------------------------------
Benchmark           Time             CPU   Iterations
-----------------------------------------------------
Simple_Pow       3287 ns         3287 ns       204790
```

```cpp
double pow_simple(double A, int n) {
    double result = 1.0;
    while (n) {
        result *= A;
        --n;
    }
    return result;
}
```

Contoso

x^n

$$n = 45$$

$$n = 00101101\text{b}$$

0	0	1	0	1	1	0	1
x^{128}	x^{64}	x^{32}	x^{16}	x^8	x^4	x^2	x^1
0	0	x^{32}	0	x^8	x^4	0	x^1

$$x^{45} = x^{32} \times x^8 \times x^4 \times x^1$$

代码

```
2020-11-08T23:44:20+08:00
Running /mnt/d/C++/Math/build/Pow
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.17, 0.05, 0.01
-----------------------------------------------------
Benchmark             Time             CPU   Iterations
-----------------------------------------------------
Simple_Pow         3287 ns         3287 ns       204790
Fast Pow            753 ns          753 ns       907192
```

```cpp
double pow_fast(double A, int n) {
    double result = 1.0;
    while (n) {
        if (n & 1)
            result *= A;
        A *= A;
        n >>= 1;
    }
    return result;
}
```

Contoso

$$x^a \quad a \in R$$

$$x^a = e^{a \ln x}$$

$$e^x$$

$$\ln x$$

Contoso

e^x

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \frac{x^6}{6!} + \frac{x^7}{7!} + O(x^8)$$

$$Z_k = 1 + \frac{x}{k} Z_{k+1}$$

$$Z_n = 1 + \frac{x}{n}$$

代码

```
2020-11-09T09:30:13+08:00
Running /mnt/d/C++/Math/build/Exp
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.38, 0.16, 0.11
-----------------------------------------------------
Benchmark             Time             CPU   Iterations
-----------------------------------------------------
Simple_Exp     253826 ns       253825 ns         2725
```

```cpp
double exp_simple(double A) {
    const int degree = 1000;
    double k, z = 1.0;
    for (int i = degree; i > 0; --i) {
        k = 1.0 / static_cast<double>(i);
        z = 1.0 + k * A * z;
    }
    return z;
}
```

Contoso

太慢了

还能更快吗?

Contoso

Template Metaprogramming

$$n! = n \times (n-1)!$$

```
using ull = unsigned long long;

template <typename Real, ull N>
constexpr Real Factorial = static_cast<Real>(N) * Factorial<Real, N - 1ULL>;

template <typename Real>
constexpr Real Factorial<Real, 0ULL> = static_cast<Real>(1.0);
```

Template Metaprogramming

Contoso

Source:

```cpp
#include <iostream>

using ull = unsigned long long;

template <typename Real, ull N>
constexpr Real Factorial = static_cast<Real>(N) * Factorial<Real, N - 1ULL>;

template <typename Real>
constexpr Real Factorial<Real, 0ULL> = static_cast<Real>(1.0);

int main() {
  auto fac5 = Factorial<int, 5>;
  std::cout << fac5 << std::endl;
}
```

Insight:

```cpp
#include <iostream>

using ull = unsigned long long;



template<typename Real, ull N>
constexpr const Real Factorial = static_cast<Real>(N) * Factorial<Real, N - 1ULL>;

template<>
constexpr const int Factorial<int, 5> = static_cast<int>(5ULL) * Factorial<int, 5ULL - 1ULL>;
template<>
constexpr const int Factorial<int, 4> = static_cast<int>(4ULL) * Factorial<int, 4ULL - 1ULL>;
template<>
constexpr const int Factorial<int, 3> = static_cast<int>(3ULL) * Factorial<int, 3ULL - 1ULL>;
template<>
constexpr const int Factorial<int, 2> = static_cast<int>(2ULL) * Factorial<int, 2ULL - 1ULL>;
template<>
constexpr const int Factorial<int, 1> = static_cast<int>(1ULL) * Factorial<int, 1ULL - 1ULL>;
template<>
constexpr const int Factorial<int, 0> = static_cast<int>(1.0);

template <typename Real>
constexpr Real Factorial<Real, 0ULL> = static_cast<Real>(1.0);

int main()
{
  int fac5 = Factorial<int, 5>;
  std::cout.operator<<(fac5).operator<<(std::endl);
}

```

Template Metaprogramming

Contoso

A Save/Load + Add new... Vim CppInsights Quick-bench C++

```
1  #include <iostream>
2
3  using ull = unsigned long long;
4
5  template <typename Real, ull N>
6  constexpr Real Factorial = static_cast<Real>(N) * Factorial<Real, N - 1ULL>;
7
8  template <typename Real>
9  constexpr Real Factorial<Real, 0ULL> = static_cast<Real>(1.0);
10
11 int main() {
12   auto fac5 = Factorial<int, 5>;
13   std::cout << fac5 << std::endl;
14 }
```

x86-64 clang 10.0.1 -std=c++2a

A Output... Filter... Libraries + Add new... Add tool...

```
1  __cxx_global_var_init:                  # @__cxx_global_var_init
2          pushq   %rbp
3          movq    %rsp, %rbp
4          movabsq $_ZStL8__ioinit, %rdi
5          callq   std::ios_base::Init::Init() [complete object constructor]
6          movabsq $_ZNSt8ios_base4InitD1Ev, %rax
7          movq    %rax, %rdi
8          movabsq $_ZStL8__ioinit, %rsi
9          movabsq $__dso_handle, %rdx
10         callq   __cxa_atexit
11         popq    %rbp
12         retq
13 main:                                   # @main
14         pushq   %rbp
15         movq    %rsp, %rbp
16         subq    $16, %rsp
17         movl    $120, -4(%rbp)
18         movl    -4(%rbp), %esi
19         movabsq $_ZSt4cout, %rdi
20         callq   std::basic_ostream<char, std::char_traits<char> >::operator<<(in
21         movq    %rax, %rdi
22         movabsq $_ZSt4endlIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_, %rs
23         callq   std::basic_ostream<char, std::char_traits<char> >::operator<<(st
24         xorl    %ecx, %ecx
25         movq    %rax, -16(%rbp)         # 8-byte Spill
26         movl    %ecx, %eax
27         addq    $16, %rsp
28         popq    %rbp
29         retq
```

Output (0/0) x86-64 clang 10.0.1 - 2438ms (297759B)

Template Metaprogramming

Contoso

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \frac{x^6}{6!} + \frac{x^7}{7!} + O(x^8)$$

$$Z_k = 1 + \frac{x}{k} Z_{k+1}$$

$$Z_n = 1 + \frac{x}{n}$$

Template Metaprogramming

Contoso

```
template <typename Real = double, ull degree = 1000, ull I = 1>
struct Expv1 {
    static Real evaluate(Real x) {
        constexpr Real c = 1.0 / static_cast<Real>(I);
        x = 1.0 + c * x * Expv1<Real, degree, I + 1>::evaluate(x);
        return x;
    }
};

template <typename Real, ull degree>
struct Expv1<Real, degree, degree> {
    static Real evaluate(Real x) {
        constexpr Real c = 1.0 / static_cast<Real>(degree);
        x = 1.0 + c * x;
        return x;
    }
};
```

Template Metaprogramming

Contoso

```
2020-11-09T09:30:13+08:00
Running /mnt/d/C++/Math/build/Exp
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.38, 0.16, 0.11
-----------------------------------------------------
Benchmark             Time             CPU   Iterations
-----------------------------------------------------
Simple_Exp       253826 ns       253825 ns         2725
TMPv1_Exp         44001 ns        44000 ns        15521
```

Template Metaprogramming

Contoso

$$e^x = \lim_{n \to \infty} \left(1 + \frac{x}{n}\right)^n$$

$$Z_n = \left(1 + \frac{x}{n}\right)^n$$

Template Metaprogramming

Contoso

```
template <typename Real, ull degree = 50, ull i = 0>
struct Expv2 {
    static Real evaluate(Real x) {
        x = Expv2<Real, degree, i + 1>::evaluate(x);
        return x * x;
    }
};

template <typename Real, ull degree>
struct Expv2<Real, degree, degree> {
    static Real evaluate(Real x) {
        constexpr Real c = 1.0 / static_cast<Real>(1ULL << degree);
        x = 1.0 + c * x;
        return x;
    }
};
```

Template Metaprogramming

Contoso

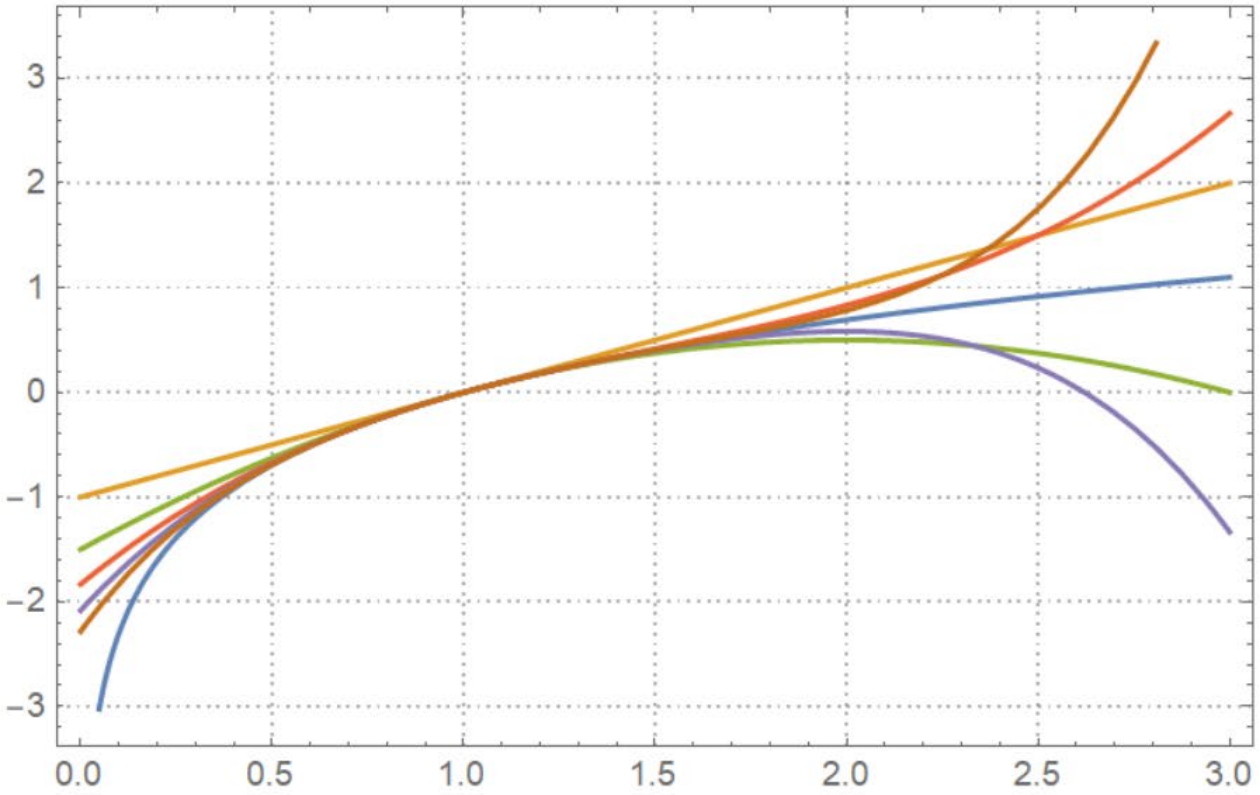
```
2020-11-09T09:30:13+08:00
Running /mnt/d/C++/Math/build/Exp
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.38, 0.16, 0.11
```

Benchmark	Time	CPU	Iterations
Simple_Exp	253826 ns	253825 ns	2725
TMPv1_Exp	44001 ns	44000 ns	15521
TMPv2_Exp	1084 ns	1084 ns	625550

$\ln x$

$$\ln x = (x-1) - \frac{1}{2}(x-1)^2 + \frac{1}{3}(x-1)^3 - \frac{1}{4}(x-1)^4 + \frac{1}{5}(x-1)^5 - \frac{1}{6}(x-1)^6 + O(x-1)^7$$

$\ln x$



$\log(x)$

$-1 + x$

$-1 - \frac{1}{2}(-1+x)^2 + x$

$-1 - \frac{1}{2}(-1+x)^2 + \frac{1}{3}(-1+x)^3 + x$

$-1 - \frac{1}{2}(-1+x)^2 + \frac{1}{3}(-1+x)^3 - \frac{1}{4}(-1+x)^4 + x$

$-1 - \frac{1}{2}(-1+x)^2 + \frac{1}{3}(-1+x)^3 - \frac{1}{4}(-1+x)^4 + \frac{1}{5}(-1+x)^5 + x$

$\ln x$

$$\ln(x) = 2\tanh^{-1}\left(\frac{1-x}{1+x}\right) = 2\sum_{n=0}^{\infty}\frac{1}{2n+1}\left(\frac{1-x}{1+x}\right)^{2n+1}$$

$$记\ A = \frac{1-x}{1+x}$$

$$\ln(x) = 2\tanh^{-1}(A) = 2\sum_{n=0}^{\infty}\frac{1}{2n+1}(A)^{2n+1}$$

代码

```
2020-11-09T11:04:29+08:00
Running /mnt/d/C++/Math/build/Log
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.21, 0.07, 0.02
-----------------------------------------------------
Benchmark             Time             CPU   Iterations
-----------------------------------------------------
Simple_Log     249378 ns       249379 ns         2855
```

```
double ln_simple(double A) {
    double x = (A - 1.0) / (A + 1.0);
    double xx = x * x;
    double k = 2.0 * 1000 + 1.0;
    double y = 0;
    while (k > 0.0) {
        y = 1.0 / k + xx * y;
        k -= 2.0;
    }
    return 2.0 * x * y;
}
```

Contoso

太慢了

还能更快吗?

Template Metaprogramming

Contoso

```
2020-11-09T11:04:29+08:00
Running /mnt/d/C++/Math/build/Log
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.21, 0.07, 0.02
-----------------------------------------------------
Benchmark             Time             CPU   Iterations
-----------------------------------------------------
Simple_Log       249378 ns       249379 ns         2855
TMP_Log           41580 ns        41580 ns        15609
```

```cpp
template <typename Real = double, ull degree = 1000, ull I = 1>
struct Atanh {
    static Real evaluate(Real x) {
        constexpr Real k = 1.0 / (2.0 * static_cast<Real>(I) - 1.0);
        x = k + x * x * Atanh<Real, degree, I + 1>::evaluate(x);
        return x;
    }
};

template <typename Real, ull degree>
struct Atanh<Real, degree, degree> {
    static Real evaluate(Real x) {
        constexpr Real k1 = 1.0 / (2.0 * static_cast<Real>(degree) - 1.0);
        constexpr Real k2 = 1.0 / (2.0 + k1);
        x *= x;
        x = k1 + x * k2;
        return x;
    }
};

template <typename Real = double, ull degree = 1000>
struct Ln {
    static Real evaluate(Real x) {
        Real A = (x - 1.0) / (x + 1.0);
        Real ln = 2 * A * Atanh<Real, degree>::evaluate(A);
        return ln;
    }
};
```

Contoso

还是慢

还能更快吗?

Assembly

```
2020-11-09T11:04:29+08:00
Running /mnt/d/C++/Math/build/Log
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.21, 0.07, 0.02
-----------------------------------------------------
Benchmark             Time             CPU   Iterations
-----------------------------------------------------
Simple_Log       249378 ns       249379 ns         2855
TMP_Log           41580 ns        41580 ns        15609
Asm_Log            1802 ns         1802 ns       381033
```

```
double ln_asm(double x) {
    asm volatile("fldln2  ;"
                 "fldl  %0;"
                 "fyl2x   ;"
                 "fstpl %0;"
                 : "+m"(x));
    return x;
}

double log10_asm(double x) {
    asm volatile("fldlg2  ;"
                 "fldl  %0;"
                 "fyl2x   ;"
                 "fstpl %0;"
                 : "+m"(x));
    return x;
}

double log2_asm(double x) {
    asm volatile("fld1    ;"
                 "fldl  %0;"
                 "fyl2x   ;"
                 "fstpl %0;"
                 : "+m"(x));
    return x;
}
```

Assembly

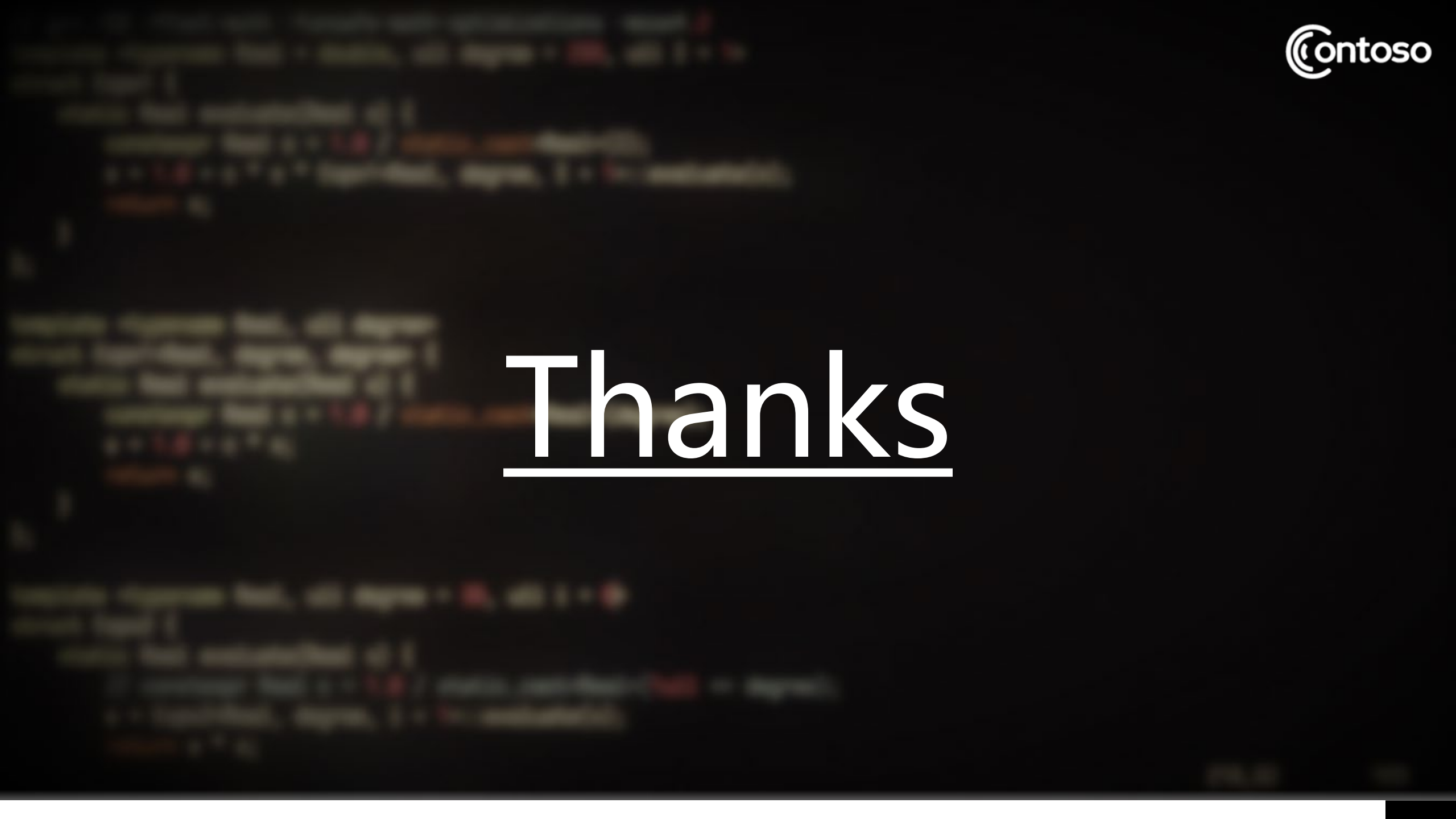
```
2020-11-09T11:11:40+08:00
Running /mnt/d/C++/Math/build/Sqrt
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.17, 0.06, 0.01
-----------------------------------------------------
Benchmark                Time             CPU   Iterations
-----------------------------------------------------
Bisection_Sqrt      270108 ns       270106 ns         2480
Newton_Sqrt          24501 ns        24501 ns        28500
Carmack_Sqrt          6952 ns         6952 ns        96835
Asm_Sqrt              2064 ns         2064 ns       325977
```

```
double sqrt_asm(double A) {
    asm volatile("fldl  %0;"
                 "fsqrt   ;"
                 "fstpl %0;"
                 : "+m"(A));
    return A;
}
```

Assembly

```
2020-11-09T11:14:39+08:00
Running /mnt/d/C++/Math/build/Sin
Run on (8 X 2808 MHz CPU s)
CPU Caches:
  L1 Data 32 KiB (x4)
  L1 Instruction 32 KiB (x4)
  L2 Unified 256 KiB (x4)
  L3 Unified 6144 KiB (x1)
Load Average: 0.02, 0.04, 0.00
-----------------------------------------------------
Benchmark             Time             CPU   Iterations
-----------------------------------------------------
Simple_Sin       100161 ns       100164 ns         6522
Asm_Sin           36492 ns        36492 ns        18060
```

```
double sin_asm(double A) {
    asm volatile("fldl  %0;"
                 "fsin    ;"
                 "fstpl %0;"
                 : "+m"(A));
    return A;
}
```


Contoso
Thanks