1000 input values :

Incl.	Self	Called	Function	Location
11 812	2 400	200	Add(double, double)	stddev: lib.h
10 050	4 715	113	Exponent(double, int)	stddev: lib.h
1 401	400	1	Root(double, int)	stddev: lib.h
150	36	2	Divide(double, double)	stddev: lib.h
130	24	2	Subtract(double, double)	stddev: lib.h
73	12	1	Multiply(double, double)	stddev: lib.h

100 input values:

Incl.	Self	Called	Function	Location
118 012	24 000	2 000	Add(double, double)	stddev: lib.h
91 050	43 415	1 013	Exponent(double, int)	stddev: lib.h
150	36	2	Divide(double, double)	stddev: lib.h
130	24	2	Subtract(double, double)	stddev: lib.h
1 401	400	1	Root(double, int)	stddev: lib.h
73	12	1	Multiply(double, double)	stddev: lib.h

10 input values:

Incl.	Self	Called	Function	Location
1 950	845	23	Exponent(double, int)	stddev: lib.h
1 192	240	20	Add(double, double)	stddev: lib.h
130	24	2	Subtract(double, double)	stddev: lib.h
150	36	2	Divide(double, double)	stddev: lib.h
73	12	1	Multiply(double, double)	stddev: lib.h
1 401	400	1	Root(double, int)	stddev: lib.h

Above are reports from KCachegrind displaying function usage when calculating standard deviation with 10, 100 and 1000 input values.

Sadly, Profiling through vTune was unfortunate and couldn't be taken in count, the code ran too quick to capture any reasonable data.

The program mostly calls Exponent and Add function. Add is called 2 times the number count and exponent is called 1 times the number count plus 13. Looking at the optimalization, the exponential function can be further enhanced.