Range Reduction

One execution

TIME OC: 799.05626700000005 TIME CC: 761.56983049265386

```
SDOUBLE x    = LOAD_DOUBLE_VEC(&input[i]);

const SDOUBLE ranges_away = MUL_DOUBLE_S(x, one_over_2_pi);
const SDOUBLE num_ranges_away = FLOOR_DOUBLE_S(ranges_away);
const SDOUBLE range_multiple = MUL_DOUBLE_S(num_ranges_away, two_pi);
const SDOUBLE in_outer_range = SUB_DOUBLE_S(x, range_multiple);

SIMD_TO_DOUBLE_VEC(&res[i], in_outer_range);
```

Two executions

TIME OC: 794.39666099999999 TIME CC: 788.55947432130688

```
SDOUBLE x    = LOAD_DOUBLE_VEC(&input[i]);

const SDOUBLE ranges_away = MUL_DOUBLE_S(x, one_over_2_pi);
const SDOUBLE num_ranges_away = FLOOR_DOUBLE_S(ranges_away);
const SDOUBLE range_multiple = MUL_DOUBLE_S(num_ranges_away, two_pi);
const SDOUBLE in_outer_range = SUB_DOUBLE_S(x, range_multiple);

const SDOUBLE ranges_away1 = MUL_DOUBLE_S(in_outer_range, one_over_2_pi);
const SDOUBLE num_ranges_away1 = FLOOR_DOUBLE_S(ranges_away1);
const SDOUBLE range_multiple1 = MUL_DOUBLE_S(num_ranges_away1, two_pi);
const SDOUBLE in_outer_range1 = SUB_DOUBLE_S(in_outer_range, range_multiple1);

SIMD_TO_DOUBLE_VEC(&res[i], in_outer_range);
```

Taylor Polynom Plot

/home/admin/MasterArbeit/code/plots/taylor_degree_test.png

Graph

This Gives the result in the plot for one loop

```
    SDOUBLE x    = LOAD_DOUBLE_VEC(&input[i]);
    const SDOUBLE centered_values = SUB_DOUBLE_S(x, center_point);
    SDOUBLE result = LOAD_DOUBLE(TAYLOR_COEFF_SIN[taylor_last_coeff]);

    for (int j = taylor_loop_iteration; j >= 0; --j) {
    SDOUBLE coeff = LOAD_DOUBLE(TAYLOR_COEFF_SIN[j]);
    result = MUL_DOUBLE_S(result, centered_values);
    result = ADD_DOUBLE_S(result, coeff);
    }

    SIMD_TO_DOUBLE_VEC(&res[i], result);
```

This gives the results in the plot for two loops:

```
SDOUBLE x    = LOAD_DOUBLE_VEC(&input[i]);
const SDOUBLE centered_values = SUB_DOUBLE_S(x, center_point);
SDOUBLE result = LOAD_DOUBLE(TAYLOR_COEFF_SIN[taylor_last_coeff]);

for (int j = taylor_loop_iteration; j >= 0; --j) {
SDOUBLE coeff = LOAD_DOUBLE(TAYLOR_COEFF_SIN[j]);
result = MUL_DOUBLE_S(result, centered_values);
result = ADD_DOUBLE_S(result, coeff);
}

const SDOUBLE centered_values1 = SUB_DOUBLE_S(x, center_point);
SDOUBLE result1 = LOAD_DOUBLE(TAYLOR_COEFF_SIN[taylor_last_coeff]);

for (int j = taylor_loop_iteration; j >= 0; --j) {
SDOUBLE coeff = LOAD_DOUBLE(TAYLOR_COEFF_SIN[j]);
result1 = MUL_DOUBLE_S(result1, centered_values1);
result = ADD_DOUBLE_S(result1, coeff);
}
SIMD_TO_DOUBLE_VEC(&res[i], result);
```

Quadrant Evaluation setup

The following test setup took

- TIME OC: 770.9451940000001
- TIME CC: 775.73264921769305

```
SDOUBLE x    = LOAD_DOUBLE_VEC(&input[i]);

const SDOUBLE multiplied_quadrants = MUL_DOUBLE_S(x, quadrant_multiplier);
const SDOUBLE quadrant_evaluation = ADD_DOUBLE_S(multiplied_quadrants, addition_ve
const SDOUBLE quadrant_evaluated_result = MUL_DOUBLE_S(x, quadrant_evaluation);

const SDOUBLE multiplied_quadrants1 = MUL_DOUBLE_S(x, quadrant_evaluated_result);
const SDOUBLE quadrant_evaluation1 = ADD_DOUBLE_S(multiplied_quadrants1, addition_
const SDOUBLE quadrant_evaluated_result1 = MUL_DOUBLE_S(quadrant_evaluated_result,

SIMD_TO_DOUBLE_VEC(&res[i], quadrant_evaluated_result1);
```

The following test setup took – TIME OC: 818.80264299999999 – TIME CC: 795.44322714426596

```
SDOUBLE x    = LOAD_DOUBLE_VEC(&input[i]);

const SDOUBLE multiplied_quadrants = MUL_DOUBLE_S(x, quadrant_multiplier);
const SDOUBLE quadrant_evaluation = ADD_DOUBLE_S(multiplied_quadrants, addition_vect
const SDOUBLE quadrant_evaluated_result = MUL_DOUBLE_S(x, quadrant_evaluation);

SIMD_TO_DOUBLE_VEC(&res[i], quadrant_evaluated_result);
```