Проведено сравнение сложности реализации двух алгоритмов: Sum of Squared Difference (SSD) и Normalized Sum of Squared Difference (NSSD). Данные алгоритмы могут применяться для выполнения объёмной работы вычисления корреляции цифровых изображений, например для нахождения смещений в образцах материалов до и после нагружения (сжатие, растяжение, изгиб, температурное и т.д.).

Ниже приведен код, реализующий оба алгоритмы на языке программирования C# (код был написан во времена студенчества).

```
internal class SSD : ICorrelating
{
    public double GetCorrelation(byte[,] f, byte[,] g)
    {
        double result = 0;

        for (int i = 0; i < f.GetLength(0); i++)
        {
            for (int j = 0; j < f.GetLength(1); j++)
            {
                result += Math.Pow(f[i, j] - g[i, j], 2);
            }
        }

        return result;
    }
}
```

```
internal class NSSD : ICorrelating
{
    public double GetCorrelation(byte[,] f, byte[,] g)
    {
        double result = 0;

        double f2 = 0;
        double g2 = 0;

        for (int i = 0; i < f.GetLength(0); i++)
        {
            for (int j = 0; j < f.GetLength(1); j++)
            {
                f2 += Math.Pow(f[i, j], 2);
                g2 += Math.Pow(g[i, j], 2);
            }
        }

        for (int i = 0; i < f.GetLength(0); i++)
        {
            for (int j = 0; j < f.GetLength(1); j++)
            {
                result += Math.Pow((f[i, i] / f2) - (g[i, i] / g2), 2);
            }
        }

        return result;
    }
}
```

Для алгоритма SSD видно, что у нас два цикла, один из которых вложенный. Сразу можно понять, что сложность по времени $O(N^2)$. Сложность по памяти O(N), т.к. мы для любого набора входных данных мы храним только этот самый набор и временные переменные, кол-во которых не зависит от роста объёма данных.

В алгоритме NSSD четыре цикла, два из которых вложенные. Затраты памяти от увеличения объёмов входных данных также как и в случае с SSD растут линейно, следовательно сложность по памяти O(N). Так как у нас два цикла, то получается, что каждый из них имеет сложность $O(N^2)$. Складывая сложности получим: $N^2 + N^2 = 2\ N^2$. Общий множитель “2” не оказывает существенно влияния на рост сложности с увеличением объёмов данных, следовательно им можно пренебречь. Итоговая временная сложность: $O(N^2)$.

Сложности по времени и по памяти для данных алгоритмов оказались эквивалентными, но следует понимать, что в любом случае итоговое время работы и затраты памяти для них на одном и том же наборе входных данных не будет одинаковым и даже учитывая совпадающий характер роста сложности, конечный выбор следует делать опираясь на ограничения ресурсов, требования к точности алгоритма и результаты тестирования с конкретными показателями времени и памяти..