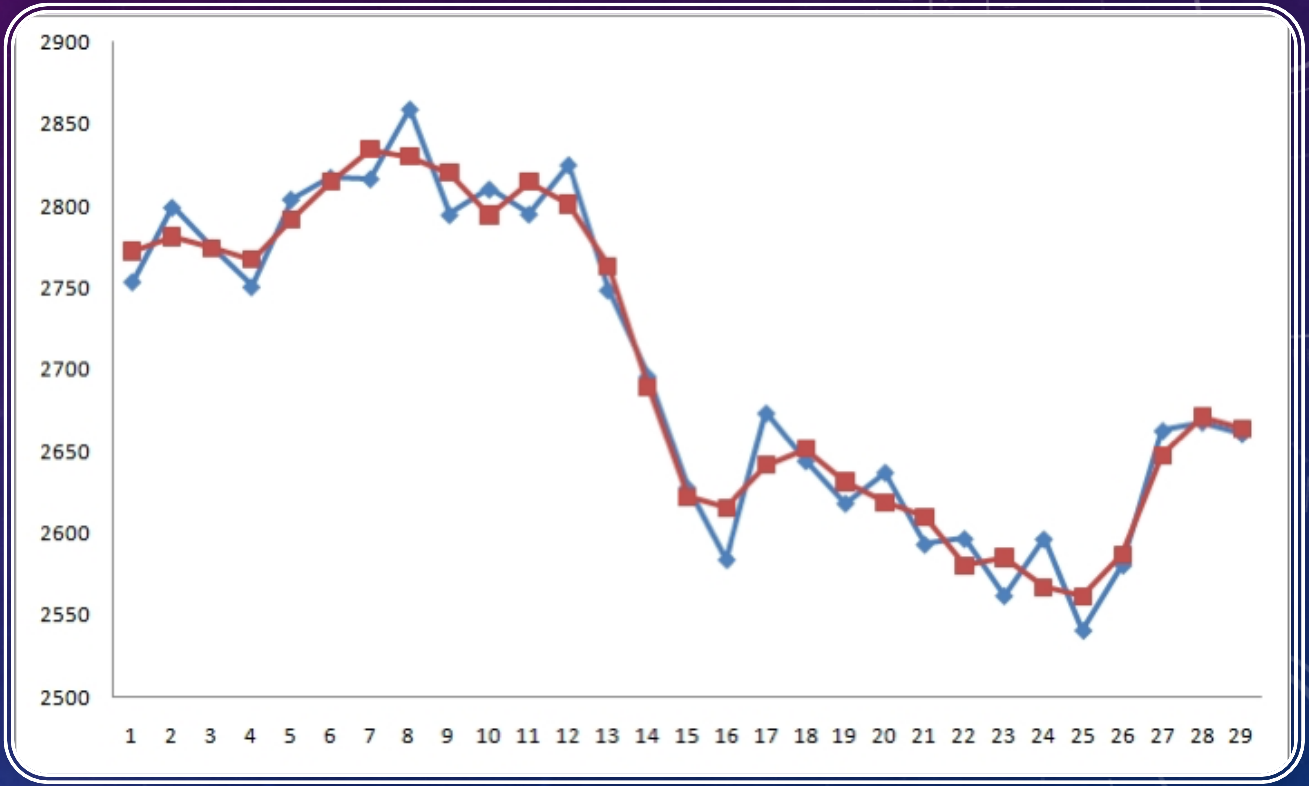
SAVITZKY-GOLAY SMOOTHING FILTER

B649 DATAFLOW SUPERCOMPUTING: ASSIGNMENT 2

Submitted By: Rishabh Monga (rmonga)

INTRODUCTION

- A Savitzky–Golay filter is a digital filter
- it can be applied to a set of digital data points for the purpose of smoothing the data
- to increase the signal-to-noise ratio without greatly distorting the signal.

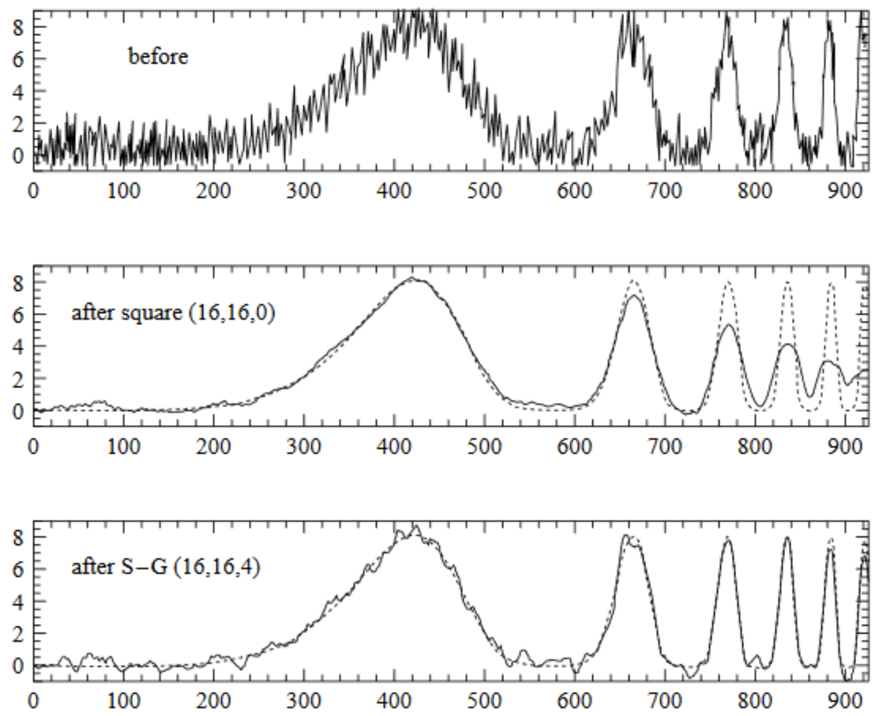


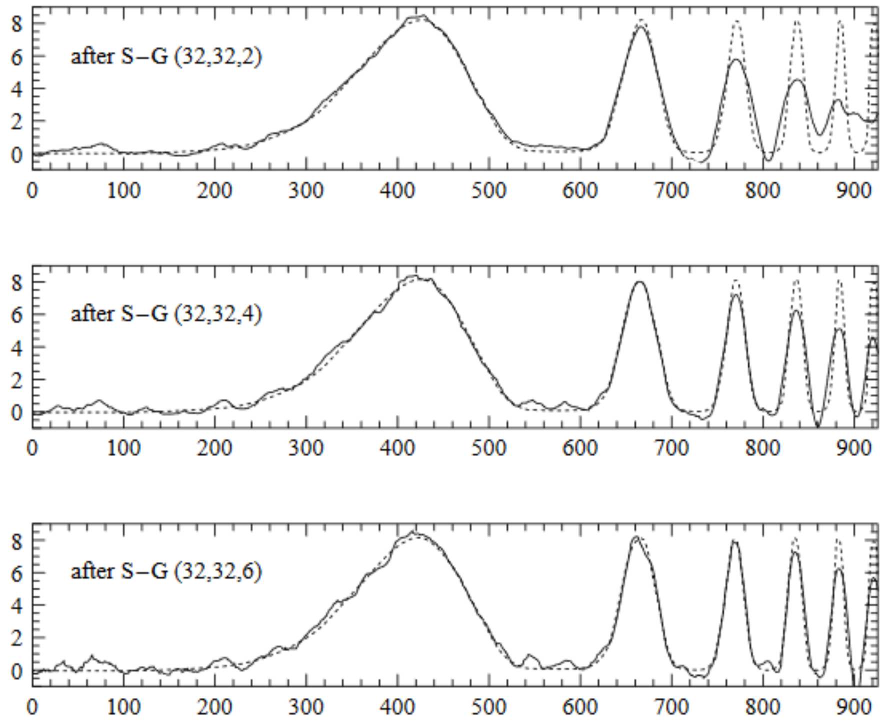
PROPERTIES OF THE FILTER

- The Savitzky-Golay is a type of low-pass filter
- particularly suited for smoothing noisy data
- The main idea behind this approach is to make for each point
  - a least-square fit with a polynomial of high order over a odd-sized window
  - centered at the point.

ADVANTAGES OF THE FILTER

- The Savitzky-Golay filter removes high frequency noise from data.
- It has the advantage of preserving the original shape
- and features of the signal better than other types of filtering approaches
- such as moving averages techniques.


before
after square (16,16,0)
after S–G (16,16,4)


after S–G (32,32,2)
after S–G (32,32,4)
after S–G (32,32,6)

EXAMPLE

EXTRACTING KERNELS

- We need to identify key loops in the algorithm
- That can be transformed into Maxeler kernels
- The c language implementation of Savitzky Golay smoothing filter has 47 loops
    - of which 22 are nested
- So extraction of key loops was challenging

EXTRACTED KERNEL

- I was able to identify a loop that was repeated 7 times inside the nested and doubly loops
- The extracted loop was used to find the distance between matrices

```
package smoothingfilter;

import com.maxeler.maxcompiler.v2.kernelcompiler.Kernel;
import com.maxeler.maxcompiler.v2.kernelcompiler.KernelParameters;
import com.maxeler.maxcompiler.v2.kernelcompiler.types.base.*;
import com.maxeler.maxcompiler.v2.kernelcompiler.stdlib.memory.*;
import com.maxeler.maxcompiler.v2.utils.*;
import com.maxeler.maxcompiler.v2.kernelcompiler.stdlib.core.*;

class SmoothinKernel extends Kernel {

  protected SmoothinKernel(KernelParameters parameters) {
        super(parameters);

        DFEVar x = io.input("x", dfeFloat(8, 24));
        DFEVar sum = io.scalarInput("sum", dfeUInt(32));

        DFEVar prev = stream.offset(x, -1);
        DFEVar next = stream.offset(x, 1);

        DFEVar count = control.count.simpleCounter(32, size);

        DFEVar low = count > 0;
        DFEVar high = count < size - 1;

        DFEVar mid = low & high;

        DFEVar prev = low ? prevOrig : 0;
        DFEVar next = high ? nextOrig : 0;

        DFEVar divisor = mid ? constant.var(dfeFloat(8, 24), 3) : 2;

        DFEVar sum = prev + x + next;
        DFEVar result = sum / divisor;

        io.output("y", result, dfeFloat(8, 24));
    }
}
```

KERNEL MANAGER

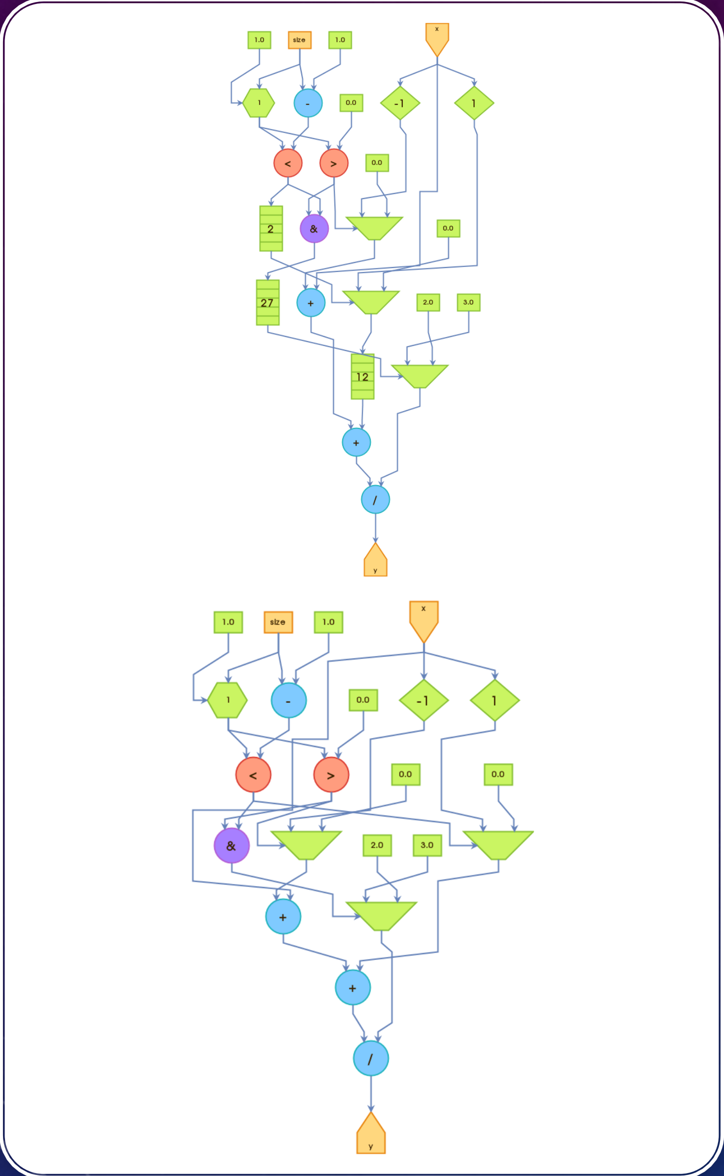
```java
package smoothingfilter;

import com.maxeler.maxcompiler.v2.build.EngineParameters;
import com.maxeler.maxcompiler.v2.kernelcompiler.Kernel;
import com.maxeler.maxcompiler.v2.managers.standard.Manager;
import com.maxeler.maxcompiler.v2.managers.standard.Manager.IOType;

class SmoothinKernelManager {
  public static void main(String[] args) {
    EngineParameters params = new EngineParameters(args);
    Manager manager = new Manager(params);

    // Instantiate the kernel
    Kernel kernel = new SmoothinKernel(manager.makeKernelParameters());

    manager.setKernel(kernel);
    manager.setIO(IOType.ALL_CPU); // Connect all kernel ports to the CPU
    manager.createSLiCinterface();
    manager.build();
  }
}
```



KERNEL GRAPHS

CONCLUSION

- Savitzky and Golay's paper is one of the most widely cited papers in the Analytical Chemistry journal
- it was classed by that journal as one of its "10 seminal papers"
  - saying "it can be argued that the dawn of the computer-controlled analytical instrument can be traced to this article".
- The method has been extended for the treatment of 2- and 3-dimensional data.
- Dataflow techniques can introduce critical improvement in execution of this method
  - As it has numerous loops, with many of these loops nested and doubly nested