A Program to simulate Data Fusion

Amritam Sarcar

September 8, 2008

1 Program

```
import java.util.Random;
import java.io.*;
public class DataFusion {

        /**
         * @param args
         */
        public static void main(String[] args) throws IOException{
                // TODO Auto-generated method stub

                if(args.length!=3){
                        System.out.println("Wrong format!");
                        System.out.println("java DataFusion <Xactual> <n> <N>");
                        System.exit(-1);
                }


                int n,N;
                double Xactual=0,Xnew=0,summation=0,experimentalSigma=0,actualSigma=0;

                // getting the arguments
                Xactual = Double.parseDouble(args[0]);
                n = Integer.parseInt(args[1]);
                N = Integer.parseInt(args[2]);

                DataFusion obj = new DataFusion(); // creating an object

                // var to save values of X
                double[] X = new double[n];

                 // var to save values of sigma for each X
                double[] sigma = new double[n];

                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

                // read individual s.d. inputs for n
                for(int i=0;i<n;i++){
                        // read the inputs from the user
```

```
                try {
                        System.out.print("Input s.d. for " + i + ":: ");
                        sigma[i] = Double.parseDouble(br.readLine());
                } catch (IOException ioe) {
                        System.out.println(
                        "IO error trying to read the standard deviation!");
                        System.exit(1);
                        }
        }
        // compute Xnew for N times
        for(int loop = 0;loop<N;loop++){
                // compute each X
                X = obj.computeX(sigma,Xactual);

                // compute Xnew
                Xnew = obj.computeXnew(sigma,X);

                // summation of the difference square
                summation += Math.pow((Xnew-Xactual),2.0);
        }

        // sigma computed from this experiment
        experimentalSigma = Math.sqrt(summation/N);

        System.out.println("Experimental sigma :: " + experimentalSigma);

        // computing the theoretical value
        for(int i=0;i<sigma.length;i++)
                actualSigma += 1/Math.pow(sigma[i], 2.0);

        actualSigma = Math.sqrt(1/actualSigma);

        System.out.println("Actual sigma :: " + actualSigma);

}

/* Computes Xnew from the given values of x,sigma */
private double computeXnew(double[] sigma, double[] X){

        double Xnew=0, den=0, sum=0;

        // computing the value
        for(int i=0;i<sigma.length;i++){
                den = Math.pow(sigma[i], 2.0);
                Xnew += X[i]/den;
                sum += 1/den;
        }

        return Xnew/sum;
}

/* Computes each X from the given values of Xactual,sigma's */
private double[] computeX(double[] sigma, double Xactual){

        double[] X = new double[sigma.length];

        //computes X
```

```
                for(int i=0;i<sigma.length;i++)
                        X[i] = Xactual + generateGauss() * sigma[i];

                return X;
        }

        /**
         * Generates Gaussian random numbers of length limit
         * @return a Gaussian number
         */
        public double generateGauss(){

                double gauss=0;

                Random gen = new Random();

                for(int j=0;j<12;j++)
                        gauss += gen.nextDouble()-0.5;

                return gauss;
        }

}
```

2 Output

The above program was run several times by varying the parameters. The sample outputs are as follows -

```
java DataFusion 140 4 5000
Input s.d.  for 0::  10
Input s.d.  for 1::  20
Input s.d.  for 2::  30
Input s.d.  for 3::  40
Experimental sigma ::  8.409789324740263
Actual sigma ::  8.381163549234937

java DataFusion 140 4 5000
Input s.d.  for 0::  10
Input s.d.  for 1::  10
Input s.d.  for 2::  10
Input s.d.  for 3::  10
Experimental sigma ::  5.048501934677643
Actual sigma ::  5.0
```

The following table gives a snapshot of various outputs.

SI	X_{actual}	NoofValues(i)	NoOfIterations(N)	σ_{actual}	σ_{expl}
1	140	4	5000	8.3812	8.4098
2	1	2	2000	10.6390	11.0508
3	11	5	10000	1.5053	1.5468
4	11	7	1000	0.9165	0.9296
5	11	10	6000	1.0681	1.0731

Table 1: The different ouputs