FİZ433 FİZİKTE BİLGİSAYAR UYGULAMALARI

(DERS NOTLARI)

Hazırlayan:

Prof.Dr. Orhan ÇAKIR

Ankara Üniversitesi, Fen Fakültesi, Fizik Bölümü

Ankara, 2017

İÇİNDEKİLER

KONU 12

BENZETİM II

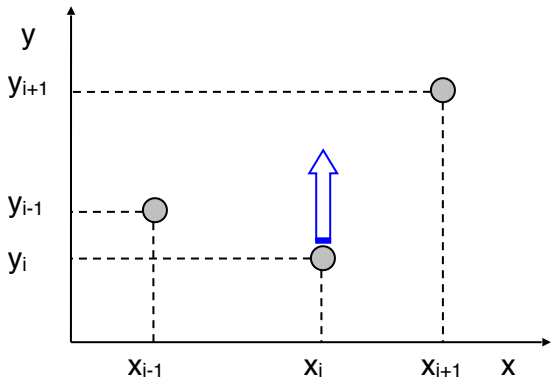
Verilerin Dağılımının Düzgünleştirilmesi

Verilerin düzgünleştirilmesi (smoothing) yöntemi, genellikle fiziksel ölçümlerden çıkarılan gerçek verilerle ilgilenen fen ve mühendislikte kullanılmaktadır. Bütün fiziksel ölçümler deneysel ölçüm hataları içerir, ölçü aletleri kusursuz olmadığı için deneylerle ilişkili simulasyonlar yapmaya çalışıldığında bunların ideal olayları temsil etmediğini görürüz.

Deneyden alınan veriler onların ham şekilde kullanımını sınırlayacak kadar "gürültülü" olabilir. Böyle bir durumda birkaç yol izlenebilir. Bunlar, deneyin iptali, deneyin yeniden tasarlanması veya verileri düzgünleştirecek yöntemlerin bulunması şeklindedir. Burada düzgünleştirme, birçok veri noktasını alarak birçeşit ortalama alma işlemlerini içerir. Eğer gürültü tarafsız bir şekilde oluşuyorsa, hatalar pozitif olabileceği kadar negatif de olabilir. Bu durumda ortalama alma işlemi hataların birbirini yok edeceği etkisini verecektir. Gerçekte veriyi düzgünleştirdiğimiz veya ortalamasını aldığımızda yakın komşu verilerin farkını azalttığımızı düşünürüz. Komşu verilerin bu farklılıkları gürültüden, rastgelelikten kaynaklanabilir veya gerçek de olabilir. Düzgünleştirme olayın doğasına bakılmaksızın yapılan bir işlemdir. Bu işlemin az yapılması gürültüyü azaltacak aynı zamanda verilerdeki değişimin gerçek yapısını ayıracaktır. Burada önemli olan verilerin orijinal yapısını koruyarak kabul edilebilir bir seviyeye kadar gürültüyü azaltmaktır.

Formulasyon

Verileri ortalama alarak düzgünleştirmek en basit yaklaşımdır. N adet kesikli veri noktası $y_i(x_i)$ olduğunu düşünelim, burada i [1:N] bir tamsayıdır. Eski gürültülü veri kümesinden, yeni düzgünleştirilmiş veri kümesi elde etmek istiyoruz. Burada her bir y_i değerini komşu verilerle aynı hizaya gelecek şekilde ayarlarız. Bazı durumlarda x-değerlerinde de gürültü olabilir, burada belli x değerlerinde y_i ölçümlerinin yapıldığını varsayarak sadece y değerlerindeki gürültü ile ilgileniyoruz. Grafiksel olarak yöntem Şekil 7.1'deki gibi uygulanır.



Şekil 7.1 Ölçüm değeri y_i nin düzgünleştirilmesi

Düzgünleştirme işlemi matematiksel olarak aşağıdaki gibi ifade edilebilir. Üç komşu nokta düşünüldüğünde (*i*-1, *i*, *i*+1),

$$y_i = \frac{y_{i-1} + y_i + y_{i+1}}{3} \quad , \quad y_1 = \frac{2y_1 + y_2}{3} \quad , \quad y_n = \frac{2y_n + y_{n-1}}{3}$$

bağıntıları kullanılır. Burada başlangıç ve son noktaların hesabında eski noktaların iki kat değerleri kullanılmıştır.

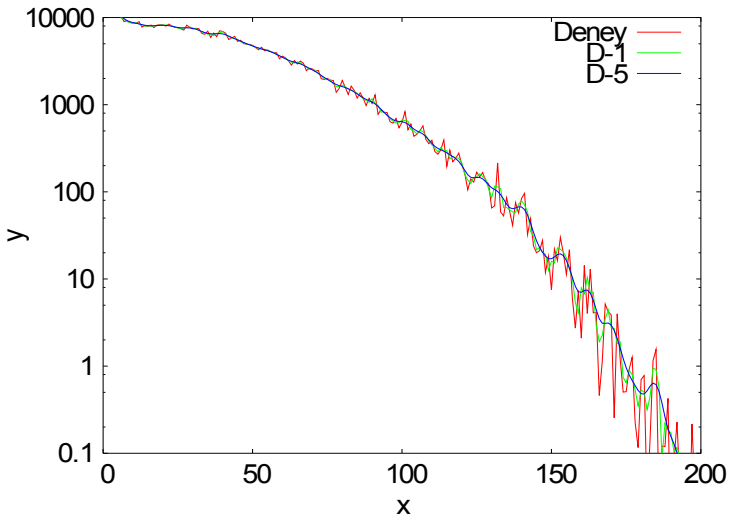
Örnek: Veri dosyasındaki (tab_1.txt) değerleri okuyup düzgünleştirme yapan ve bunu başka bir veri dosyasına (tab_2.txt) yazan bir FORTRAN programı yazalım. Programda veriler yo(n) ve yn(n) dizilerinde saklanmaktadır.

- **FORTRAN program**

```
program duzgun

implicit real*8 (a-h,o-z)

parameter (nmax=300)

dimension yo(nmax),yn(nmax)

open(1,file="tab_1.txt",status="old")

open(2,file="tab_2.txt",status="new")
```

```
indis=0
do i=1,nmax
indis=indis+1
write(*,*)indis
read(1,*)yo(indis)
enddo
print*,"toplam veri sayisi: ",indis
xkat=1.0/3.0
do i=2,indis-1
yn(i)=xkat*(yo(i-1)+yo(i)+yo(i+1))
enddo
yn(1)=xkat*(2.*yo(1)+yo(2))
yn(nmax)=xkat*(2.*yo(nmax)+yo(nmax-1))
do j=1,nmax
write(2,*)yn(j)
enddo
end
```

Şekil 7.2'de simulasyon verilerine göre, bir (D-1) ve beş (D-5) defa düzgünleştirme yönteminin uygulanmasıyla elde edilen verilere göre çizilen grafikler gösterilmiştir.



Şekil 7.2 Deney verileri üzerine bir defa uygulanan düzgünleştirme (D-1) ve beş defa uygulanan düzgünleştirme (D-5)

Örnek Problemler:

- Radyoaktif bozunma için basit bir fortran simulasyon programı yazınız.

```
Program simradec

Open(5,file="simradec.txt")

N0=100

Nn=N0

Alf=0.01

Idt=1

Itmin=1

Itmax=300

Do 10 it=itmin,itmax

T=it*1.

Do 20 in=0,nn

Xran=rand()

Dt=1.*idt

If(xran<dt*alf) then
```

```
          Nn=nn-1
          Endif
          Xnt=1.*n0*exp(-alf*t)
20        Continue
          Write(5,*)t,nn,xnt
10        Continue
          end
```

• Monte Carlo yöntemi ile Integral hesaplayan bir C programı

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#include<math.h>

void main(void);
double m_carlo(double a,double b,double h,long n);
double f(double x);

      void main(void){
          double a,b;
          double h;
          long n;
          double integral;
          time_t curr_time;

printf("\n \t Monte Carlo Integration \n");
/*printf("Enter lower limit of integral: ");
scanf("%f",&a);
printf("Enter upper limit of integral: ");
scanf("%f",&b);
printf("Enter height of bounding rectangle: ");
scanf("%f",&h);
printf("Enter number of trials: ");
scanf("%3d",&n);*/

a=0.; b=3.; h=9.; n=1000000;
printf("\n Number of events: %d",n);

integral=m_carlo(a,b,h,n);

printf("\n Result of the integral: %1f \n",integral);
printf("\t  End \n");
}

/* generate random numbers */
```

```
 double m_carlo(double a,double b,double h,long n){
 double result;
 double x,y;
 long count=0;
 long i;

 for (i=1;i<=n; i++){
  x=((float) rand()/RAND_MAX)*abs(b-a)+a;
  y=((float) rand()/RAND_MAX)*h;

  if (y<f(x))
    ++count;
 }
 result=(b-a)*h*(float)count/n;

 return(result);
}

/* function to integrate */

 double f(double x){
 double fx;
 fx=x*x;
 return(fx);
}
```