

Histogram

1. V ROOTu vytvořte histogram z naměřených hodnot uložených v souboru `data.dat`.
Nalezněte optimální šířku binu.

Práce s histogramy v ROOTu:

- vytvoření histogramu (constructor)

```
TH1D *hist = new TH1D("hist", "histogram", nbins, x_min, x_max);
```

typ objektu (1D histogram)

ukazatel na histogram
(objekt v paměti)

jméno
histogramu

nadpis

počet binů

minimum

maximum

- metody objektu TH1D

přidání prvku x_i do histogramu

```
hist-> Fill(x[i]);
```

nakreslení histogramu

```
hist -> Draw();
```

Histogram

histogram2.c

```
//histogram
#define ndata 1000 //pocet dat
//define nbins 100 //pocet binu

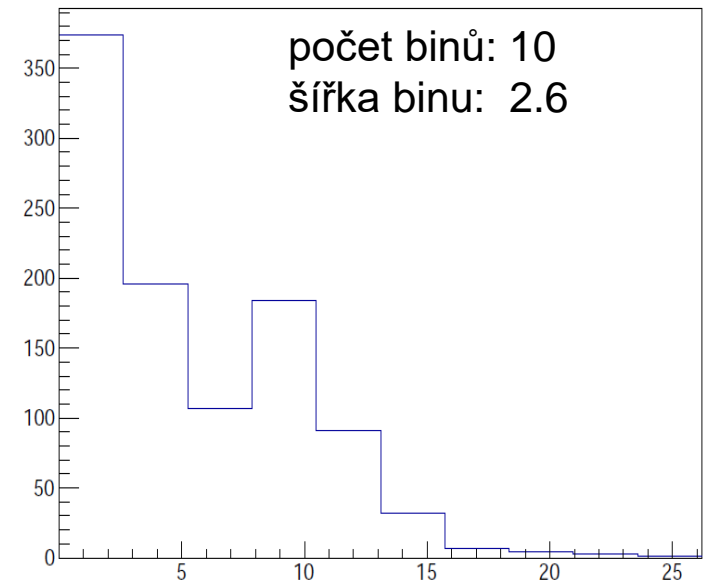
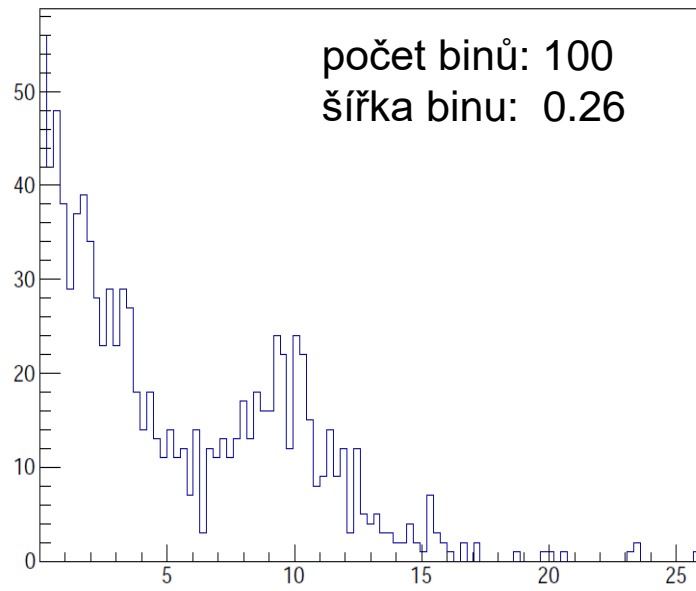
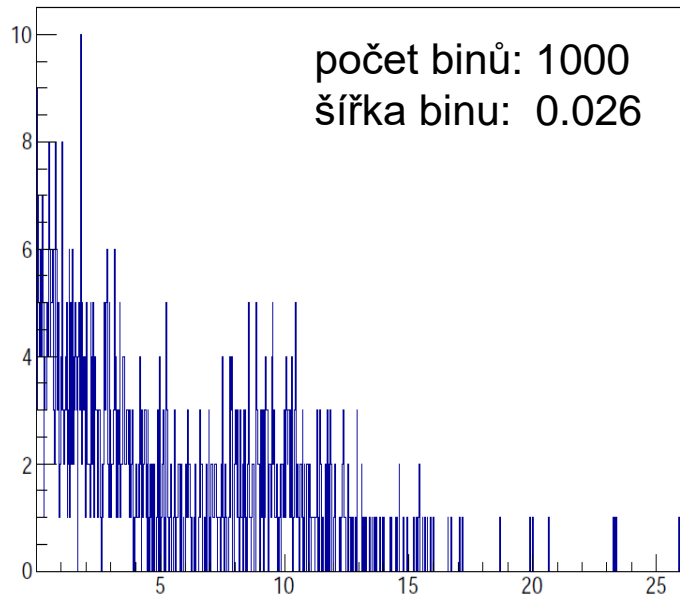
int run(int nbins) //pocet binu promenna funkce "run"
{
    FILE *fin,*fout; //soubory odkud se prectou a kam se zapisi data
    double x[ndata];
    double x_min,x_max; //minimalni a maximalni hodnota - meze histogramu
    fin=fopen("data.dat","r"); //vstupni soubor s daty - pro cteni "r"
    fout=fopen("histogram.txt","w"); //vystupni soubor s daty - pro zapsani "w"
    for (int i=0; i<ndata; i++)
    {
        fscanf(fin,"%lf",&x[i]); //nacteni dat
        if(i==0) x_min=x_max=x[i]; //aby nebyly nedefinovane meze
        if(x[i]<x_min) x_min=x[i]; //hledani minima
        if(x[i]>x_max) x_max=x[i]; //hledani maxima
    }
    fclose(fin);
    printf("min value: %lf\n",x_min);
    printf("max value: %lf\n",x_max);
    printf("bin width: %lf\n",(x_max-x_min)/nbins);
    TH1D *hist = new TH1D("hist", "histogram", nbins, x_min, x_max); //definice histogramu
    for (int i=0; i<ndata; i++)
        hist->Fill(x[i]); //naplneni histogramu
    TCanvas *c1 = new TCanvas("c1", "histogram",10,10,800,600); //definice okna
    hist->Draw(); //nakresleni histogramu
    for (int i=0; i<nbins; i++) //zapis histogramu do souboru fout
    {
        counts=hist->GetBinContent(i); //zjisteni hodnoty v i-tem binu
        fprintf(fout,"%ld %ld\n",i,counts); //zapis hodnoty do souboru fout
    }
    fclose(fout);
    return(0);
}
```

Histogram

1. V ROOTu vytvořte histogram z naměřených hodnot uložených v souboru `data.dat`.
Nalezněte optimální šířku binu.

počet dat: 1000
minimum: 0.004
maximum: 26.198

`histogram.c`



Histogram + Monte Carlo simulace

2. Hodnoty v souboru `data.dat` odpovídají situaci, kdy v experimentu mohou být detekovány dva typy událostí:


- v $\frac{3}{4}$ případů exponenciální rozpad s časovou konstantou 4
- v $\frac{1}{4}$ případů výběr z normálního rozdělení s očekávanou hodnotou 10 a standardní odchylkou 2

V ROOTu proveďte simulaci tohoto experimentu a nasimulovaná data uložte do histogramu.

Generátor náhodných čísel v ROOTu:

- vytvoření generátoru náhodných čísel (constructor objektu) perioda $2^{19937}-1$

```
TRandom3 *gRandom = new TRandom3();
```


typ objektu (generátor náhodných čísel)

ukazatel na generátor
(objekt v paměti)

- generování náhodného čísla $U(0,1)$

```
x[i]=gRandom -> Rndm();
```

Další rozdělení:

```
gRandom -> Exp(tau)
```

```
gRandom -> Gaus(mean, sigma)
```

```
gRandom -> Poisson(mean)
```

```
gRandom -> Binomial(n, p)
```

Histogram + Monte Carlo simulace

hist-simulate2.c

```
//generator nahodnych cisel z rozdeleni exp + norm
#define nmax 1000 //pocet hodnot
//define nbins 100 //pocet binu
#define tau 4 //parametr exponencialniho rozdeleni (casova konstanta)
#define mu 10 //ocekavana hodnota normalniho rozdeleni
#define sigma 2 //standardni odchylka normalniho rozdeleni
#define P_exp 0.75 //pravdepodobnost ze to bude exponencialni rozdeleni
int simulate(int nbins) //pocet hodnot a binu promenne funkce "simulate"
{
    FILE *fout; //soubor kam se zapisi data
    double x[nmax];
    double branch;
    double x_min,x_max; //minimalni a maximalni hodnota - meze histogramu
    TRandom3 *gRandom = new TRandom3(); //vytvoreni generatoru nahodnych cisel
    fout=fopen("data.dat","w"); //vystupni soubor s daty - pro zapsani "w"
    for (int i=0; i<nmax; i++)
    {
        branch=gRandom->Rndm(); //exponencialni nebo normalni rozdeleni?
        if(branch<P_exp) x[i]=gRandom->Exp(tau); //generovani nahodneho cisla E(tau)
        else x[i]=gRandom->Gaus(mu,sigma); //generovani nahodneho cisla N(mu,sigma)
        if(i==0) x_min=x_max=x[i]; //aby nebyly nedefinovane meze
        if(x[i]<x_min) x_min=x[i]; //hledani maxima
        if(x[i]>x_max) x_max=x[i]; //hledani minima
        fprintf(fout,"%lf\n",x[i]); //zapis dat do souboru fout
    }
    printf("min value: %lf\n",x_min);
    printf("max value: %lf\n",x_max);
    TH1D *hist = new TH1D("data", "exp+gauss", nbins, x_min, x_max); //definice histogramu
    for (int i=0; i<nmax; i++)
        hist->Fill(x[i]); //naplneni histogramu
    TCanvas *c1 = new TCanvas("c1", "histogram",5,5,800,600); //definice okna
    hist->Draw(); //nakresleni histogramu
    fclose(fout);
    return(0);
}
```