

Distribuzioni di probabilità

- Note

```
filename: distribuzioni.mws
version: 2.0
  tested: Maple V Release 6.02 on Windows 2000
    date: 15 marzo 2003
  author: Claudio Marsan
    Liceo Cantonale di Mendrisio
    Via Agostino Maspoli
    CH-6850 Mendrisio
    e-mail: claudio.marsan@liceomendrisio.ch
```

```
> restart:
```

```
> graphics_options := style=LINE, color=BLUE, thickness=6,
  labels=[``, ``]:
```

- Distribuzione binomiale

- Modello

Consideriamo l'esperimento che consiste nel lanciare n volte una moneta per la quale la probabilità che esca "testa" sia p ; sia X la variabile aleatoria associata al numero k di "teste" uscite. Allora la distribuzione di $P(X=k)$ è una distribuzione binomiale che possiamo calcolare con la funzione `bin_distr(n,p,k)`.

- Definizione della funzione per la distribuzione binomiale

```
> bin_distr := proc(n::posint, p, k::nonnegint)
  local p0;
  p0 := convert(p, float);
  if k>n then
    ERROR(`Il 3° argomento non può essere maggiore del
1°!`);
  end if;
  if p0=0.0 then
    RETURN(0.0);
  end if;
  if p0=1.0 then
    RETURN(1.0);
  end if;
  if (p0<0.0 or p0>1.0) then
    ERROR(`Il 2° argomento deve essere in [0,1]!`);
  end if;
  evalf(binomial(n,k) * p0^k * (1-p0)^(n-k));
end;
```

```

bin_distr := proc(n::posint, p, k::nonnegint)
local p0;
  p0 := convert(p, float);
  if n < k then ERROR( `Il 3° argomento non può essere maggiore del 1°!` ) end if
  ;
  if p0 = 0. then RETURN(0.) end if;
  if p0 = 1.0 then RETURN(1.0) end if;
  if p0 < 0. or 1.0 < p0 then ERROR( `Il 2° argomento deve essere in [0,1]!` )
  end if;
  evalf(binomial(n, k)*p0^k*(1 - p0)^(n - k))
end proc

```

– Definizione della funzione per la rappresentazione grafica della distribuzione binomiale

```

> plot_bin_distr := proc(n::posint, p)
  local S, i;
  S := seq([[i, 0], [i, bin_distr(n, p, i)]], i=0..n);
  plot([S], x=0..n, graphics_options);
end;

plot_bin_distr := proc(n::posint, p)
local S, i;
  S := seq([[i, 0], [i, bin_distr(n, p, i)]], i = 0 .. n);
  plot([S], x = 0 .. n, graphics_options)
end proc

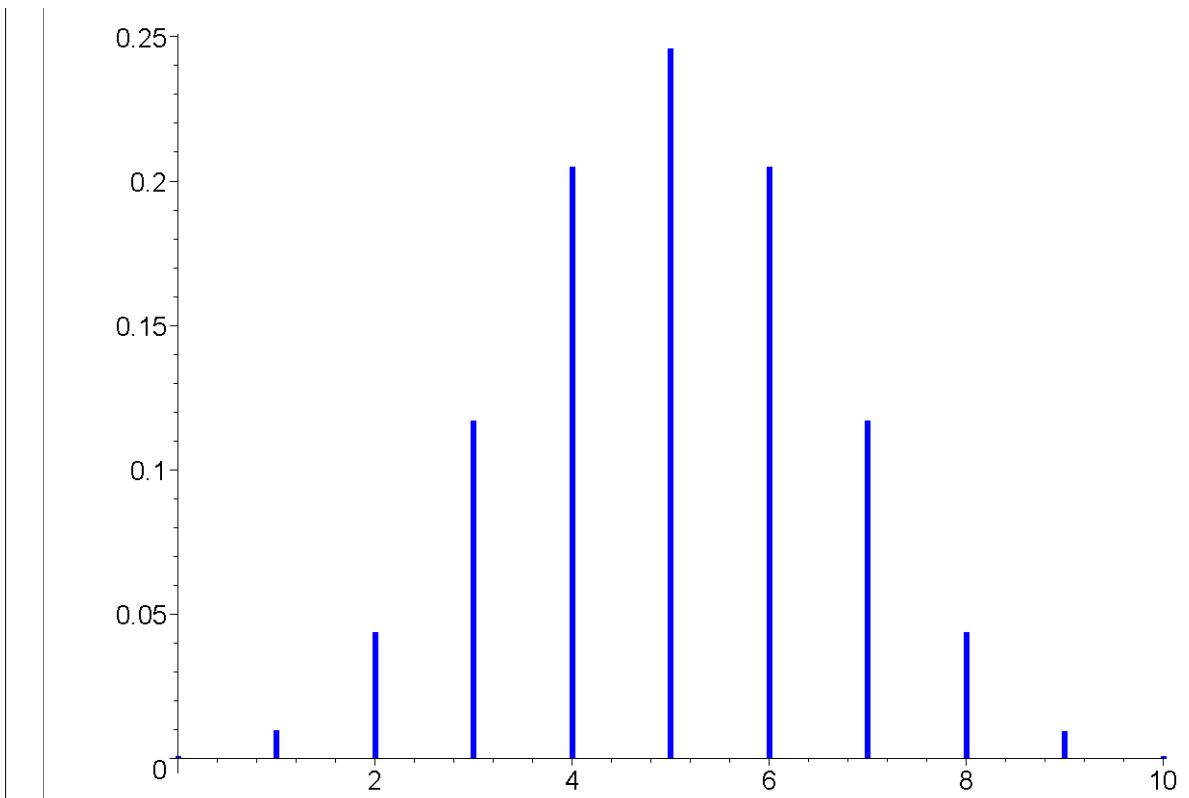
```

– Esempio 1: successione di 10 prove, ognuna con probabilità $p=0.5$ di successo

```

> for i from 0 to 10 do
  bin_distr(10, 0.5, i);
end do;
> plot_bin_distr(10, 0.5);
.0009765625
.0097656250
.0439453125
.1171875000
.2050781250
.2460937500
.2050781250
.1171875000
.0439453125
.0097656250
.0009765625

```



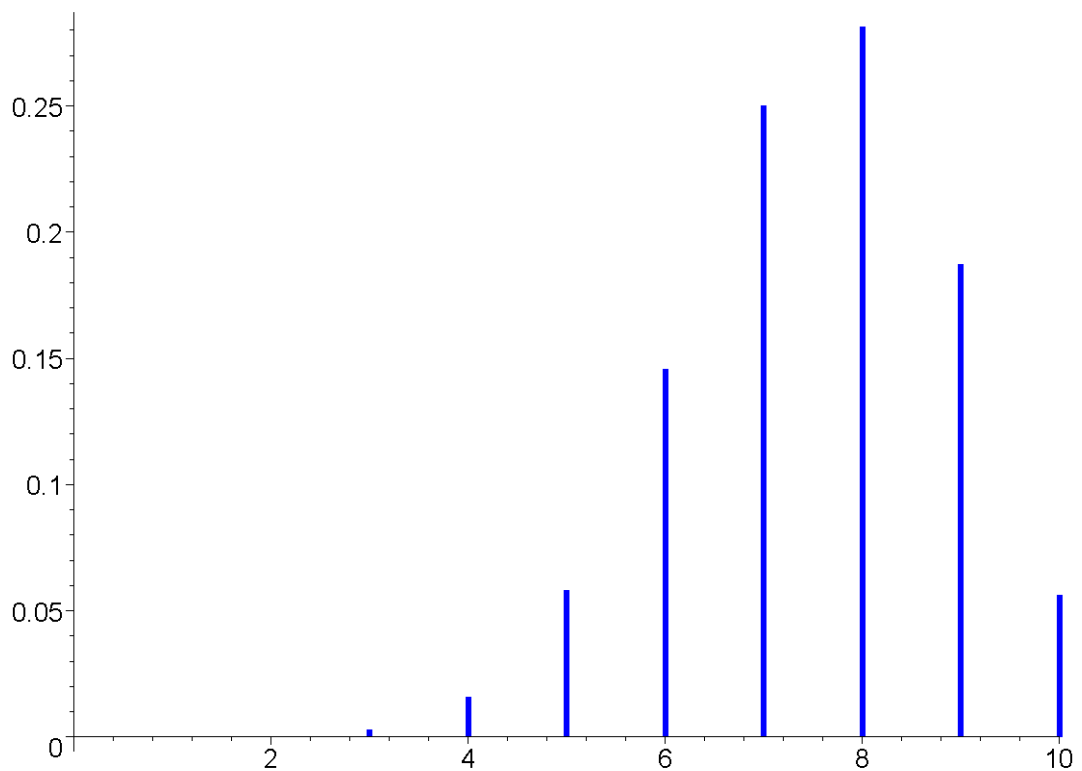
— Esempio 2: successione di 10 prove, ognuna con probabilità $p=0.25$ di successo

```

> for i from 0 to 10 do
  bin_distr(10, 0.25, i);
end do;
> plot_bin_distr(10, 0.25);

```

.05631351471
.1877117157
.2815675734
.2502822876
.1459980011
.05839920044
.01622200012
.003089904785
.0003862380981
.00002861022950
.9536743164 10⁻⁶



- Distribuzione geometrica

- Modello

Lanciamo una moneta ("testa" può uscire con probabilità p) finché non esce "testa" per la prima volta; sia X la variabile aleatoria associata al numero di lanci n necessari affinché "testa" esca per la prima volta. Allora la distribuzione di $P(X=n)$ è una distribuzione geometrica che possiamo calcolare con la funzione `geom_distr(n,p)`.

- Definizione della funzione per la distribuzione geometrica

```
> geom_distr := proc(n::posint, p)
  local p0;
  p0 := convert(p, float);
  if p0=0.0 then
    RETURN(0.0);
  end if;
  if p0=1.0 then
    if n=1 then
      RETURN(1.0);
    else
      RETURN(0.0);
    end if;
  end if;
  if (p0<0.0 or p0>1.0) then
    ERROR(`Il 2° argomento deve essere in [0,1]!`);
  end if;
  evalf((1-p0)^(n-1) * p0);
end proc;
```

```

end;
geom_distr := proc(n::posint, p)
local p0;
  p0 := convert(p, float);
  if p0 = 0. then RETURN(0.) end if;
  if p0 = 1.0 then if n = 1 then RETURN(1.0) else RETURN(0.) end if end if;
  if p0 < 0. or 1.0 < p0 then ERROR( `Il 2° argomento deve essere in [0,1]!` )
  end if;
  evalf((1 - p0)^(n - 1)*p0)
end proc

```

— Definizione della funzione per la rappresentazione grafica della distribuzione geometrica

```

> plot_geom_distr := proc(n::posint, p)
  local S, i;
  S := seq([[i, 0], [i, geom_distr(i, p)]], i=1..n);
  plot([S], x=0..n, graphics_options);
end;

plot_geom_distr := proc(n::posint, p)
local S, i;
  S := seq([[i, 0], [i, geom_distr(i, p)]], i = 1 .. n);
  plot([S], x = 0 .. n, graphics_options)
end proc

```

— Esempio 1: distribuzione geometrica con $p=0.5$

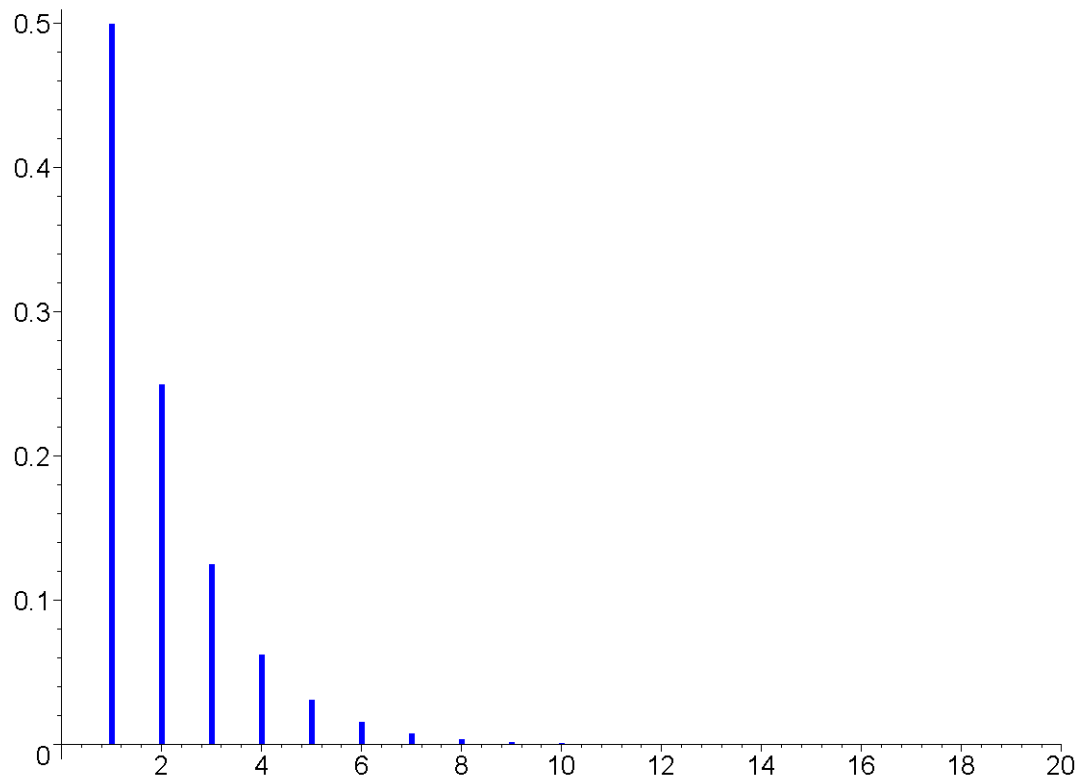
```

> for i from 1 to 20 do
  geom_distr(i, 0.5);
end do;
> plot_geom_distr(20, 0.5);

      .5
      .25
      .125
      .0625
      .03125
      .015625
      .0078125
      .00390625
      .001953125
      .0009765625
      .00048828125
      .000244140625
      .0001220703125

```

```
.00006103515625  
.00003051757812  
.00001525878906  
.7629394530 10-5  
.3814697266 10-5  
.1907348633 10-5  
.9536743165 10-6
```

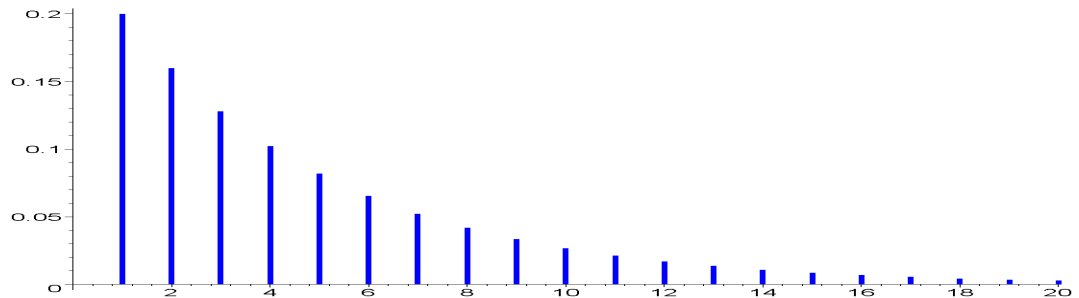


- Esempio 2: distribuzione geometrica con $p=0.2$

```
> for i from 1 to 20 do  
  geom_distr(i, 0.2);  
end do;  
> plot_geom_distr(20, 0.2);
```

```
.2  
.16  
.128  
.1024  
.08192  
.065536  
.0524288  
.04194304  
.033554432  
.0268435456
```

```
.02147483648  
.01717986918  
.01374389535  
.01099511628  
.008796093022  
.007036874418  
.005629499534  
.004503599628  
.003602879702  
.002882303762
```



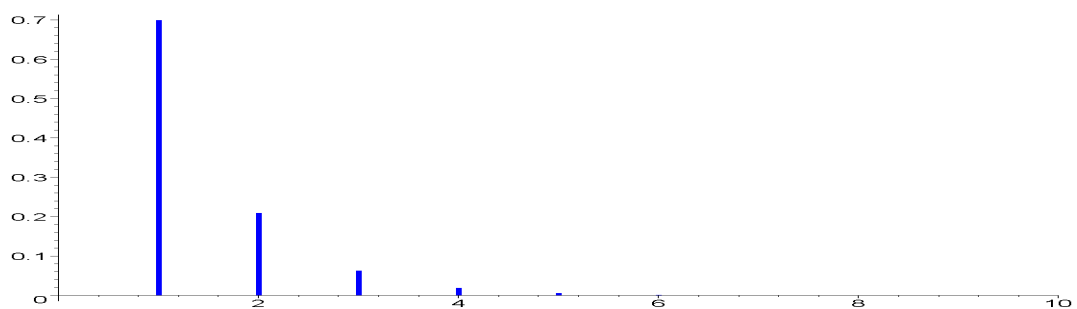
- Esempio 3: distribuzione geometrica con $p=0.7$

```
> for i from 1 to 10 do  
  geom_distr(i, 0.7);  
end do;  
> plot_geom_distr(10, 0.7);
```

```
.7  
.21  
.063  
.0189  
.00567  
.001701  
.0005103  
.00015309
```

.000045927

.0000137781



- Distribuzione binomiale negativa (o di Pascal)

- Modello

Lanciamo una moneta ("testa" può uscire con probabilità p) finché non esce "testa" per la k -esima volta; sia X la variabile aleatoria associata al numero di lanci n necessari affinché "testa" esca per la k -esima volta. Allora la distribuzione di $P(X=n)$ è una distribuzione binomiale negativa che possiamo calcolare con la funzione `neg_bin_distr(n,k,p)`.

- Definizione della funzione per la distribuzione binomiale negativa

```
> neg_bin_distr := proc(n::posint, k::posint, p)
  local p0;
  p0 := convert(p, float);
  if k>n then
    RETURN(0.0);
  end if;
  if p0=0.0 then
    RETURN(0.0);
  end if;
  if p0=1.0 then
    if n=k then
      RETURN(1.0);
    else
      RETURN(0.0);
    end if;
  end if;
```

```

    end if;
    if (p0<0.0 or p0>1.0) then
        ERROR(`Il 3° argomento deve essere in [0,1]!`);
    end if;
    evalf(binomial(n-1, k-1) * p0^k * (1-p0)^(n-k));
end;

neg_bin_distr := proc(n::posint, k::posint, p)
local p0;
    p0 := convert(p, float);
    if n < k then RETURN(0.) end if;
    if p0 = 0. then RETURN(0.) end if;
    if p0 = 1.0 then if n = k then RETURN(1.0) else RETURN(0.) end if end if;
    if p0 < 0. or 1.0 < p0 then ERROR(`Il 3° argomento deve essere in [0,1]!`)
    end if;
    evalf(binomial(n - 1, k - 1)*p0^k*(1 - p0)^(n - k))
end proc

```

– Definizione della funzione per la rappresentazione grafica della distribuzione binomiale negativa

```

> plot_neg_bin_distr := proc(n::posint, k::posint, p)
    local S, i;
    S := seq([[i, 0], [i, neg_bin_distr(i, k, p)]], i=1..n);
    plot([S], x=0..n, graphics_options);
end;

plot_neg_bin_distr := proc(n::posint, k::posint, p)
local S, i;
    S := seq([[i, 0], [i, neg_bin_distr(i, k, p)]], i = 1 .. n);
    plot([S], x = 0 .. n, graphics_options)
end proc

```

– Esempio 1: $n=15, k=3, p=0.3$

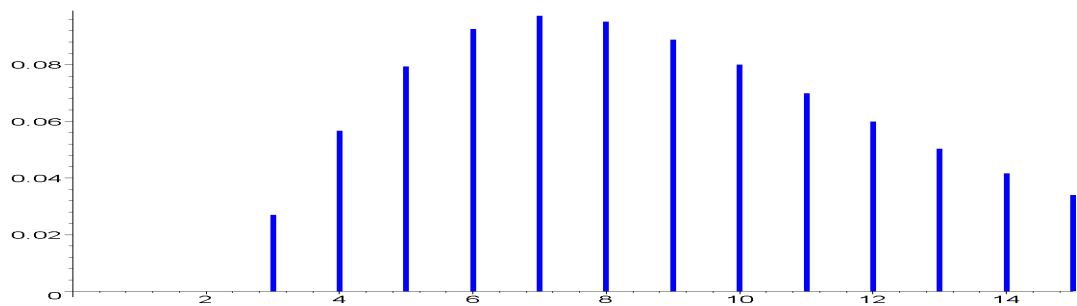
```

> for i from 1 to 15 do
    neg_bin_distr(i, 3, 0.3);
end do;
> plot_neg_bin_distr(15, 3, 0.3);

```

0.
 0.
 .027
 .0567
 .07938
 .092610
 .0972405

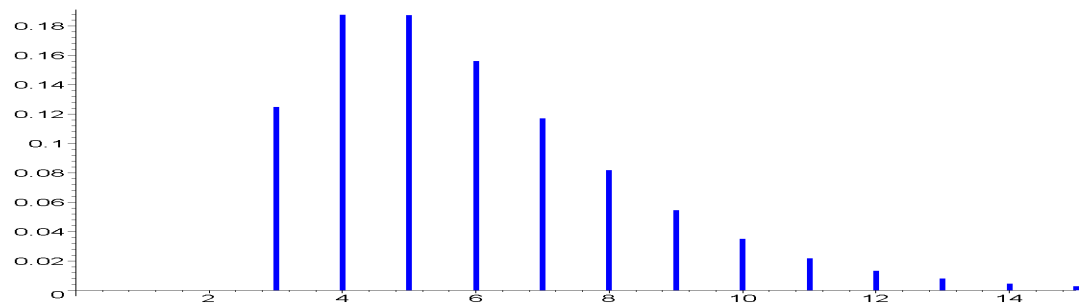
```
.09529569  
.088942644  
.0800483796  
.07004233215  
.05992510640  
.05033708937  
.04164250121  
.03400804265
```



— Esempio 2: $n=15, k=3, p=0.5$

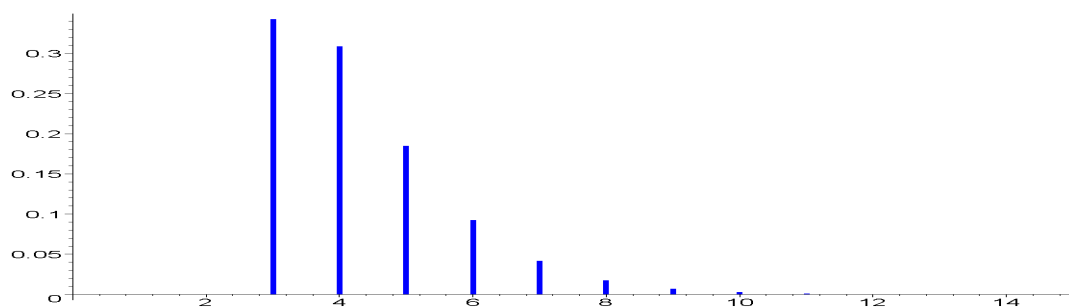
```
> for i from 1 to 15 do  
  neg_bin_distr(i,3,0.5);  
end do;  
> plot_neg_bin_distr(15,3,0.5);  
0.  
0.  
.125  
.1875  
.18750  
.156250  
.1171875  
.08203125  
.054687500  
.0351562500
```

```
.02197265625  
.01342773438  
.008056640625  
.004760742188  
.002777099609
```



— Esempio 3: $n=15, k=3, p=0.7$

```
> for i from 1 to 10 do  
  neg_bin_distr(i,3,0.7);  
end do;  
> plot_neg_bin_distr(15,3,0.7);  
0.  
0.  
.343  
.3087  
.18522  
.092610  
.0416745  
.01750329  
.007001316  
.0027005076
```



- Distribuzione di Poisson

- Modello

Possiamo approssimare una distribuzione binomiale con n grande e p piccolo mediante una distribuzione di Poisson di parametro $\lambda = n p$. Se X è la variabile aleatoria associata al numero di successi k che si verificano in n prove allora la distribuzione di $P(X=k)$ può essere approssimata dalla distribuzione di Poisson che possiamo calcolare con la funzione `poisson_distr(lambda, k)`.

- Definizione della funzione per la distribuzione di Poisson

```
> poisson_distr := (lambda::positive, k::nonnegint) ->
  evalf(lambda^k/k! * exp(-lambda));
```

$$poisson_distr := (\lambda::positive, k::nonnegint) \rightarrow evalf\left(\frac{\lambda^k e^{(-\lambda)}}{k!}\right)$$

- Definizione della funzione per la rappresentazione grafica della distribuzione di Poisson

```
> plot_poisson_distr := proc(n::posint, lambda::positive)
  local S, i;
  S := seq([[i, 0], [i, poisson_distr(lambda, i)]],
    i=0..n);
  plot([S], x=0..n, graphics_options);
end;
```

```
plot_poisson_distr := proc(n::posint, lambda::positive)
```

```
local S, i;
```

```

S := seq([[i, 0], [i, poisson_distr( $\lambda$ , i)]], i = 0 .. n);
plot([S], x = 0 .. n, graphics_options)
end proc

```

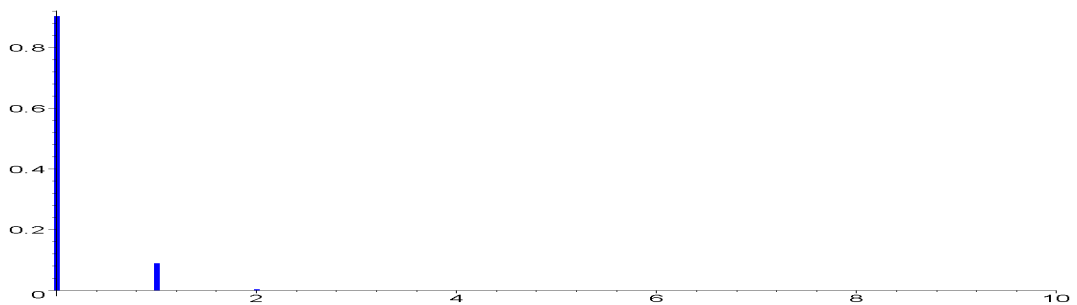
Esempio 1: distribuzione di parametro $\lambda = .1$

```

> for i from 0 to 10 do
  poisson_distr(0.1, i);
end do;
> plot_poisson_distr(10, 0.1);

```

.9048374180
.09048374180
.004524187090
.0001508062364
.3770155909 10⁻⁵
.7540311816 10⁻⁷
.1256718636 10⁻⁸
.1795312337 10⁻¹⁰
.2244140421 10⁻¹²
.2493489357 10⁻¹⁴
.2493489357 10⁻¹⁶



Esempio 2: distribuzione di parametro $\lambda = 1.0$

```

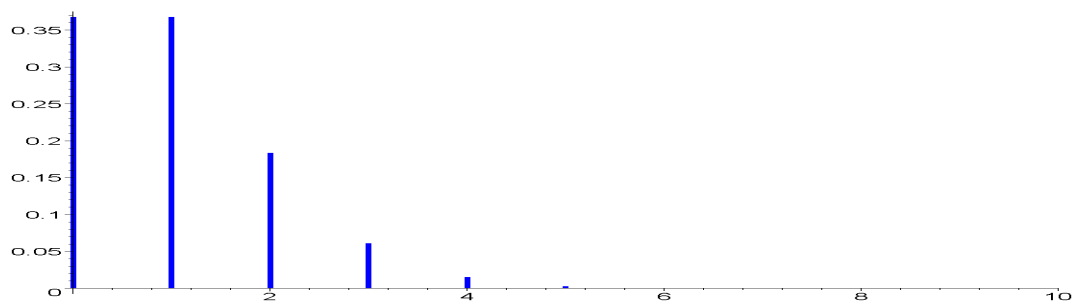
> for i from 0 to 10 do
  poisson_distr(1.0, i);
end do;

```

```

end do;
> plot_poisson_distr(10, 1.0);
.3678794412
.3678794412
.1839397206
.06131324021
.01532831005
.003065662010
.0005109436684
.00007299195261
.9123994077 10-5
.1013777120 10-5
.1013777120 10-6

```



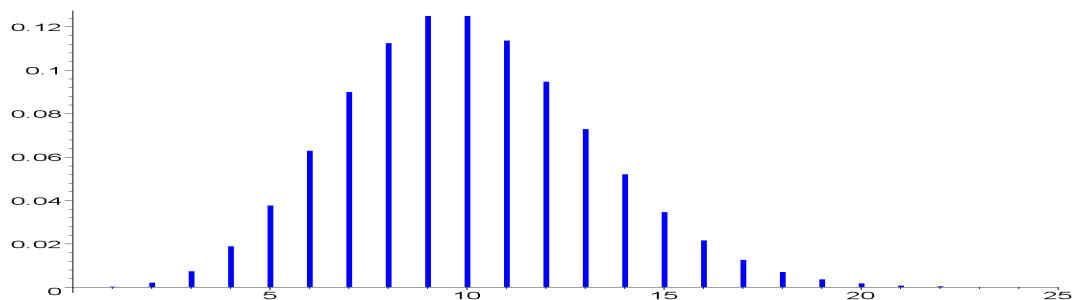
Esempio 3: distribuzione di parametro $\lambda = 10.0$

```

> for i from 0 to 10 do
  poisson_distr(10, i);
end do;
> plot_poisson_distr(25, 10.0);
.00004539992976
.00004539992976
.002269996488
.007566654962

```

.01891663740
.03783327480
.06305545801
.09007922571
.1125990321
.1251100357
.1251100357



- Distribuzione ipergeometrica

- Modello

Consideriamo un'urna che contiene N palline di cui k sono rosse e $N-k$ sono blu. Si estraggono, senza reimbussolamento, n palline dall'urna. Sia X la variabile aleatoria associata al numero x di palline rosse presenti fra le n scelte. La distribuzione di probabilità di $P(X=x)$ è una distribuzione ipergeometrica che possiamo calcolare con la funzione `hypergeom_distr(N,k,n,x)`.

- Definizione della funzione per la distribuzione ipergeometrica

```
> hypergeom_distr :=  
proc(N::posint, k::nonnegint, n::posint, x::nonnegint)  
  if k>N then  
    ERROR(`Il 2° argomento non può essere maggiore del  
1°!`);  
  end if;  
  if n>N then  
    ERROR(`Il 3° argomento non può essere maggiore del
```

```

1°!`);
  end if;
  if x>n then
    ERROR(`Il 4° argomento non può essere maggiore del
3°!`);
  end if;
  evalf(binomial(k,x) * binomial(N-k,n-x) /
binomial(N,n));
end;

```

```

hypergeom_distr := proc(N::posint, k::nonnegint, n::posint, x::nonnegint)
  if N < k then ERROR( `Il 2° argomento non può essere maggiore del 1°!` ) end if
  ;
  if N < n then ERROR( `Il 3° argomento non può essere maggiore del 1°!` )
  end if;
  if n < x then ERROR( `Il 4° argomento non può essere maggiore del 3°!` ) end if
  ;
  evalf(binomial(k,x)*binomial(N-k,n-x) / binomial(N,n))
end proc

```

— Definizione della funzione per la rappresentazione grafica della distribuzione ipergeometrica

```

> plot_hypergeom_distr := proc(N::posint, k::nonnegint,
n::posint)
  local S, i;
  S := seq([[i, 0], [i, hypergeom_distr(N,k,n,i)]],
i=0..n);
  plot([S], x=0..n, graphics_options);
end;

plot_hypergeom_distr := proc(N::posint, k::nonnegint, n::posint)
local S, i;
  S := seq([[i, 0], [i, hypergeom_distr(N, k, n, i)]], i = 0 .. n);
  plot([S], x = 0 .. n, graphics_options)
end proc

```

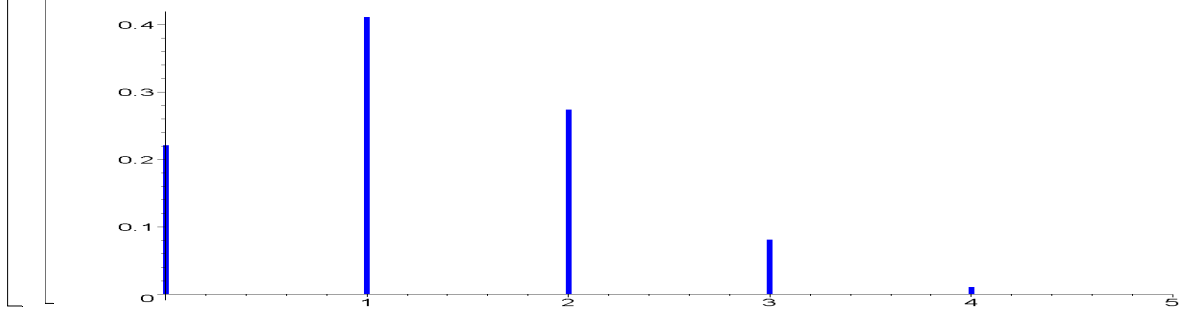
— Esempio 1: $N=52, k=13, n=5$

```

> for i from 0 to 5 do
  hypergeom_distr(52, 13, 5, i);
end do;
> plot_hypergeom_distr(52, 13, 5);
.2215336134
.4114195678
.2742797119
.08154261705

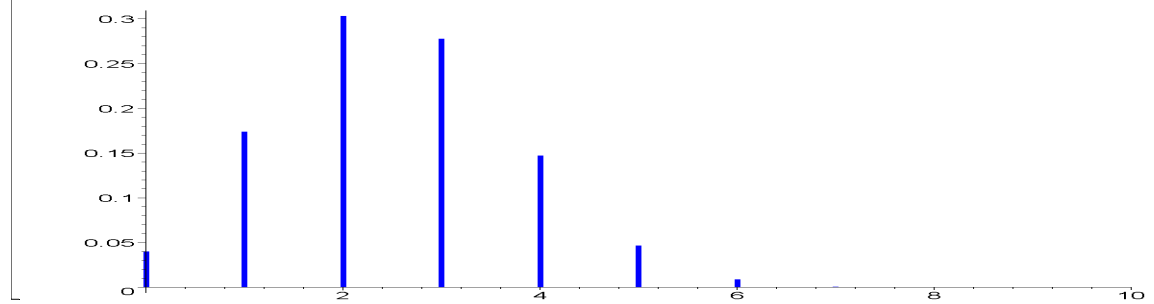
```

.01072929172
.0004951980792



— Esemplio 2: $N=52, k=13, n=10$

```
> for i from 0 to 10 do  
  hypergeom_distr(52, 13, 10, i);  
end do;  
> plot_hypergeom_distr(52, 13, 10);  
.04018612027  
.1741398545  
.3033403917  
.2780620257  
.1474571349  
.04683932519  
.008921776227  
.0009913084697  
.00006028227181  
.1762639526 10-5  
.1807835412 10-7
```



— Esemplio 3: $N=8, k=5, n=5$

```
> for i from 0 to 5 do
  hypergeom_distr(8, 5, 5, i);
end do;
> plot_hypergeom_distr(8, 5, 5);
0.
0.
.1785714286
.5357142857
.2678571429
.01785714286
```

