Moivre-Laplace und Stetigkeitskorrektur

Abstract:
Vorstellung und Veranschaulichung des Satzes (mit wxMaxima), Stetigkeitskorrektur, Beispiel für seine Benutzung

Moivre-Laplace Theorem

Sei $b_{n,p}(k) = \binom{n}{k} p^k (1-p)^{n-k}$ die Dichtefunktion einer Binomialverteilung mit $\mu = np$ und $\sigma^2 = np(1-p)$ dann gilt

$$\lim_{n\to\infty} b_{n,p}(k) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(k-\mu)^2}{2\sigma^2}\right] = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{k-\mu}{\sigma}\right)^2} = \varphi_{\mu,\sigma}(k)$$

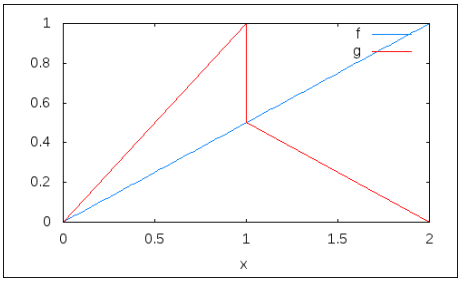
wobei der Ausdruck auf der rechten Seite der Dichte der Normalverteilung entspricht. Verbal ausgedrückt:
Die diskrete Binomialverteilung kann für große n durch die stetige Normalverteilung angenähert werden.
"Große n" heißt als Faustregel

$$\sigma^2 > 9 \Leftrightarrow \sigma > 3$$

Bevor wir uns das in wxMaxima veranschaulichen, vielleicht noch eine kurze Einführung in gnuplot (das wxMaxima verwendet):
Es verfügt über einen Linienzeichenprogramm, das sich sehr leicht scripten lässt - die Syntax sieht so aus: `wxplot2d([discrete, Liste[Liste1, Liste2, ....])`
Liste1, Liste2 sind Koordinatenlisten von Punkten abwechselnd x und y. Diese Punkte werden durch Linien verbunden. 'wxplot2d' nimmt als Input wiederum eine Liste von Funktionen (diskret oder kontinuierlich). Am besten wir schauen uns das an einem Beispiel an:

```
(%i15) wxplot2d([0.5*x,[discrete, [[0,0,1,1], [1,0.5,2,0]]]], [x,0,2], [legend,"f","g"]);
```

(%t15)



Es werden 2 Funktionen gezeichnet: $0.5x$ und eine diskrete aus den Punkten $(0,0)$ und $(1,1)$ und aus den Punkten $(1,0.5)$ und $(2,0)$. Alle diese Punkte werden durch Linien verbunden (default - Einstellung; sie lässt sich auch ändern - aber das ist kein Kurs über GnuPlot)
Außerdem wird er x-Bereich der Zeichnung (Intervall $[0,2]$) festgelegt und wie die Legende auszuschauen hat.

Aber jetzt weiter in unserem Beispiel:
Wir nehmen z.B. eine Binomialverteilung mit den Parametern $p = 0,4 \Rightarrow q = 0,6$ und erhöhen n so, dass obige Faustregel erfüllt ist: z.B. $n = 50$

Geben wir das im wxMaxima ein:

```
(%i1)  n:50$p:0.4$q:1-p$
```

```
(%i4)  bin(k):=binomial(n,k)*p^k*q^(n-k);
```

(%o4) $\mathrm{bin}\,(k) := \binom{n}{k}\, p^k\, q^{n-k}$

```
(%i5)  w1:makelist([i,0,i,bin(i),i+1,bin(i)],i,5,35)$
```

zur Erinnerung: `makelist(expr,var,start,end)` erstellt eine Liste, indem die Variable `var` alle Integerwerte von `start` bis `end` durchläuft: Also Linie von (i,0) nach (i,bin(i)) weiter nach (i+1,bin(i)) - das ist eine Senkrechte und eine Waagrechte und so gehts weiter für alle $i \in [5, 35]$. Bei allen übrigen sind die Wahrscheinlichkeiten so klein, dass sie im Diagramm verschwinden!

```
(%i6)  %mu:n*p;
```

(%o6) 20.0

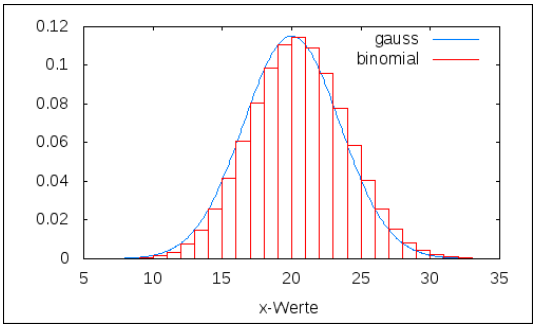
```
(%i7)  %sigma:sqrt(%mu*q);
```

(%o7) 3.464101615137754 Man sieht $\boxed{\sigma > \mathbf{3}}$

```
(%i8)  f(x):=1/(%sigma*sqrt(2*%pi))*%e^(-0.5*((x-%mu)/%sigma)^2),numer;
```

(%o8) $\mathrm{f}\,(x) := \dfrac{1}{\sigma\,\sqrt{2\,\pi}}\, e^{(-0.5)\,\left(\frac{x-\mu}{\sigma}\right)^2}$

```
(%i9)  wxplot2d([f(x), [discrete,w1]],[x,5,35] ,[xlabel,"x-Werte"],[legend,"gauss","binomial"]);
```



(%o9)

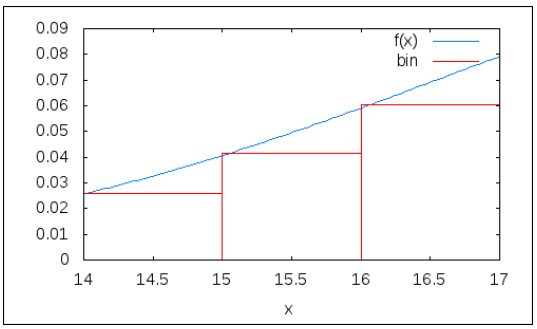
Diesen Befehl haben wir weiter oben schon ausführlich behandelt. Man sieht leicht, wie die Wahrscheinlichkeiten der Binomialverteilung gut gegen die Gaußkurve "konvergieren" - nur hie und da gibt es leichte Abweichungen. Also wäre ein erster Ansatz:

$$b_{n,p}(k) \approx \int_{k}^{k+1} \varphi(x)\, dx$$

aber es geht noch besser, dazu zoomen wir in das Diagramm!

```
(%i10) wxplot2d([f(x), [discrete,w1]],[x,14,17],[legend,"f(x)","bin"] );
```

(%t10)

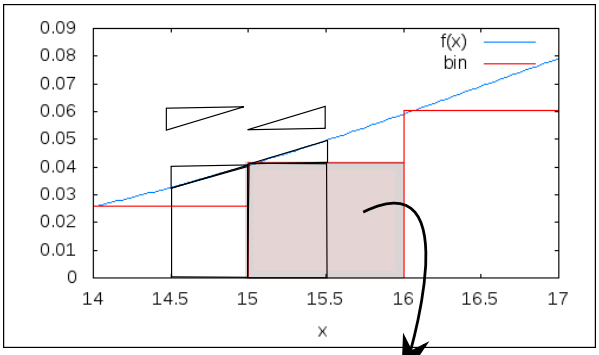


deutlich kann man erkennen:

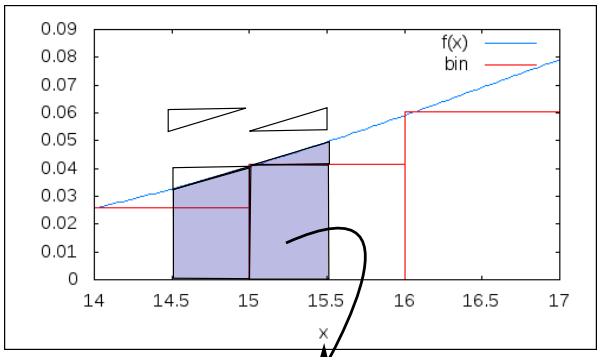
$$P(X=15) = b_{50;0,4}(15) < \int_{15}^{15+1} \varphi(x)\,dx$$

dies gilt natürlich für alle Werte der Zufallsvariablen $X < \mu$ - rechts von μ drehen sich die Größenverhältnisse um.
Würde der Graph von φ eine Gerade sein, könnten wir ein flächengleiches Rechteck "basteln", indem wir die Integrationsgrenzen um $\frac{1}{2}$ verschieben - wie die unteren Zeichnungen deutlich machen:

(%o10)



$b_{n,p}(15)$



$$\int_{15-\frac{1}{2}}^{15+\frac{1}{2}} \varphi_{\mu,\sigma}(x)\,dx$$

Halten wir fest:

$$b_{n;p}(k) \approx \int_{k}^{k+1} \varphi(x)\,dx \qquad \textbf{gut}$$

$$b_{n;p}(k) \approx \int_{k-\frac{1}{2}}^{k+1-\frac{1}{2}} \varphi(x)\,dx = \int_{k-\frac{1}{2}}^{k+\frac{1}{2}} \varphi(x)\,dx \qquad \textbf{ist besser!}$$

Wie man sich leicht überzeugen kann, gilt für Summen:

$$P(a \le X \le b) = \sum_{i=a}^{b} b_{n;p}(i) \approx \int_{a-\frac{1}{2}}^{b+\frac{1}{2}} \varphi(x)\,dx$$

Wir schauen uns die Ergebnisse jetzt numerisch in wxMaxima an:

```
(%i11) sum(bin(i),i,15,17);
```
$\rightarrow \sum_{i=15}^{17} b_{n,p}(i)$

(%o11) 0.182920766322365

`quad_qags` ist einer der numerischen Integrationsalgorithmen von wxMaxima, als output liefert es den Wert des Integrals und den Fehler und einige andere Dinge, die hier nicht wesentlich sind (bedenken Sie, dass die Gaußfunktion über keine Stammfunktion verfügt und daher die "übliche Methode" `integrate(expr, var, start, end)` `nicht` zum Ziel führt!

```
(%i12) quad_qags ( f(x), x , 15, 17+1);
```
$\rightarrow \int_{15}^{17+1} f(x)\,dx$

(%o12) $[.2073940942360036, 2.302536985920908\,10^{-15}, 21, 0]$

```
(%i13) quad_qags ( f(x), x , 15-0.5, 17+0.5);
```

(%o13) $[.1790676121841628, 1.988049860115059\,10^{-15}, 21, 0]$

```
(%i14) sum(bin(i),i,24,26);
```

(%o14) .1247627778007346

```
(%i15) quad_qags ( f(x), x , 24, 26+1);
```

(%o15) $[.1024524680895658, 1.137450890027101\,10^{-15}, 21, 0]$

```
(%i16) quad_qags ( f(x), x , 24-0.5, 26+0.5);
```

(%o16) $[.1258597259821049, 1.39732365658346\,10^{-15}, 21, 0]$

Jetzt das Ganze noch mit dem Maxima-Modul "distrib" (steht für distribution - dt. Verteilung)
`pdf_*` steht für "probability density function" also die Dichtefunktion
`cdf_*` steht für "cumulative density function" als die Verteilungsfunktion; bei der Normalverteilung oft mit $\Phi(z;\mu;\sigma)$ bezeichnet

```
(%i17) load("distrib");
```

(%o17) */usr/share/maxima/*5.29.1*/share/distrib/distrib.mac*

```
(%i18) cdf_normal(18,%mu,%sigma)-cdf_normal(15,%mu,%sigma), numer;
```

(%o18) .2073940942360036

```
(%i19) cdf_binomial(17, n, p)- cdf_binomial(14, n, p);
```

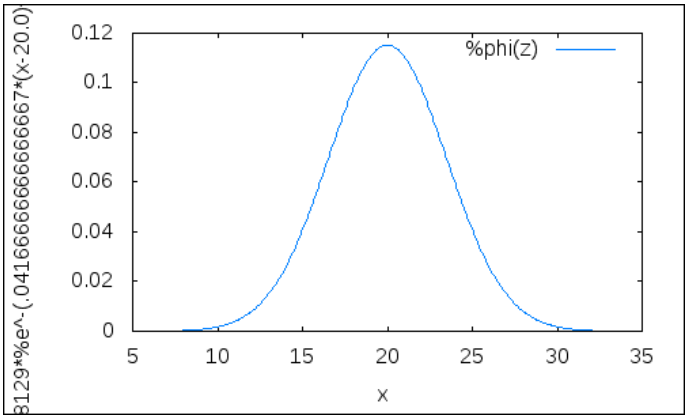
(%o19) .1829207663223664

```
(%i20) sum(pdf_binomial(i,n,p),i,15,17);
```

(%o20) 0.182920766322365

```
(%i24) wxplot2d(pdf_normal(x,%mu,%sigma),[x,5,35],[legend,Phi(z)]);
```

(%t24)



(%o24)

Ein weiterer Grund für die Benutzung des Theorems ist, das $b_{n;p}(k)$ für große n praktisch ohne Spezialsoftware unberechenbar wird:

$$b_{n;p}(k) = \underbrace{\binom{n}{k}}_{\to\infty} \underbrace{p^k(1-p)^{n-k}}_{\to 0} = ?$$

Ein Summenzeichen davor macht die Sache auch nicht besser! Bleiben wir bei unserem obigen Beispiel $p = 0,4$ und erhöhen wir n auf 500. Sei X die Anzahl mit der unser spezifisches Ereignis eintritt und wir möchten wissen, wie groß die Wahrscheinlichkeit ist, dass diese Anzahl um weniger als eine Standardabweichung vom Erwartungswert abweicht:

$$P(|X-\mu| < \sigma) = P(\mu - \sigma < X < \mu + \sigma) = P(190 \le X \le 210) = \sum_{i=190}^{210} \binom{500}{i} 0.4^i\, 0.6^{500-i}$$

Die Summanden der letzten Summe sind für einen "normalen" Taschenrechner praktisch nicht berechenbar, z.B.:

$$\binom{500}{200} \approx 5 \cdot 10^{144} \qquad \text{und} \qquad 0.4^{200} \cdot 0.6^{300} \approx 7 \cdot 10^{-147}$$

$$\Phi_{\mu,\sigma}(210,5) - \Phi_{\mu,\sigma}(189,5) \underbrace{=}_{standardisieren} \Phi(\frac{10,5}{\sigma}) - \Phi(-\frac{10,5}{\sigma}) = 2\Phi(\frac{10,5}{\sigma}) - 1 = 2\Phi(0.9585) - 1$$

Während obige Binomial-Wahrscheinlichkeiten nicht mehr mit jedem x-beliebigen Taschenrechner zu berechnen sind, ist das letzte Ergebnis mit jeder Normalverteilungstabelle leicht zu ermitteln!

Wie sieht das in wxMaxima aus?

```
(%i22) n:500;
```

(%o22) 500

```
(%i23) %mu:n*p;
```

(%o23) 200.0

```
(%i24) %sigma:sqrt(%mu*q);
```

(%o24) 10.95445115010332

```
(%i25) f(x):=1/(%sigma*sqrt(2*%pi))*%e^(-0.5*((x-%mu)/%sigma)^2)$
```

```
(%i26) quad_qags ( f(x), x , 189.5, 210.5);
```

(%o26) $[.6621966043761834, 7.3518591700703471\,10^{-15}, 21, 0]$

Hier das Integral mit Stetigkeitskorrektur!

```
(%i27) 2*cdf_normal(0.9585,0,1)-1, numer;
```

(%o27) .6621893085136601

Normalverteilungstabelle

```
(%i28) cdf_binomial(210, 500, p)- cdf_binomial(189, 500, p);
```

(%o28) .6621956027530275

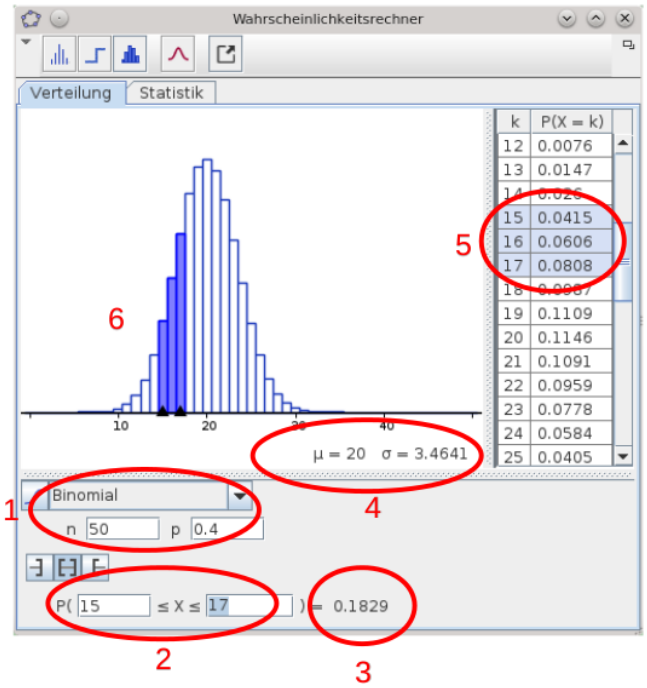
Hier die Summe mit der "eingebauten" Verteilungsfunktion: Beachte die Grenze 189

```
(%i29) display(sum(bin(i),i,190,210))$
```

$$\sum_{i=190}^{210} \mathrm{bin}\,(i) = .6621956027529964$$

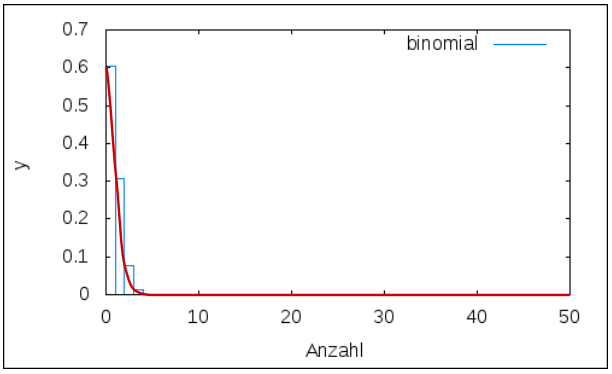
Hier die Summe "zu Fuß"

Fazit: Es ist gleichgültig, welche Methode man wählt, Hauptsache man macht es richtig!



Apropos "gleichgültig, welche Methode man wählt". Sehr einfach gehts jetzt auch mit Geogebra 4.2+. Hier verbirgt sich im Menü "ABC" ein Wahrscheinlichkeitsrechner, der extrem einfach zu bedienen ist. In 1 stellt man die Parameter ein ($b_{n;p}$), in 2 die Intervallgrenzen und schon kann man das Ergebnis in 3 ablesen. Als "Zucker" gibts in 4 μ und σ, in 5 die "Teilwahrscheinlichkeiten" und in 6 eine grafische Aufbereitung. Dieser Rechner ist extrem schnell und macht eigentlich dieses Dokument über Stetigkeitskorrektur beinahe überflüssig!

Nachsatz: Vergessen Sie nicht auf auf die Voraussetzung $\sigma^2 > 9$. Im andern Fall (p weicht weit von 0,5 ab und/oder n ist nicht allzu groß), kann die Binomialverteilung "sehr schief" sein und die Symmetrie bricht zusammen!



Hier nur als Warnung eine Binomialverteilung mit den Parametern $n = 50$, $p = 0.05$. An den Eckpunkten ist eine kontinuierliche rote Kurve gezogen - von einer Glockenkurve ist hier nichts zu sehen. Irgendwie erinnert das an eine fallende Exponentialfunktion - aber die Poissonverteilung ist wieder eine andere Geschichte!