# Common distributions in Julia, Python and R

Please report errors on https://github.com/sylvaticus/commonDistributionsInJuliaPythonR

## Loading packages

- Julia: `using Distributions`
- Python: `from scipy import stats`
- R: `library(extraDistr)`

## Discrete distributions

- **Discrete Uniform** : Complete ignorance
- **Bernoulli** : Single binary trial
- **Binomial** : Number of successes in independent binary trials
- **Categorical** : Individual categorical trial
- **Multinomial** : Number of successes of the various categories in independent multinomial trials
- **Geometric** : Number of independent binary trials until (and including) the first success (discrete time to first success)
- **Hypergeometric** : Number of successes sampling without replacement from a bin with given initial number of items representing successes
- **Multivariate hypergeometric** : Number of elements sampled in the various categories from a bin without replacement
- **Poisson** : Number of independent arrivals in a given period given their average rate per unit time
- **Pascal** : Number of independent binary trials until (and including) the n-th success (discrete time to n-th success).

| Name | Parameters | Support | PMF | Expectations | Variance | CDF |
|---|---|---|---|---|---|---|
| **D. Unif** | a,b ∈ Z with b ≧ a | $x \in \{a, a+1, \dots, b\}$ | $\frac{1}{b-a+1}$ | $\frac{a+b}{2}$ | $\frac{(b-a)(b-a+2)}{12}$ | $\frac{x-a+1}{b-a+1}$ |
| **Bern** | p ∈ [0,1] | x ∈ {0,1} | $p^x(1-p)^{1-x}$ | $p$ | $p(1-p)$ | $\sum_{i=0}^x p^i(1-p)^{1-i}$ |
| **Bin** | p ∈ [0,1], n in $N^+$ | $x \in \{0, \dots, n\}$ | $\binom{n}{x}p^x(1-p)^{1-x}$ | $np$ | $np(1-p)$ | $\sum_{i=0}^x \binom{n}{i}p^i(1-p)^{1-i}$ |
| **Cat** | $p_1, p_2, \dots, p_K$ with $p_k \in [0,1]$ and $\sum_{k=1}^K p_k = 1$ | x ∈ {1,2,...,K} | $\prod_{k=1}^K p_k^{1(k=x)}$ | | | |
| **Multin** | $n, p_1, p_2, \dots, p_K$ with $p_k \in [0,1]$, $\sum_{k=1}^K p_k = 1$ and $n \in N^+$ | $x \in \mathbb{N}_0^K$ | $\binom{n}{x_1, x_2, \dots, x_K}\prod_{k=1}^K p_k^{x_K}$ | | | |
| **Geom** | p ∈ [0,1] | x ∈ $N^+$ | $(1-p)^{x-1}p$ | $\frac{1}{p}$ | $\frac{1-p}{p^2}$ | $1-(1-p)^x$ |
| **Hyperg** | $n_s, n_f, n \in \mathbb{N}_0$ | $x \in \mathbb{N}_0$ with $x \leq n_s$ | $\frac{\binom{n_s}{x}\binom{n_f}{n-x}}{\binom{n_s+n_f}{n}}$ | $n\frac{n_s}{n_s+n_f}$ | $n\frac{n_s}{n_s+n_f}\frac{n_f}{n_s+n_f}\frac{n_s+n_f+n}{n_s+n_f+1}$ | |
| **Multiv hyperg** | $n_1, n_2, \dots, n_K$, $n$ with $n \in \mathbb{N}_+, n_i \in \mathbb{N}_0$ | $x \in \mathbb{N}_0^K$ with $x_i \leq n_i \; \forall i$, $\sum_{i=1}^K x_i = n$ | $\frac{\prod_{i=1}^K \binom{n_i}{x_i}}{\binom{\sum_{i=1}^K n_i}{n}}$ | $n\frac{n_i}{\sum_{i=1}^K n_i}$ | $n\frac{\sum_{j=1}^K n_j - n}{\sum_{j=1}^K n_j - 1}\frac{n_i}{\sum_{j=1}^K n_j}\left(1-\frac{n_i}{\sum_{j=1}^K n_j}\right)$ | |
| **Pois** | λ in $R^+$ | x ∈ $N_0$ | $\frac{\lambda^x e^{-\lambda}}{x!}$ | $\lambda$ | $\lambda$ | |
| **Pasc** | n ∈ $N^+$, p in [0,1] | x in $N^+$ | $\binom{x-1}{n-1}p^n(1-p)^{x-n}$ | $\frac{n}{p}$ | $\frac{n(1-p)}{p^2}$ | |

| Distribution | Julia | Python (stats.[distributionName]) | R |
|---|---|---|---|
| Discrete uniform | `DiscreteUniform(lRange,uRange)` | `randint(lRange,uRange)` | `dunif(lRange,uRange)` |
| Bernoulli | `Bernoulli(p)` | `bernoulli(p)` | `bern(p)` |
| Binomial | `Binomial(n,p)` | `binom(n,p)` | `binom(n,p)` |
| Categorical | `Categorical(ps)` | Not Av. | `cat(ps)` |
| Multinomial | `Multinomial(n, ps)` | `multinomial(n, ps)` | `mnom(n,ps)` |
| Geometric | `Geometric(p)` | `geom(p)` | `geom(p)` |
| Hypergeometric | `Hypergeometric(nS, nF, nTrials)` | `hypergeom(nS+nF,nS,nTrials)` | `hyper(nS, nF, nTrias)` |
| Mv hypergeometric | Not Av. | `multivariate_hypergeom(initialNByCat,nTrials)` | `mvhyper(initialNByCat,nTrials)` |
| Poisson | `Poisson(rate)` | `poisson(rate)` | `pois(rate)` |
| Negative Binomial | `NegativeBinomial(nSucc,p)` | `nbinom(nSucc,p)` | `nbinom(nSucc,p)` |

# Continuous distributions

- **Uniform** complete ignorance, pick at random, all equally likely outcomes
- **Exponential** waiting time to first event whose rate is λ (continuous time to first success)
- **Normal** The asymptotic distribution of a sample means
- **Erlang** Time of the n-th arrival
- **Cauchy** The ratio of two independent zero-means normal r.v.
- **Chi-squared** The sum of the squared of iid standard normal r.v.
- **T distribution** The distribution of a sample means
- **F distribution** : The ratio of the ratio of two indep $X^2$ r.v. with their relative parameter
- **Beta distribution** The Beta distribution
- **Gamma distribution** Generalisation of the exponential, Erlang and chi-square distributions

| Name | Parameters | Support | PMF | Expectations | Variance | CDF |
|---|---|---|---|---|---|---|
| **Unif** | a,b ∈ R with b ≧ a | x \in [a,b] | $\frac{1}{b-a}$ | $\frac{a+b}{2}$ | $\frac{(b-a)^2}{12}$ | $\frac{x-a}{b-a}$ |
| **Expo** | λ ∈ $R^+$ | x ∈ $R^+$ | $\lambda e^{-\lambda x}$ | $\frac{1}{\lambda}$ | $\frac{1}{\lambda^2}$ | $1-e^{-\lambda x}$ |
| **Normal** | μ ∈R, $\sigma^2$ ∈ $R^+$ | x ∈ R | $\frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-(x-\mu)^2}{2\sigma^2}}$ | $\mu$ | $\sigma^2$ | |
| **Erlang** | n ∈ $N^+$, λ ∈ $R^+$ | x ∈ $R_0$ | $\frac{\lambda^n x^{n-1}e^{-\lambda x}}{(n-1)!}$ | $\frac{n}{\lambda}$ | $\frac{n}{\lambda^2}$ | |
| **Cauchy** | $x_0$ ∈ R (*location*), γ ∈ $R^+$ (*scale*) | $\frac{1}{\pi\gamma(1+(\frac{x-x_0}{\gamma})^2)}$ | | | | |
| **Chi-sq** | d ∈ $N^+$ | x ∈ $R^+$ | $\frac{1}{2^{\frac{d}{2}}\Gamma(\frac{d}{2})}x^{\frac{d}{2})-1}e^{-\frac{x}{2}}$ | $d$ | $2d$ | |
| **T** | ν ∈ $R^+$ | x ∈ R | $\frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}\left(1+\frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}$ | | | |
| **F** | $d_1$ ∈ $N^+$ $d_2$ ∈ $N^+$ | x ∈ $R^+$ | $\frac{\sqrt{\frac{(d_1x)^{d_1}d_2^{d_2}}{(d_1x+d_2)^{d_1+d_2}}}}{x\mathrm{B}\left(\frac{d_1}{2},\frac{d_2}{2}\right)}$ | $\frac{d_2}{d_2-2}$ for $d_2>2$ | $\frac{2d_2^2(d_1+d_2-2)}{d_1(d_2-2)^2(d_2-4)}$ for $d_2>4$ | |
| **Beta** | α, β ∈ $R^+$ | x ∈ [1,0] | $\frac{1}{B(\alpha,\beta)}x^{\alpha-1}(1-x)^{\beta-1}$ | $\frac{\alpha}{\alpha+\beta}$ | $\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$ | |
| **Gamma** | α ∈ $R^+$ (*shape*), β ∈ $R^+$ (*rate*) | x ∈ $R^+$ | $\frac{\beta^\alpha}{\Gamma(\alpha)}x^{\alpha-1}e^{-\beta x}$ | $\frac{\alpha}{\beta}$ | $\frac{\alpha}{\beta^2}$ | |

**Beta function** : $B(\alpha,\beta)=\frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}=\frac{\alpha+\beta}{\alpha\beta}$
**Gamma function**: $\Gamma(x)=(x-1)!\ \forall x\in N$

| Distribution | Julia | Python (stats.[distributionName]) | R |
|---|---|---|---|
| Uniform | `Uniform(lRange,uRange)` | `uniform(lRange,uRange)` | `unif(lRange,uRange)` |
| Exponential | `Exponential(rate)` | `expon(rate)` | `exp(rate)` |
| Normal | `Normal(μ,sqrt(σsq))` | `norm(μ,math.sqrt(σsq))` | `norm(μ,sqrt(σsq))` |
| Erlang | `Erlang(n,rate)` | `erlang(n,rate)` | Use gamma |
| Cauchy | `Cauchy(μ, σ)` | `cauchy(μ, σ)` | `cauchy(μ,σ)` |
| Chisq | `Chisq(df)` | `chi2(df)` | `chisq(df)` |
| T Dist | `TDist(df)` | `t(df)` | `t(df)` |
| F Dist | `FDist(df1, df2)` | `f(df1, df2)` | `f(df1,df2)` |
| Beta Dist | `Beta(shapeα,shapeβ)` | `beta(shapeα,shapeβ)` | `beta(shapeα,shapeβ)` |
| Gamma Dist | `Gamma(shapeα,1/rateβ)` | `gamma(shapeα,1/rateβ)` | `gamma(shapeα,1/rateβ)` |

Note: The Negative Binomial returns the number of failures before n successes instead of the total trials to n successes as the Pascal distribution

# Usage

$y=CDF(x)$, i.e. y ∈ [0,1]

| | Julia | Python | R |
|---|---|---|---|
| **Mean** | `mean(d)` | `d.mean()` | |
| **Variance** | `var(d)` | `d.var()` | |
| **Median** | `median(d)` | `d.median()` | |
| **Sample** | `rand(d)` | `d.rvs()` | `r[distributionName](1,distributionParameters)` , e.g. `runif(1,10,20)` |
| **Quantiles** $(F^{-1}(y))$ | `quantile(d,y)` | `d.ppf(y)` | `q[distributionName](y, distributionParameters)` , e.g. `qunif(0.2,10,20)` |

| | Julia | Python | R |
|---|---|---|---|
| **PDF/PMF** | `pdf(d,x)` | `d.pmf(x)` for discrete r.v. and `d.pdf(x)` for continuous ones | `d[distributionName](x, distributionParameters)` , e.g. `dunif(15,10,20)` |
| **CDF** | `cdf(d,x)` | `d.cdf(x)` | `p[distributionName](x, distributionParameters)` , e.g. `punif(15,10,20)` |