# *Distributions.jl Cheat Sheet*

Carlos Lesmes

September 26, 2025

## 1 Basics

### Install Julia Language

Go to https://julialang.org/downloads/ to download and install Julia.

### Install packages

In the Julia REPL (Read-Eval-Print Loop) type `]` to enter the package manager mode, then type:

```
julia> ]
Pkg> add Distributions, Statistics, Random, StatsBase
julia> using Distributions, Statistics, StatsBase
```

To go back to the Julia REPL type `backspace`, `using` is a command to load the installed packages.

### Create a distribution

```
p = [.4,.3,.2,.1]  # categorical ditribution
d0 = Categorical(p)
d1 = Binomial(10, 0.5)   # Binomial distribution with n=10 and p=0.5
d2 = Normal(10,4)    #Normal distribution with mean 10 and std 4
d3 = FDist(4,6)  # Fdistribution with (4,6) df
d4 = Exponential(2.5) # Exponential distribution scale 2.5
d5 = Gamma(2,3) # Gamma distribution with α=2, θ=3
d6 = Chisq(3) # Chi square distribution with 3 df
```

### Parameter retrieval

If method applies to the distribution:

```
params(d1) # Return a tuple of parameters.
scale(d2) # Get the scale parameter.
location(d2)  # Get the location parameter
shape(d5) # Get the shape parameter
scale(d4) # Get the scale parameter
rate(d4) # Get the rate parameter
ncategories(d0)  # # Get the number of categories
probs(d0)  # Get the probability vector
ntrials(d1) # Get the number of trials.
succprob(d1) # Get the probability of success
failprob(d1) # Get the probability of failure.
```

## 2 Statistics

### Computation of Statistics

```
minimum(d1)  # Return the maximum of the support of d.
maximum(d1)  # Return the minimum of the support of d.
extrema(d2)  # Return the minimum and maximum of the support of d as a
2-tuple.
mean(d2) # Compute the expectation.
var(d2) # Compute the variance.
std(d2) # Return the standard deviation of distribution d, i.e.
sqrt(var(d)).
median(d2) # Return the median value of distribution d. The median is
the smallest x in the support of d for which cdf(d, x) ≥ 1/2.
Corresponding to this definition as 1/2-quantile, a fallback is provided
calling the quantile function.
mode(d2)  # Returns the first mode.
skewness(d2) # Compute the skewness.
kurtosis(d2)  # Computes excess kurtosis by default. Proper kurtosis can
be returned with correction=false
isplatykurtic(d2)  # Return whether d is platykurtic (i.e kurtosis(d) <
0).
isleptokurtic(d2)  # Return whether d is leptokurtic (i.e kurtosis(d) >
0).
ismesokurtic(d2)  # Return whether d is mesokurtic (i.e kurtosis(d) ==
0).
entropy(d2)  # Compute the entropy value of distribution d.
mgf(d2,.5)  # Evaluate the moment-generating function of distribution d
at t.
cgf(d2,1)  # Evaluate the cumulant-generating function of distribution d
at t.
cf(d2,1)  # Evaluate the characteristic function of distribution d.
pdfsquaredL2norm(d2)  # Return the square of the L2 norm of the
probability density function f(x)
```

## 3 Probability

### Probability evaluation

```
insupport(d1,2) # When x is a scalar, it returns whether x is within the
support of d. When x is an array, it returns whether every element in x
is within the support of d.
pdf(d2,1) # Evaluate the probability density (mass) at x.
logpdf(d2,-1) # Evaluate the logarithm of probability density (mass) at
x.
cdf(d2,1) # Evaluate the cumulative probability at x.
logcdf(d2,0) # The logarithm of the cumulative function value(s)
evaluated at x.
logdiffcdf(d2,0,1) # The natural logarithm of the difference between the
cumulative density function at x and y
ccdf(d2,-1) # The complementary cumulative function evaluated at x, i.e.
1 - cdf(d, x)
logccdf(d2,0) # The logarithm of the complementary cumulative function
values evaluated at x
quantile(d2, 0.8) # Evaluate the (generalized) inverse cumulative
distribution function at q.
cquantile(d2,0.3) # The complementary quantile value, i.e. quantile(d,
1-q)
invlogcdf(d2,-0.2) # The (generalized) inverse function of logcdf
invlogccdf(d2,-.5) # The (generalized) inverse function of logccdf
```

## 4 Sampling

### Random number generation

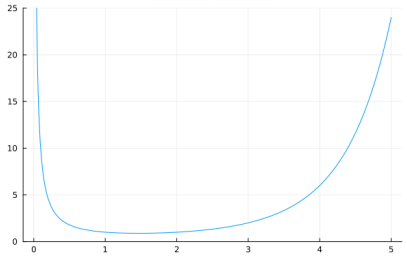
```
Random.seed!(1234) # set random seed for reproducibility
rand(d2,n) # Generate a n-vector sample from d
```

## 5 Special functions

### Gamma function

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t}\, \mathrm{d}t$$

```
using SpecialFunctions
gamma(5) # Calculates gamma function at 5
gamma(0.5)
```




### Beta function

$$B(x,y) = \frac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)}$$

```
beta(2,3) # B(2,3)
```

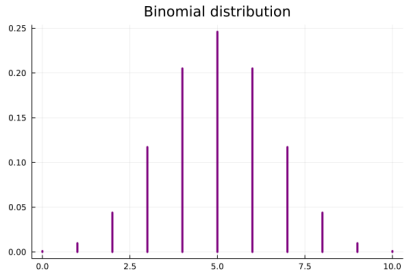
## 6 Plots

### Install packages

Install `StatsPlots` package. This package is a drop-in replacement for Plots.jl that contains many statistical recipes for concepts and types introduced in the JuliaStats organization.

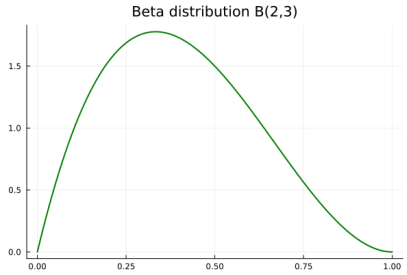
```
Pkg> add StatsPlots
julia> using StatsPlots
julia> plot(Binomial(10,0.5), leg= false)
julia> title!("Binomial distribution")
julia> savefig("binomial.png")
```

### Discrete distribution




### Continuous distribution

```
julia> plot(Beta(2,3), leg= false)
julia> title!("Beta distribution B(2,3)")
julia> savefig("beta.png")
```




## 7 Distributions

### Discrete

Bernoulli
BernoulliLogit
BetaBinomial
Binomial
Biweight
Categorical
Dirac
DiscreteNonParametric
DiscreteUniform
Geometric
Hypergeometric
NegativeBinomial
Poisson
PoissonBinomial
Skellam
Soliton

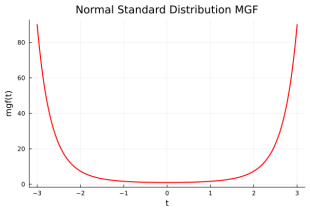
### Continuous

Arcsine
Beta
BetaPrime
Cauchy
Chernoff
Chi
Chisq
Cosine
Epanechnikov
Erlang
Exponential
FDist
Frechet
Gamma
GeneralizedExtremeValue
GeneralizedPareto
Gumbel
InverseGamma
InverseGaussian
JohnsonSU
KSDist
KSOneSided
Kolmogorov
Kumaraswamy
Laplace
Levy
Lindley
LogNormal
LogUniform
Logistic
LogitNormal
NoncentralBeta
NoncentralChisq
NoncentralF
NoncentralT
Normal
NormalCanon
NormalInverseGaussian
PGeneralizedGaussian
Pareto
Rayleigh
Rician
Semicircle
SkewNormal
SkewedExponentialPower
StudentizedRange
SymTriangularDist
TDist
TriangularDist
Triweight
Uniform
VonMises
Weibull

## 8 Another plot

### Normal Moment generation function

```
  using Plots
f(x)=mgf(Normal(),x)
plot(f,-3,3, leg=false)
title!("Normal Standard Distribution MGF")
xlabel!("t")
ylabel!("mgf(t)")
savefig("normal-mgf.png")
```




## 9 References

1. Distributions.jl manual: https://juliastats.org/Distributions.jl/stable/
2. HTML Cheat Sheet: https://carloslesmes.github.io