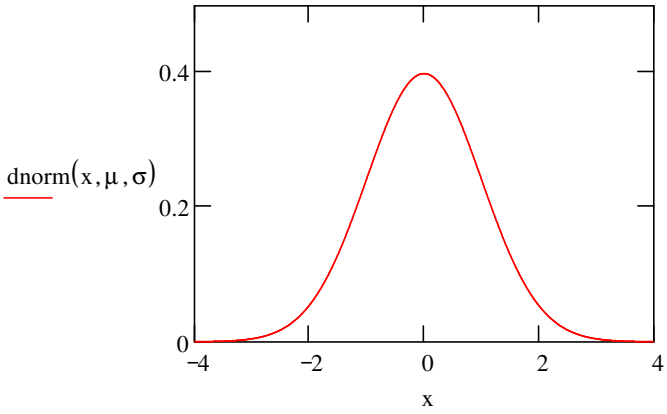
PH24010 Data Handling and Statistics

The Gaussian or Normal distribution

MathCAD provides a built-in function, dnorm(x,μ,σ) which returns the normal or Gaussian distribution, referred to as G(x,μ,σ) in your other notes.
Using the quickplot facility of MathCAD, create a graph of dnorm. Use variables for μ and σ so that you can easily see the effect of changing the mean and standard deviation of the distribution.

$\mu := 0$ Mean of distribution
$\sigma := 1$ Standard deviation



You will need to set the limits on the axes of the graph in order to get the plot like this.

Can you create your own function G(x,μ,σ) using the definition given in the statistics worksheet ?
Compare your function with dnorm(x,μ,σ) on a graph.

You can use MathCAD's numerical integration to verify tht the total area under the probability curve is in fact 1

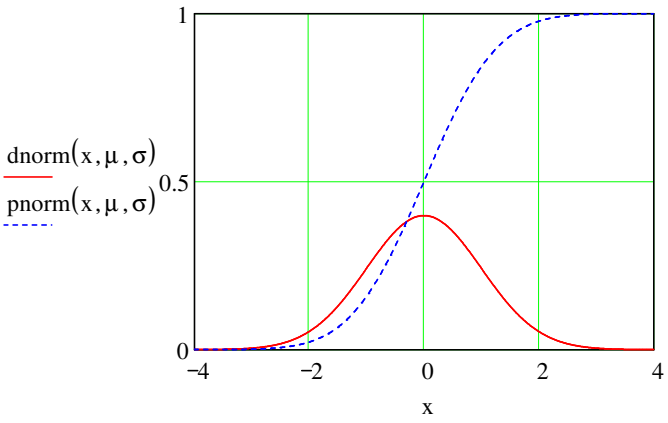
$$\int_{-\infty}^{\infty} \text{dnorm}(x,\mu,\sigma)\,dx = 1$$

The definite integral operator and infinity symbols are to be found on the calculus toolbar

You can also verify that half of the distribution is below the mean

$$\int_{-\infty}^{0} \text{dnorm}(x,\mu,\sigma)\,dx = 0.5$$

MathCAD also provides a function pnorm(x,μ,σ) which models the cumulative distribution, here I have plotted the 2 functions on the same graph.



You can use the cumulative probability function (which is the same as the integral of the dnorm() function) to determine probabilities of finding measurements a within a certain distance of the mean.

For example,

pnorm(-1,μ,σ) tells us the probability of finding a measurement smaller that 1 standard deviation below the mean.

$$\text{pnorm}(-1,\mu,\sigma) = 0.159$$

since the distribution is symmetrical, the probability of finding a reading greater than 1 standard deviation away from the mean is given by

$$2\cdot\text{pnorm}(-1,\mu,\sigma) = 0.317$$

The 2 in the above formula arises because the pnorm() function gives us the cumulative distribution from -infinity to the point in question on one side of the mean and we are looking for this and the corresponding distribution on the high side of the mean. Since the distribution is symmetrical, we can simply double the low side probability.

If this is unclear, try shading in portions of the normal proability distribution on the graph.

Likewise the probability of finding a reading more than 2 or 3 standard deviations from the mean is given by

$$2\cdot\text{pnorm}(-2,\mu,\sigma) = 0.046$$

$$2\cdot\text{pnorm}(-3,\mu,\sigma) = 2.7\times 10^{-3}$$

Compare these numbers with those from the statistics worksheet.

Can you use the pnorm function to give answers to the example sheet questions **Ex1.5a** and **Ex1.5b**

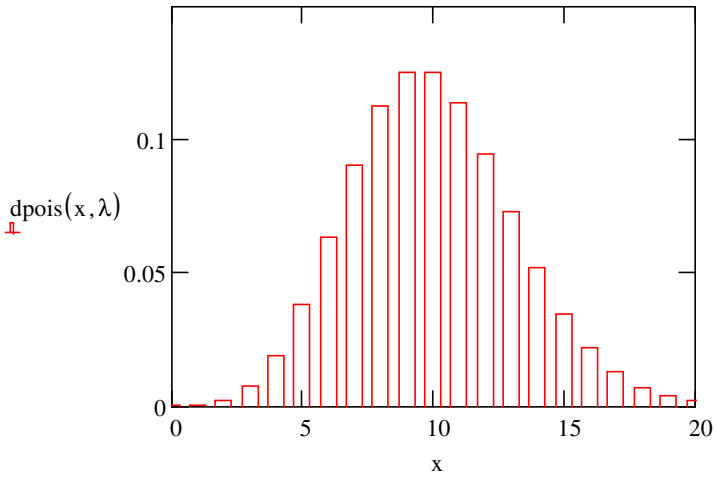
When using the cumulative probability function to answer this and similar questions, you need to consider whether you are looking at 1 tail or both and also whether the answer you are looking for is inside or outside of the distribution.

The Poisson Distribution

In the same way that mathCAD has functions for generating and analysing the normal or Gaussian distribution, so it has corrsponding functions which let us explore the Poisson distribution.

$\lambda := 10$ $\quad\quad$ $\text{limit} := \text{trunc}(2\cdot\lambda)$

$x := 0 .. \text{limit}$



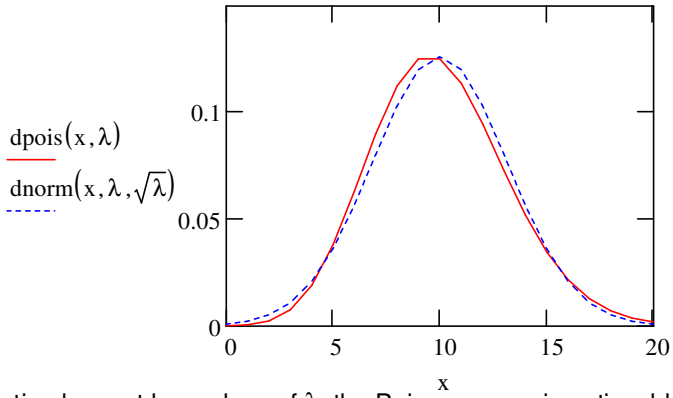
dpois(r,λ) gives the probability that r events will occur in a Poisson distribution with a mean number of event given by λ.

We can check that the total probability of all events is indeed unity.

$\sum_{i=0}^{100} \text{dpois}(i,\lambda) = 1.000000$

Although mathCAD can handle integrals with limits at +/- infinity, we need finite limits for a summation
Set the number of display digits for the result to 6

We can see how the Poisson distribution differs from the Gaussian by plotting the two on the sam graph. This is complicated slightly as the Poisson curve is only valid for integer values on the x-axis



Notice how, at low values of λ, the Poisson curve is noticeably skewed but that at high values of λ, it becomes more symmetrical and approximates to the Gaussian distribution. Alter the value of λ at the top of the page and see for yourself.

MathCAD provides the ppois() function which gives the cumulative probability function for a Poisso distribution. We can use this to solve the **Ex2.2**

Meteors fall at a rate of 36 per hour, or 3 in a 5 minute interval. What is the probability that less than 2 fall in any given 5 minute interval.

$\lambda := 3$

There are 3 equivalent expressions for the probability that less than 2 meteor fall in any given 5 minute period.

The first is equal to the probability that 0 meteors fall, added to the probability that 1 metor falls:

$$\mathrm{dpois}(0,\lambda) + \mathrm{dpois}(1,\lambda) = 0.199148$$

which may be expressed using the summation operator

$$\sum_{i=0}^{1} \mathrm{dpois}(i,\lambda) = 0.199148$$

Alternatively, we may use the ppois() function to gt the answer directly...

$$\mathrm{ppois}(1,\lambda) = 0.199148$$

Problem to solve

A Gieger-Muller tube records disintegrations coming from a sample of radio-isotope.

The average count rate recorded by the apparatus is 10 counts per second.

Calculate the probability the 5 or fewer counts occur in a given 1 second interval.

Binomial distribution

In the same way that MathCAD provides functions to work with Gaussian and Poisson distributions so the binomial distribution is provided for.

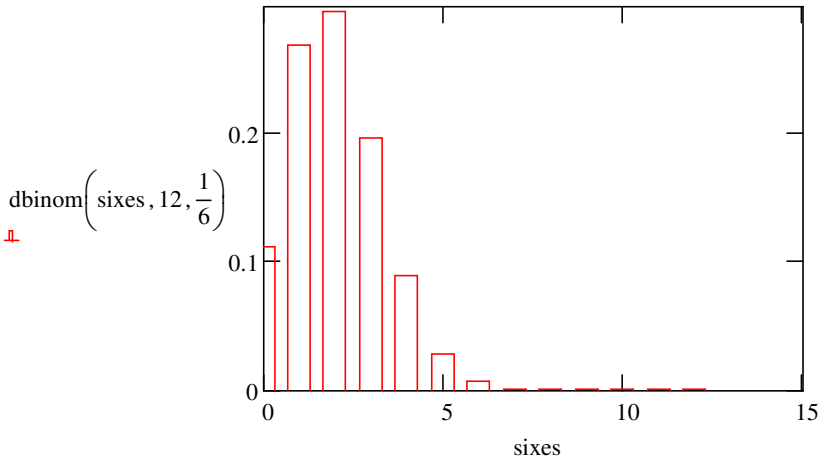
The function dbinom(k,n,p) calculates the probability of k successes in n trials, where the probability of a single success is p

Hence the probability of throwing a dice twelve times without single six is given by:

$$\mathrm{dbinom}\left(0,12,\frac{1}{6}\right) = 0.112$$

The graph of probabilities of throwing r sixes in twelve throw looks like:

$sixes := 0..12$



The corresponding cumulative probabiliy function can be used to calculate the probability that less than (or greater than) a given number of events occur

For example, the probability that 3 or less sixes are thrown in 12 throws of the dice is given by

$$pbinom\left(3, 12, \frac{1}{6}\right) = 0.875$$

Write down the expression to solve **Ex2.3a** using the binomial distribution functions

ie. If a missile defence system stops 99% of incoming, what is the probability of stopping 100 missiles ?

Question **2.3b** asks how many missiles must be launched in order to have a 50% chance of at least one getting through.

We can turn this around and look at the situation from the missile launchers point of view.
The probability of success with any missile is 1%
So the probability of all 'nn' missiles failing is given by

$nn := 30$

$1 - dbinom(0, nn, 1\%) = 0.26$

we can iterate around, changing the value of nn until we have a 50% chance of success.

Unfortunatley, there is no easy way in mathCAD to solve part b of the problem, without writing out the expansions for the binomial distribution as in the worked solution.

Random Numbers, Statistics and Histograms

In this exercise, we are going to look at statistical distributions of random numbers.

We will illustrate aspects of randomness using MathCAD's built-in random number generators.

In many of the examples we will want to draw histograms of data, so we will start by exploring the histogram function.

Histogram() Function

MathCAD provides several functions for creating histograms of data and 2 ways of operating these.

The one which we will explore in this exercise is the histogram() function, which we supply with the data we wish to draw a histogram of and a vector containing the intervals we wish to split the data into to draw the histogram.

As a first experiment, create a 5 row, 1 column vector to hold the endpoints of the intervals and fill i with the numbers from 0 to 5 by hand.

We will also create a vector of data points to analyse, use a Data Table for this (Insert|Data|Table) so that it will re-size as we add data to it. Drag the re-size handles to make the data table 1 column by about 8 or 10 rows.

$$\text{Intervals} := \begin{pmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$$

Data :=

	0
0	0
1	2
2	3
3	4
4	5
5	2
6	3
7	0

Having created our vectors of endpoints and data to analyse, we can call the function and look at the output.

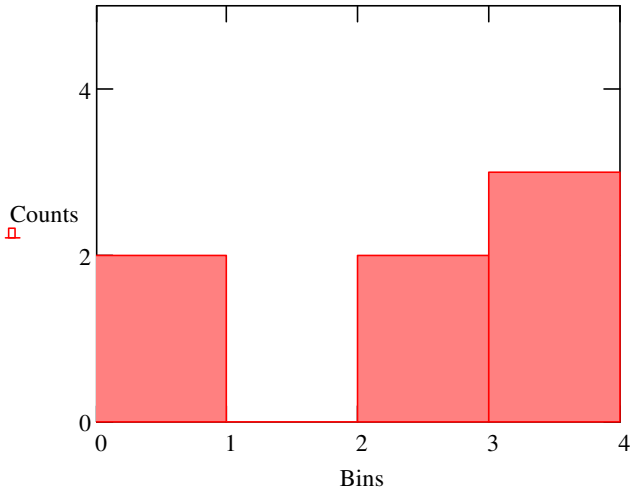
$$\text{histogram}(\text{Intervals}, \text{Data}) = \begin{pmatrix} 0.5 & 2 \\ 1.5 & 0 \\ 2.5 & 2 \\ 3.5 & 3 \end{pmatrix}$$

The first column of the result is composed of the midpoints of our intervals and the second column is the count of data points falling within each interval.

Frequently we can plot the histogram to get a graphical representation of the data. To do this, assign the results of the histogram function to a variable and then use the column extract operator to pull out the 2 columns from that data.

$$HistData := histogram(Intervals, Data)$$

$$Bins := HistData^{\langle 0 \rangle} \qquad Counts := HistData^{\langle 1 \rangle}$$



Use the Graph Format dialog box to change the type of plot for the trace to SolidBar and also set the y-axis limits to between 0 and 5 to get the plot show above.

It wasn't too much effort to put values in by hand for the endpoints in the exercise above. However we frequently need to make endpoint vectors containing many elements and it would be nice to get MathCAD to do some of the work for us. Here we will see how to do this and also introduce some more of mathCAD's programming functions.

We will create a function called IntervalVector(amin,amax,Np), which will produce a vector containing Np intervals between amin and amax.

In order to do this, we will create a program which first calculates the increment between intervals and then uses a loop to fill a vector with the endpoints.

Make sure the programming toolbar is visible on the screen and then enter the following program:

$$IntervalVector(amin, amax, Np) := \left| \begin{array}{l} \Delta a \leftarrow \dfrac{amax - amin}{Np} \\ \text{for } i \in 0..Np \\ \quad Result_i \leftarrow amin + \Delta a \cdot i \\ Result \end{array} \right.$$

You will need to watch the selection rectangle while you are entering the program. Remember that the
h key swaps the insertion point between different ends of the selection bar.

The .. symbol in the for loop is created with the ; (semi-colon) key or from m..n on the matrix toolbar.
Also note that you have to select the program operators from the programming palette rather than typing them. For instance if you type the word 'for' the program will not work.

Once you have entered the program, check that it works

$$\text{IntervalVector}(0,1,5) = \begin{pmatrix} 0 \\ 0.2 \\ 0.4 \\ 0.6 \\ 0.8 \\ 1 \end{pmatrix}$$

Note that we have created a vector with 6 elements, giving 5 intervals.

Having created our program to create vecttors of intervals, we can now test in on some larger data sets.

MathCAD provides a function rrnorm(m,mu,sugma) which returns a vector of m random numbers having
a mean of m and a standard deviation of sigma. This is a good place to start....

$\mu := 0$ $\sigma := 1$ We will assume a mean of 0 and simgma of 1

$Np := 100$ Start with 100 points, we can change this later

$\text{rnormData} := \text{rnorm}(Np, \mu, \sigma)$

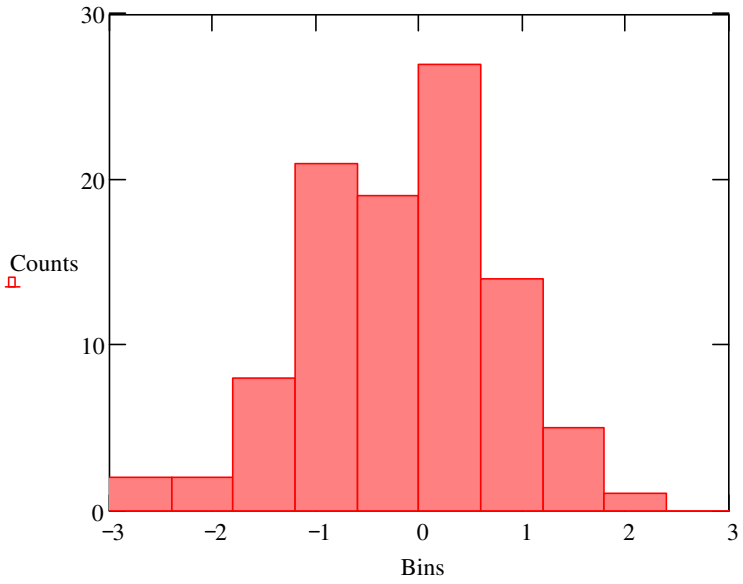
You can look at rnormData at this point if you want, and also do mean and standard deviation calculations to check they give approximately the correct answer.

Having created the data to look at, we will create a histogram with 10 intervals between -3 and +3 standard deviations. In this case we can put the call to IntervalVector() inside the call to the histogram() function.

$xmin := -3 \qquad xmax := 3 \qquad nIntVals := 10$

$rnormHistogram := histogram(IntervalVector(xmin, xmax, nIntVals), rnormData)$

$Bins := rnormHistogram^{\langle 0 \rangle} \qquad Counts := rnormHistogram^{\langle 1 \rangle}$



If you click in the Np or rnormData lines above and press the <f9> key. MathCAD will create a fresh set of data and re-plot it.

Experiment with changing the number of points Np. Try 100, 1000, 10000, 100000

Also experiment with using more or less intervals

MathCAD provides a large number of functions to produce random numbers. Look in the function dialog box under the 'random numbers' section and experiment with a few different types. As well as the built-in functions for generating random numbers, we can also create our own models of random events.

Tossing a coin

We will create a function HeadOrTail() to model the behaviour of a tossed coin. The function will return a 1 to indicate a head and 0 to indicate a tail.

For events with equal probabilities of outcomes, the rnd(z) function is very useful. This provides a single random number, with values evenly distributed between 0 and z. We can then operate on this to get the result we disire.

In this case, we could simply numbers less than 0.5 will return a 1 and those greater will return 0. We can perform the test by simply using one of the conditional operators, remembering that mathCAD uses a 0 to indicate false and 1 to indicate true. If you are unsure about this you can perform a little experimentation to convince yourself.

$0.7 < 0.5 = 0 \qquad 0.3 < 0.5 = 1$

Hence the HeadOrTail() function becomes simply:

$\text{HeadOrTail}(x) := \text{rnd}(1) < 0.5$

$\text{HeadOrTail}(0) = 1$

Again if you put the selection box in the HeadOrTail(0) line and press <f9> you will get a fresh throw of the coin.
Note that the argument of the heads() function, x, is ignored and its presence is only there so that MathCAD understands that you are defining and using a function. It is traditional in these cases to call the function with a 0 as the argument as we have done above.

We can use our new function, HeadOrTail, to investigate binomial distributions. In this case we will explore how many heads we can expect out of a given number of throws.

To do this, first create a function HowManyHeads(Nc) that calls HeadOrTail() Nc times and counts how many times it comes up 1

Like many other problems in computing there are many, many solutions. Here is one

$$\begin{aligned}&\text{HowManyHeads}(Nc) := \\ &\quad \text{count} \leftarrow 0 \\ &\quad \text{for } i \in 0..Nc-1 \\ &\quad\quad \text{count} \leftarrow \text{count} + \text{HeadOrTail}(0) \\ &\quad \text{count}\end{aligned}$$

$\text{HowManyHeads}(6) = 3$

Having create our HowManyHeads() function, we can use it to conduct a number of trail and see how we approach the binomial distribution.

We will call our resultant function CoinTrials(Nt,Nc) where Nt is the number of trials and Nc is the number of coins we throw each trial. the result will be a vector showing the number of heads thrown at each trial.

$$\text{CoinTrials}(Nt, Nc) := \left| \begin{array}{l} \text{for } i \in 0..Nt-1 \\ \quad \text{Result}_i \leftarrow \text{HowManyHeads}(Nc) \\ \text{Result} \end{array} \right.$$

We will see how this works for 1000 trials, each trial throwing 6 coins

$$\text{TrialData} := \text{CoinTrials}(10000, 6)$$

TrialData =

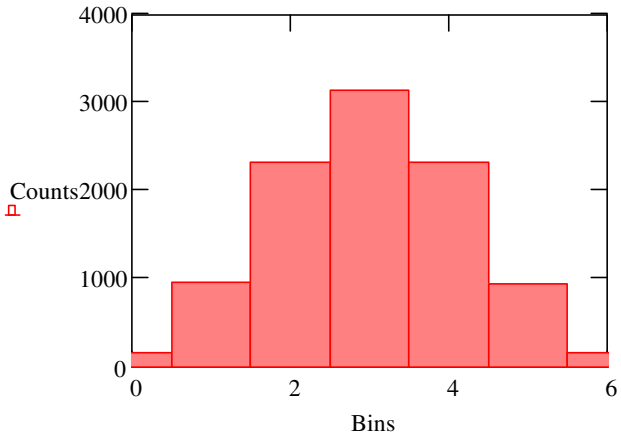
	0
0	3
1	4
2	3
3	4
4	6
5	2
6	5
7	0
8	2
9	1
10	2

Each trial can produce between 0 and 6 heads, so we need to set up our interval vector accordingly

$$\text{TrialBins} := \text{IntervalVector}(-0.5, 6.5, 7)$$

$$\text{TrialHist} := \text{histogram}(\text{TrialBins}, \text{TrialData})$$

$$\text{Counts} := \text{TrialHist}^{\langle 1 \rangle} \quad \text{Bins} := \text{TrialHist}^{\langle 0 \rangle}$$



Exercise for you to try

Look at some different ways of programming the HowManyHeads(Nc) function.

See if you can do it without using any programming operators.

Can you make it a single line function using one of the summation operators and the runif() function

Throwing dice

A dice can produce a random number from 1 to 6, so we can get the random number generator to provide numbers in the range of 0..5 before adding 1 and rounding this to provide numbers in the range 1..6

$$Dice(x) := round(rnd(5) + 1)$$

$$Dice(0) = 6$$

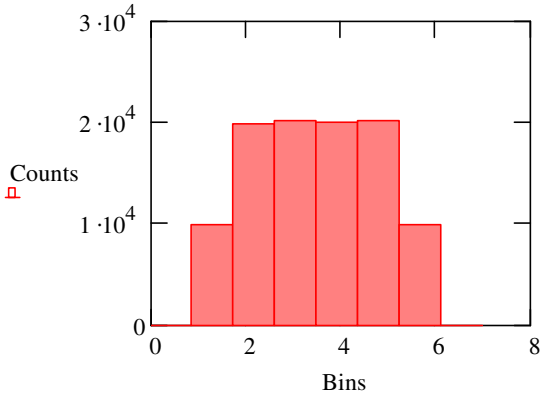
Produce a vector of many die

$$ManyDie(Nd) := \left| \begin{array}{l} \text{for } i \in 0..Nd - 1 \\ \quad Result_i \leftarrow Dice(0) \\ Result \end{array} \right.$$

$$DiceData := ManyDie(100000)$$

$$DiceHist := histogram(IntervalVector(0,7,8), DiceData)$$

$$Bins := DiceHist^{\langle 0 \rangle} \qquad Counts := DiceHist^{\langle 1 \rangle}$$



Looking at the histogram, there is clearly a bias in our dice away from 0 and 6.

Something has gone wrong in the Dice function. Can you modify it to give a balanced die ?

Once you have fixed the Dice function, create a DoubleDie() function and see the most likely score from throwing a pair of dice