# QUEEN MARY, UNIVERSITY OF LONDON

## MTH6102: Bayesian Statistical Methods

### Practical 3

### 2023-2024

### Probability distribution functions

Functions relating to probability distributions have a standard set of names. For example, for the normal distribution:

`dnorm(x)` is the probability density function $\phi(x)$
`pnorm(x)` is the cumulative distribution function $\Phi(x)$
`qnorm(p)` is the quantile function (inverse of cdf) $\Phi^{-1}(p)$
`rnorm` generates random numbers

There is a single help file for these normal distribution commands, which can be opened using

```
?dnorm
```

Unless additional parameters are given the mean is 0 and the standard deviation is 1, i.e. it is the standard normal distribution:

```
dnorm(0)
dnorm(0, mean=3, sd=1)
```

Try each of these commands, using the help file to check the options available. Check some examples where you know what the result should be. For example, $\phi(0)$ should be $1/\sqrt{2\pi}$:

```
dnorm(0)
1/sqrt(2*pi)
```

For the cdf, $\Phi(0)$ should be 0.5; $\Phi(1.96)$ should be approximately 0.975. Check these as well.

We can plot a function to check its shape:

```
curve(dnorm, from=-3, to=3)
```

Try this with `pnorm` and `qnorm` as well, possibly changing the `from` and `to` options.

Similar functions exist for other distributions, see `?distributions`.

`rnorm` generates a vector, so the first argument is the number of elements

```
z1 = rnorm(10)
z1
z2 = rnorm(10, mean=5, sd=2)
z2
```

Check that you understand what the code is doing by displaying variables on the screen, especially intermediate quantities in complicated or multi-line calculations:

```
x = qnorm(0.025)
x
```

For large vectors, datasets and other objects you can use the command `summary(...)` instead.

```
v = rnorm(1000, mean=2, sd=3)
summary(v)
```

### Binomial distribution functions

The help file

```
?dbinom
```

lists the functions for the binomial distribution.

The previous practical askd you to plot the binomial likelihood and log-likelihood. The intention then was to code the formulae from the lectures. As the binomial distribution is discrete, `dbinom` gives the probability mass function, which is also the likelihood when the observed data is $k$, with $k \sim Binom(n, q)$.

Repeat the plots from the end of practical 2, but using `dbinom`. There is more than one way of doing this. Suppose that we have given `n` and `k` values in R. We could again generate a vector for $q$ and a separate vector for the likelihood, and plot these:

```
q = seq(from=0, to=1, length.out=100)
lik = dbinom(x=k, size=n, prob=q)
plot(x=q, y=lik, type="l")
```

Or we could use `curve`, which plots a function without needing to define vectors. For this, we can define a function for this specific example that calculates the likelihood as a function of the input value of `q`:

```
binom_lik = function(q){
    return(dbinom(x=k, size=n, prob=q))
}
curve(binom_lik, from=0, to=1)
```

If you already know how, you can make the plots prettier, for example with meaningful axis titles. We will mention a few such options later in the module.

## Binomial posterior distribution

Assuming that the parameters $\alpha, \beta$ have already been given values in the R code as `alpha` and `beta`, the R code for the Beta$(\alpha, \beta)$ distribution probability density function (pdf) is

```
dbeta(x, shape1=alpha, shape2=beta)
?dbeta
```

We saw that with a Beta$(\alpha, \beta)$ prior distribution for the binomial parameter $q$, the posterior distribution for $q$ is another Beta distribution.

Once you have drawn a graph, you can add another line on the same plot in more than one way. Suppose you generated vectors for $q$ and the likelihood. Also, you need to give `alpha` and `beta` some values, e.g. both equal to 20. Then you can generate another vector for the prior pdf, and add a plot of this pdf against $q$ to an existing graph, using the `lines` command:

```
prior = dbeta(q, shape1=alpha, shape2=beta)
lines(x=q, y=prior, col="red")
```

Alternatively, if you used the `curve` command, you could define a function for the prior pdf, and then use `curve` with the option `add=TRUE`, to add this plot to an existing graph:

```
prior_pdf = function(q){
    return(dbeta(q, shape1=alpha, shape2=beta))
}
curve(prior_pdf, from=0, to=1, add=TRUE, col="red")
```

Do one of the above to plot the likelihood and prior distribution, and then also add a plot for the posterior pdf, all on the same graph.

The likelihood as a function of $q$ is not a pdf, i.e. in general it does not integrate to 1. It may be easier to compare the shape of the likelihood, prior and posterior if the likelihood is rescaled (multiplied by a constant), to integrate to 1.

It can be seen by looking at the binomial probability mass function that the likelihood is proportional to a Beta$(k+1, n-k+1)$ probability density function, so we can plot this Beta distribution as a rescaled version of the likelihood.

Plot the rescaled likelihood this way with plots of the prior and posterior distributions. Vary the prior distribution parameters $\alpha$ and $\beta$ to visualise the effect on the resulting posterior distribution.