Sensitivity Analysis & Sampling

Modern Techniques in Modelling

Introduction

What we will introduce in this session

- Local (or one/multi-way) Analysis
  - Uncertainty analysis
  - Sensitivity analysis

- Global Analysis
  - Probabilistic (or Sampling) uncertainty analysis

Uncertainty and Sensitivity

Uncertainty and *Sensitivity* are often used interchangably (rather confusingly!)

– I will used the terminology used in references below:

- *uncertainty analysis* is the **evaluation of the uncertainty in model output** caused by uncertainty in the model parameter input (most of what we do)
- *sensitivity analysis* is the attribution of this uncertainty to each parameter (either qualitatively or quantitatively)

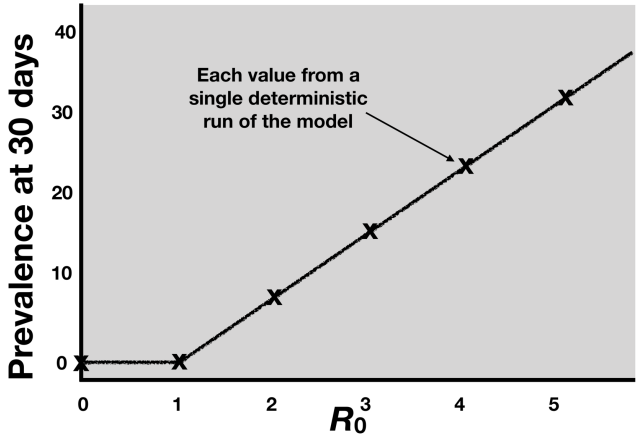
References recommended:

- 'Global Sensitivity Analysis: The Primer' by Saltelli et al. (2008)
- Marino et al. (2008) Journal of Theoretical Biology 254
- Blower & Dowlatabadi (1994) International Statistical Review 62

Local (one/multi-way) uncertainty analysis

We can **change the value of one parameter** (or composite parameter) and evaluate the change in one or more model outputs

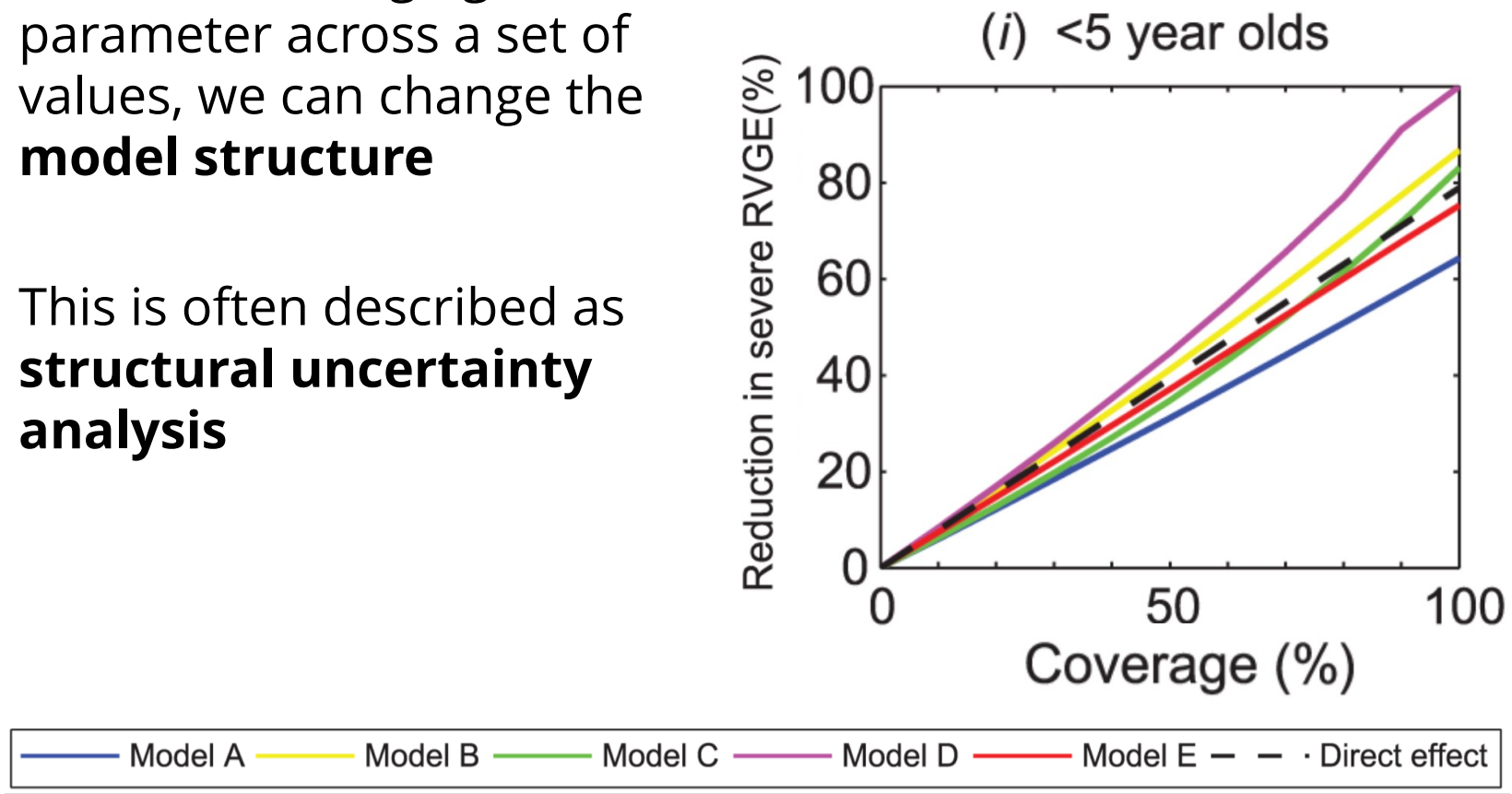
e.g. how does R_0 change the prevalence at day 30



Local (one/multi-way) uncertainty analysis

Instead of changing a parameter across a set of values, we can change the **model structure**

This is often described as **structural uncertainty analysis**



Local (one/multi-way) uncertainty analysis

Evaluating one-way sensitivity analysis is convenient when we want to understand how **changes in our assumptions influence model outcomes**.

One-way uncertainty analysis is also a good approach if one has *discrete* information about the value of a parameter or about model structure

e.g. (1) a study in Thailand suggests the R_0 of Measles is 13, whereas a study in Cambodia suggests is it 18

(2) Does it matter whether our model includes age-dependent susceptibility to infection?

But if there is continuous uncertainty in the value of our parameters (e.g. a clinical trial suggests a range of plausible vaccine efficacy of 76-87%, we might prefer a continuous approach to capture the uncertainty. For this, one can adopt ***global uncertainty analysis***

Global uncertainty analysis: Monte Carlo Sampling

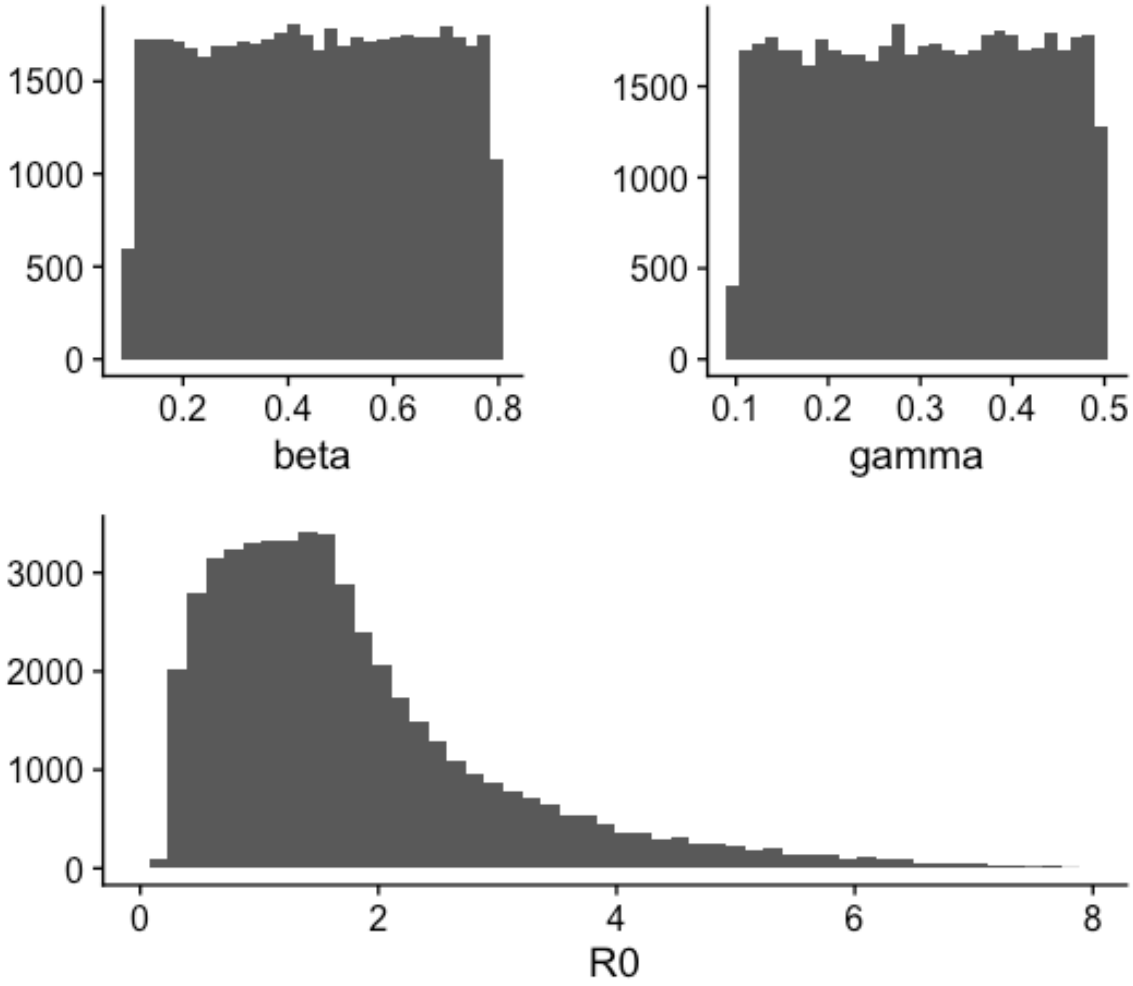
For Global Uncertainty analysis we 'integrate across' all the continuous uncertainty in our parameters to generate a distribution of outcome values

Example: How does uncertainty in the values of β and γ lead to uncertainty in values of R_0?

```
R0 = beta / gamma
```

(Uniform) uncertainty in $\beta \in [0.1,0.8]$ and $\gamma \in [0.1,0.5]$

Global uncertainty analysis: Monte Carlo Sampling

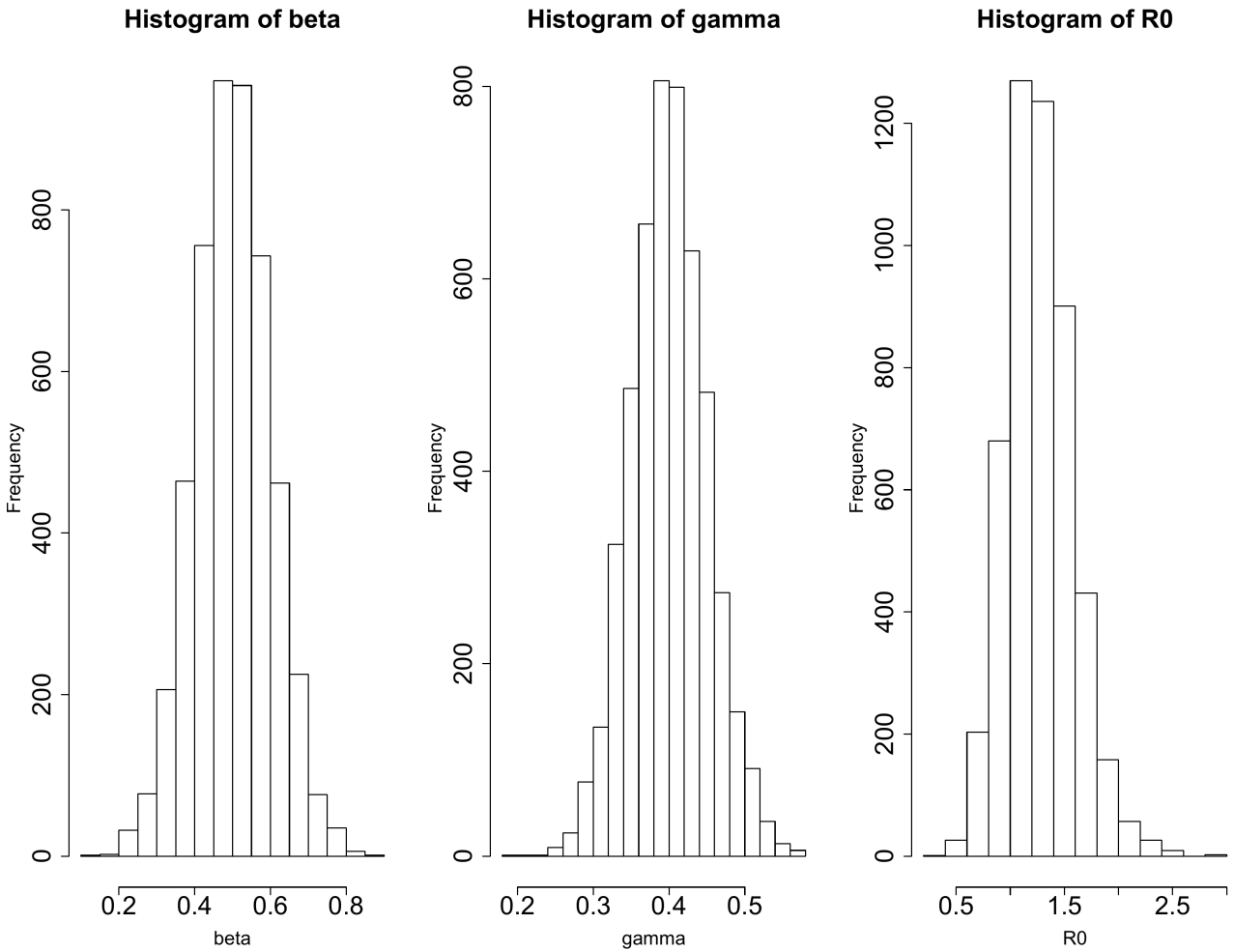


Uniform uncertainty in beta and gamma to R_0= 1.45 (0.41– 4.43)

Global uncertainty analysis: Monte Carlo Sampling

We assumed that both β and γ were distributed uniformly over their respective intervals, but we can relax this rule and choose any distribution we want

Global uncertainty analysis: Monte Carlo Sampling



Global uncertainty analysis: Monte Carlo Sampling

– To sample from any 'closed form' distribution we can use the R functions: `runif(...)`, `rnorm(...)`, `rbeta(...)` etc. (where the '`r`' stands for random number)

```
# Generate 5 random samples from the
# poisson distribution with rate 2.4
pois.samples <- rpois(5,2.4)
[1] 2 5 4 1 5
```

– More broadly, if you replace '`r`' you can get other useful functions

- `dpois(x, ...)` generates the density function at x
- `ppois(q, ...)` generates the cumulative density function at q
- `qpois(p, ...)` generates the inverse cumulative density function at p

Global uncertainty analysis: Monte Carlo Sampling

LONDON SCHOOL of HYGIENE &TROPICAL MEDICINE

– Every time you ask R to 'generate random numbers', you are actually generating *pseudorandom* numbers from random number generators.

– Random number generators (RNGs) are algorithms that produce numbers that look stochastic (random), but are deterministic.

e.g.
```
> runif(1)
[1] 0.9734249
> runif(1)
[1] 0.2389333
```

– If they are deterministic algorithms how do they produce different output every time?

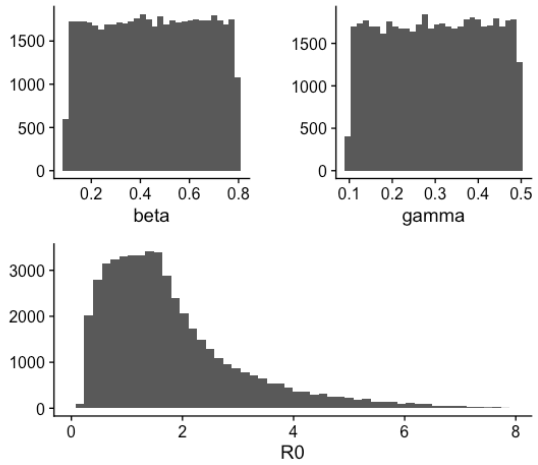
Global uncertainty analysis: Monte Carlo Sampling

- ….by starting in a different place each time.
- To generate a sequence of pseudorandom numbers that are always different, the place (or, 'seed'), where the algorithm starts must always be different.
- This is often achieved by using the current time as the seed.
- R will do this automatically for you, but you can set it manually:
  - E.g. `> set.seed(Sys.time()) # use computer's time to set the seed of the RNG`
- If you want to create the same sequence of numbers, you can use a fixed seed (if you are wanting to debug your code!)
  - E.g.

```
> my.seed <- 42
> set.seed(my.seed)
> runif(1)
> [1] 0.914806
> set.seed(my.seed)
> [1] 0.914806
```

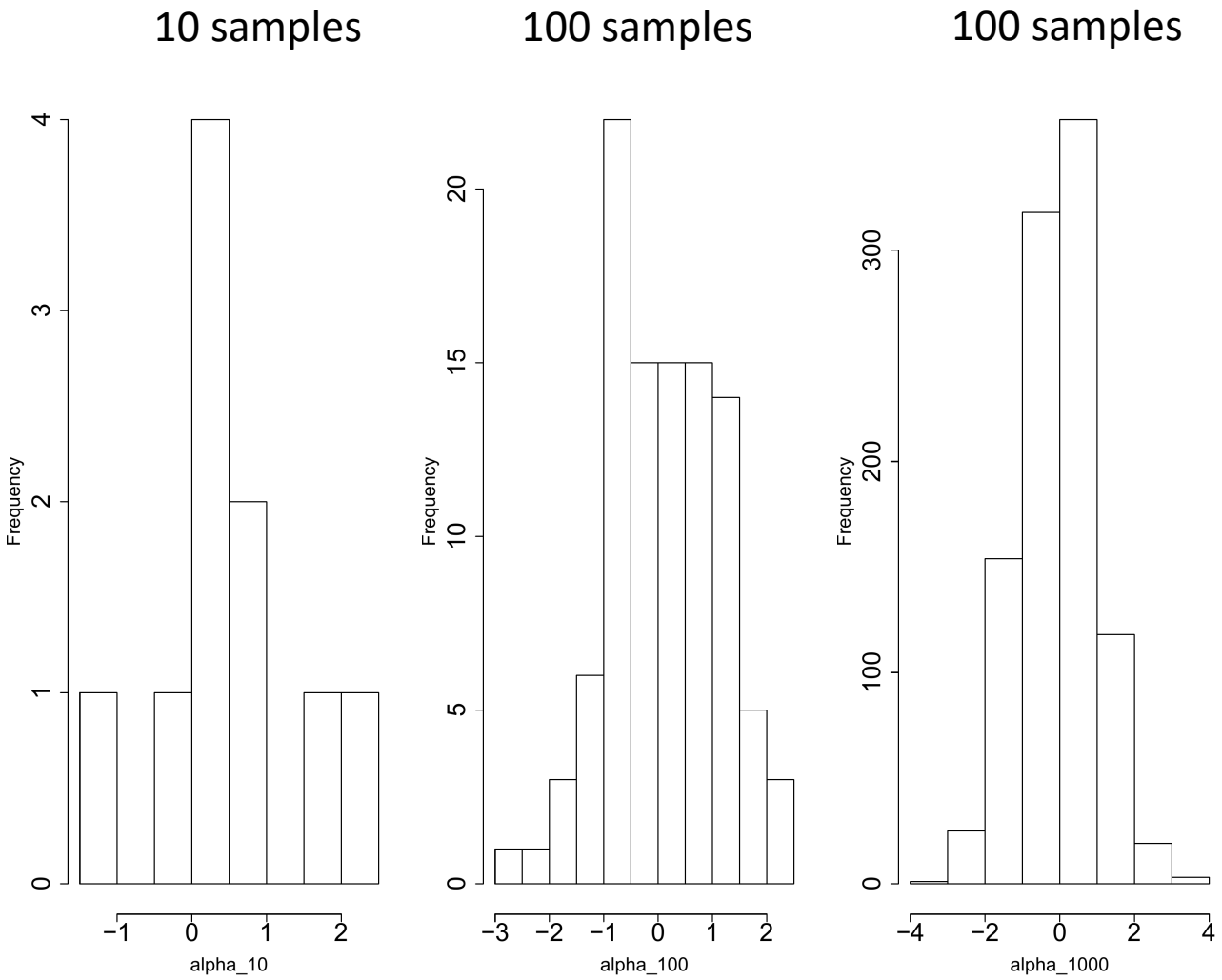
Global uncertainty analysis: Monte Carlo Sampling

Monte Carlo Sampling generates random numbers from an entire distribution. It works because we sample from a parameter distribution (parameter input) that we know to generate samples from the outcome distribution that we don't know.



However, it can be computationally expensive because the parameter samples are picked from across the entire distribution. If we don't pick enough samples, the parameter values we assemble might not be a good representation of the distribution

Global uncertainty analysis: Monte Carlo Sampling



Global uncertainty analysis: Latin Hypercube Sampling

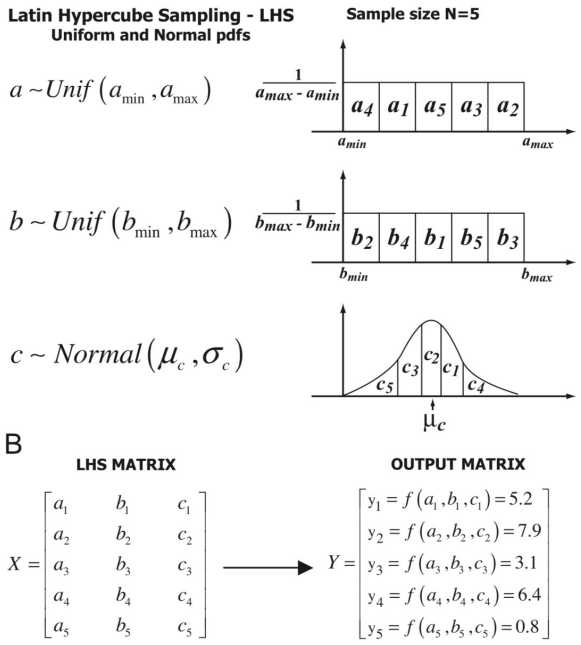
LHS is a specific type of Monte Carlo sampling

The idea is to split up the distribution into as many chunks (with equal probability density) as you want samples and randomly pick a value from each chunk

– This should give you a better representation of the entire distribution and therefore it is more efficient

– Good to use if you are sampling over many parameters at the same time

– A minimum number of samples of $4K/3$, where K is the number of parameters



Marino et al JTB 254(2008)178-196

Global uncertainty analysis: Latin Hypercube Sampling

The package '`lhs`' has pre-written functions:

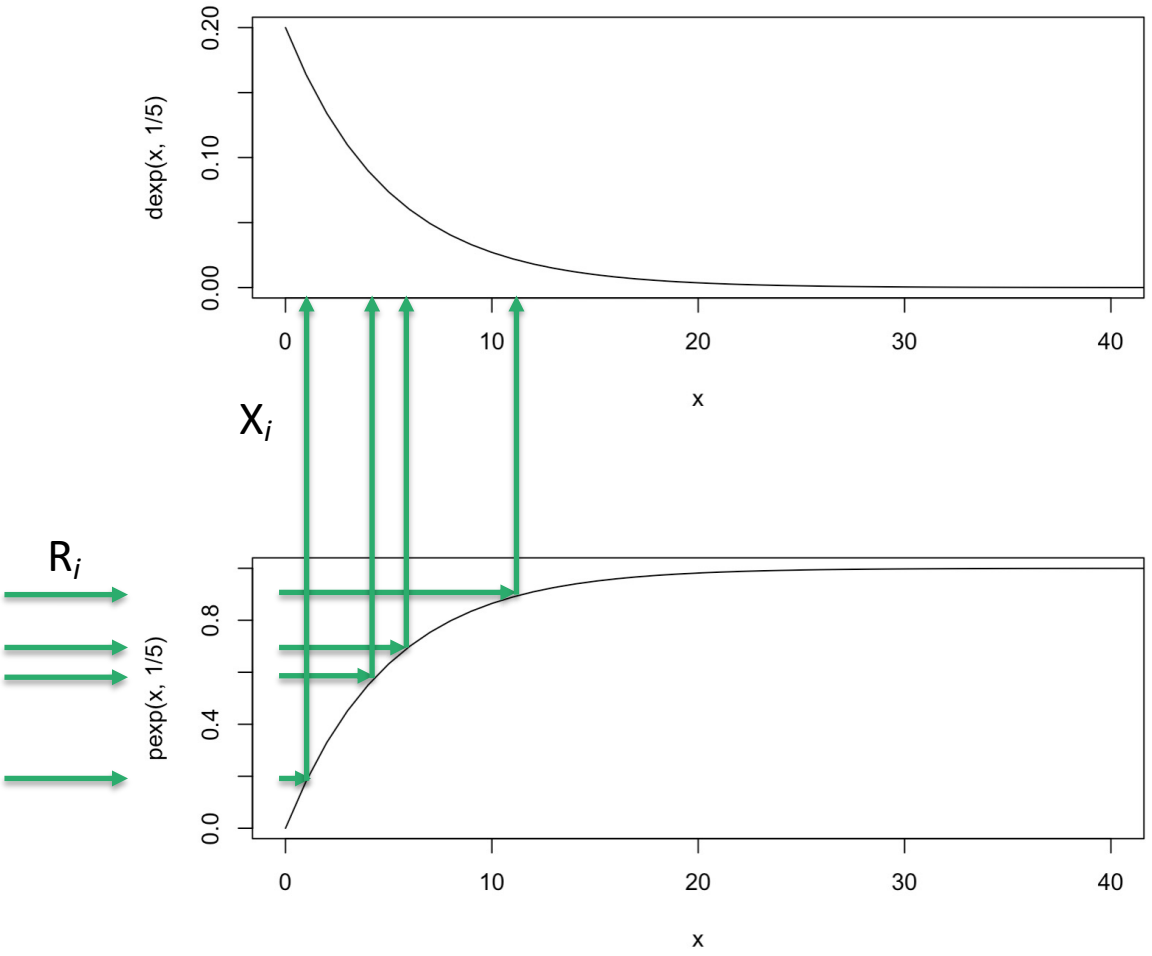
```
# LHS for a uniform distribution [0,1] for each
parameter
myLHSsamples <- lhs::randomLHS(number.of.samples,
                               number.of.parameters)
```

If you need to sample from other distributions, with a closed form $u \sim \mathcal{U}(0,1)$ can be transformed to other distributions using *sampling by inversion*.

Global uncertainty analysis: Sampling by inversion

Example:
Generate samples
$X_1, \ldots, X_n$
from an Exponential Distribution with rate parameter = 1/5

1. Sample from U(0,1):
   $R_1, \ldots, R_n$
2. Calculate a sample from Exp(1/5) using its inverse function
   $F^{-1}(y) = -\log(1-y)$
3. $X_i = -\log(1-R_i)$



Local (one/multi-way) sensitivity analysis: measures

- The **SENSITIVITY** (s) of a model with respect to a parameter is:
  - The increase or decrease in **outcome** unit per increase in **parameter** unit
  - i.e. parameter increase of 0.01 leads to an additive change in the outcome by 0.01s
- The **ELASTICITY** (e) of a model with respect to a parameter is:
  - The proportional change in the outcome when a parameter is increased
  - i.e. increasing a parameter by a factor of 1.01 would lead to the scaling of an outcome measure 1.01e

Both measures indicate how much each parameter influences the outcome measure.

For more information on Global sensitivity analyses, please consult Saltelli 2008.

Monte carlo sampling vs stochasticity

If you use Monte Carlo sampling, you will generate a distribution of outcomes (e.g. epidemic curves), because you are capturing parameter uncertainty.

However, this sampling does **not necessarily mean the underlying model is stochastic** (i.e .that it incorporates randomness).

That is, for a fixed parameter draw in your Monte Carlo samples, each model run can be **either deterministic** (the types of models we have seen so far in the course) **or stochastic** (there is randomness – we'll see examples of these tomorrow).

Over to you

– Open up `Practical_06.R`