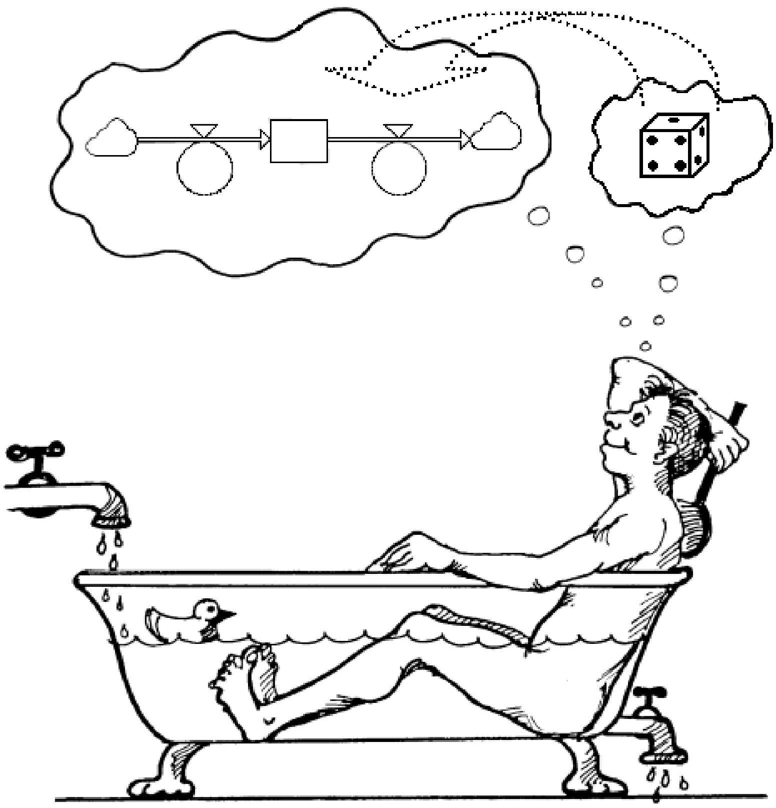
Deterministic vs. Stochastic Model Building and Simulation

Leif Gustafsson © LAB-3_Deterministic_vs_Stochastic_Model_Building.docx 2020-04-18

Contents:

Purpose:

To practice model building, simulation and analysis of various models – especially comparing deterministic and stochastic behaviours and results.



Random or not random – that is the question!

Name:	**Date:**
Course:	**Approved:**

1. Introduction - *to be read before the exercises*

1.1 Incomplete information and randomness

What is randomness? Does randomness exist in the real world? These are philosophical questions with a long history and debate. Without taking part in this debate, we will here take a pragmatic view.

For example, when throwing a die, you don't know the outcomes. But this doesn't mean that you have no information about a (fair) die. You already know that the outcome will produce an *integer value*, and that outcomes are *uniformly distributed* between 1 through 6 dots. We also expect the outcomes to follow an *irregular pattern* rather than e.g. 1, 2, 3, 4, 5, 6, 1, 2, …

However, from a physical point of view, the outcome depends on the initial position of the die, how hard you throw it, the elasticities of the die and the table, etc. Therefore, when talking about ***the system under study*** we will **NOT** use the words 'random', or 'stochastic'. We just have *incomplete information* (*uncertainty*) about the system under study or its behaviour. But we will NOT state that the die will disobey the strict laws of physics in some random way.

However, when making a ***model*** of the system under study, we cannot reproduce the outcome of a specific event (or when it will occur) because of *incomplete information*. However, we shoild still include the information we actually have about the system and its behaviour, e.g. about the *discrete* or *continuous statistical distribution* and that the outcomes are expected to be *irregular*. A model that includes randomness is called a *stochastic* model.

Randomness is implemented by drawing *random numbers* from an appropriate statistical distribution.

1.2 Why making a model stochastic?

In a deterministic CSS model, you describe both continuous matter and discrete entities as continuous amounts. Then the substance flowing to, from or between compartments is 'infinitively divisible', which means that it can be regarded as composed of an infinite number of infinitely small units. This implies that the *transition stochasticities* (see below) vanishes because of the law of large numbers.

The importance of being discrete

Modelling discrete entities as continuous flows usually leads to distortion of the model behaviour in many ways and often to disastrous results. For example:

- Except for special cases, it will result in biased estimates.
- All information about stochastic variations is lost.
- There is no possibility to calculate confidence intervals around estimates.
- A deterministic model will always produce categorical answers, e.g. that X will always win over Y, whereas a stochastic model will give the probabilities of X and Y as winners.
- Important phenomena, such as extinction of a species or elimination of an epidemic, which can happen in the system under study, will be lost in a deterministic model.
- Oscillations might disappear when they are not excited by stochasticity.
- The absence of irregularity can prevent queues from building up.
- The length of a process, e.g. an epidemic or a battle, varies between replications in a stochastic model, but is the same and usually biased in a deterministic model. In particular, an exponential decay will never reach zero.

Two main modelling rules are:

1) *Preserve the nature of the system under study in the model, i.e. describe discrete entities as discrete and continuous matter as continuous.*

2) *Include information you have about unexplained irregularities (usually by statistical distributions to draw random numbers from).*

1.3 Different types of uncertainties to handle in CSS modelling

There are different types of *uncertainties about a system under study* to be handled in model building. The uncertainty may concern the overall *Structure* of the system, but it can also be uncertainties about *Transitions* (flows) to, from or between stages, about the *Initial* situation in stages, about *Parameter* values, or about how a *Signal* (transferred information) is distorted or delayed.

In CSS modelling, *Transition uncertainty* is related to Flows, *Initial value uncertainty* to Stocks, *Parameter uncertainty* to Parameters, and *Signal uncertainties* to Links. These uncertainties can be handled by drawing *random numbers* from appropriate *statistical distributions*. *Structural uncertainty*, on the other hand, concerns the structure of the system under study, so it will require one model for each candidate structure.

In this lab, we will mainly consider *Transition uncertainty*, which is handled by random transfers (by a flow) of entities to, from or between compartments. In Lab-4, "**Stochastic Modelling of Uncertainties**", all types of uncertainties will be further discussed and exercised.

Transition stochasticity is directly involved in the dynamic evolution of the model, generated by the interaction between compartments and flows. In principle, transition stochasticity is handled by Poisson distributed random numbers according to: *Flow=PoissonFunction(argument)*.

1.4 Some fundamental differences between deterministic and stochastic models

There are many differences between deterministic and stochastic models, but three of fundamental importance are pointed out here.

Continuous matter or discrete entities?

- In a deterministic model, continuous matter is transferred between the Stocks in form of fractions. It is the 'infinite divisibility' that eliminates the transition stochasticity.
- In a stochastic model, discrete entities are transferred between Stocks in indivisible units. Usually discreteness requires a stochastic setting. The consequences of a discrete representation were described in Section 1.2, above.

The output from a model:

- A *deterministic model* creates output ***values***. One replication is enough.
- A *stochastic model* creates ***statistical distributions*** of output values. Therefore, it takes *many replications* of the model to create such a distribution.

What happens when the amount of matter or entities in a model is scaled?

- A *deterministic model* can often use both *absolute numbers* or *relative numbers* in the model. It is in this respect often *scalable*! This is not always true. When the model contains non-linear or bilinear relations, the size (amount of matter) can be crucial.
- A stochastic model, on the other hand, requires *absolute values*. It is *not scalable*. If you, for example, double the content (everything else the same), then the stochastic variations will not double – they will typically increase by a factor of $\sqrt{2}$, according to the law of large numbers. So, for very large number of entities the standard deviations will be negligible in comparison with the numbers.

2. Monte Carlo simulation - Stochastics without dynamics

Monte-Carlo simulation is a statistic method to perform estimates by multiple experiments. No dynamics are involved. Here follows a simple example of estimating π by Monte-Carlo simulation.

Exercise 1

The idea behind this exercise is that you throw darts at a board of say $2\times2 = 4$ m^2. At this board, imagine an inscribed unit circle with the area $C_{area} = \pi\cdot1^2 = \pi$ m^2, see the XY-plot in Figure 1. Therefore, if many darts are uniformly (but randomly) distributed over the board, the fraction hitting inside the unit circle should be about $\pi/4$. So, four times this fraction should be an estimate of π.

Task: Construct the model shown in Figure 1. X and Y are the random coordinates for the dart. Use Rand(-1, 1) for both X and Y. (You find it in 'Random Number Functions'.) Trial_Outcome tests if the dart will hit within the unit circle or not. Use the function IfThenElse(Condition, IfTrue, IfFalse): defined as IfThenElse([X]^2+[Y]^2 < 1, 1, 0). The hits are accumulated in HITS, and Pi_estimate calculates 4*HITS/Trials. By setting Length = 1000, DT = 1 and Trials = T() (instead of 1000), you can follow the actual estimate over 'time' (i.e. over the sequence of 1000 trials). However, to prevent *division by zero* in Pi_estimate=4*HITS/Trials at time zero, you can e.g. modify Trials to Trials = T()+a_very_small_Number (e.g. 1e-15).

Also include a Time Plot and an XY-Plot and some Number Boxes, as suggested in Figure 1. Open the XY-Plot and check 'Marker' but uncheck 'Line'. Also, uncheck 'Auto' scaling and set -1 and 1 for Min and Max of the X- and Y-axes.

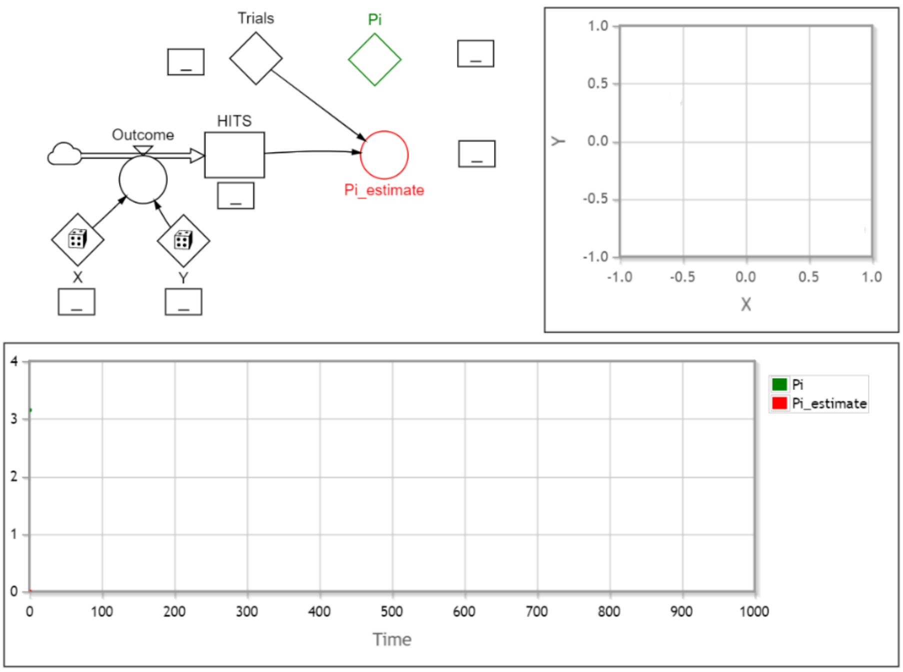


Figure 1. Estimation of π by Monte-Carlo simulation. (The circle in the XY-Plot was not included in StochSD – you may imagine it!) Note that time is *not* involved in Monte-Carlo simulation (Time unit: None). However, Length=1000 and DT=1will create a sequence of 1000 trials (darts) used to estimate the value of π.

First run one step at a time (▶|) for the first ten darts to see how they will hit the XY-Plot, and then then continue the sequence by Run (▶) to obtain the estimate. (Repeat the 1000-darts sequence a few times.) What was your estimate of π?

Answer: ....

.... ■

3. Three models of radioactive decay

Now we return to *stochastic* and *dynamic* modelling. Here we will examine three model approaches of radioactive decay: **Deterministic model**, **Adding noise to the deterministic model**, and a **Correct stochastic model**. The focus is here on the *transitions* from radioactive atoms to decayed.

Exercise 2

Construct the three models of radioactive decay, shown in Figure 2, whose behaviours we want to compare. (To build all three models in the same model window, we have to use different names for e.g. the undecayed atoms which we denote X, Y and Z, respectively.)

a) Deterministic model. where $X(t)$ represents the number of non-decayed radioactive atoms, with a decay parameter, c (decays per time unit). The initial value X(0)=30 atoms, c=0.1 and Decay_x=c*X. [Mathematically: $X(t+\Delta t) = X(t) - \Delta t \cdot c \cdot X(t)$.]

b) With noise added to the deterministic model. where $Y(t)$ represents the number of non-decayed radioactive atoms. (The same parameter c is used in all three models by ghosting. Mark c and then click the ghost symbol, and place the ghost!) Y(0)=30 atoms and Decay_y = c*Y + Noise, where Noise = RandNormal(0, 6). [Mathematically: $Y(t+\Delta t) = Y(t) - \Delta t \cdot c \cdot Y(t) + Noise(t)$, where $N(t)$ is a zero-mean white noise with standard deviation=6, here assumed normally distributed, i.e. Normal(0, 6).]

c) Stochastic model. where $Z(t)$ represents the number of non-decayed radioactive atoms. Z(0)=30 atoms and Decay_z = PoFlow(c·Z). [Mathematically: $Z(t+\Delta t) = Z(t) - Poisson[\Delta t \cdot c \cdot Z(t)] / \Delta t$.]

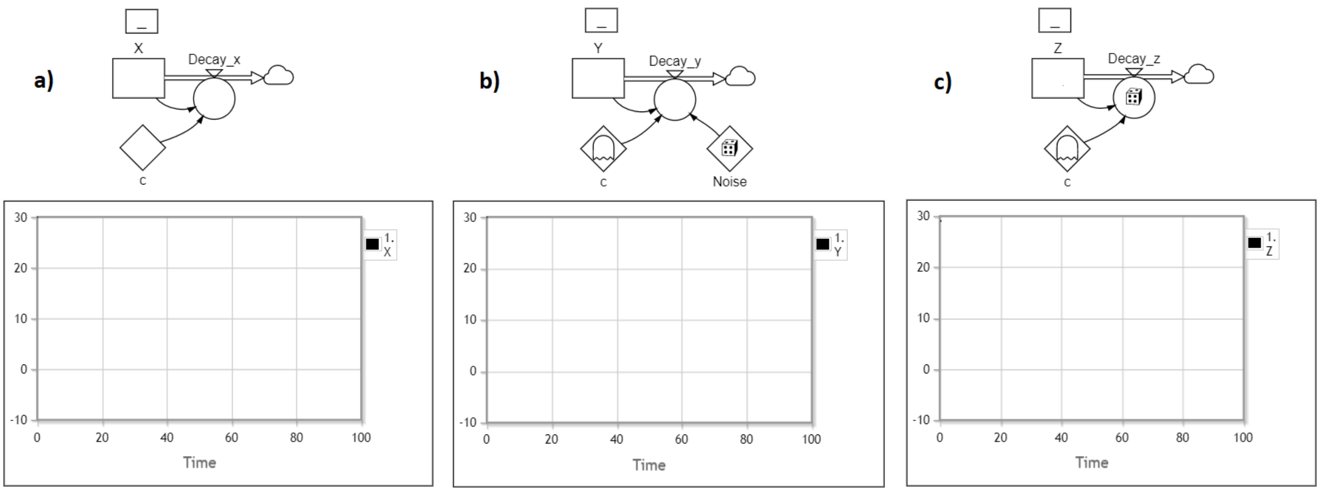


Figure 2. Three models of radioactive decay.

Simulate the models over 100 time units with DT=0.1 a number of times and sketch a typical behaviour for each model.

As illustrated by your results, the models might produce a number of artefacts. In the table, below. Answer the issues 1-6 by setting a mark (e.g. with ×) for each artefact that you find for respective model. (We will proceed with issues 7-9 later on.)

	ABSURD ARTEFACTS	a) Deterministic model	b) Added noise	c) Stochastic model
1	The number of atoms decrease continuously instead of stepwise			
2	Non-integer numbers of the remaining atoms will occur			
3	The size of stochastic variations is not related to remaining number of atoms			
4	A negative number of atoms may occur			
5	Sudden increases of the number of atoms can occur			
6	Variations will continue even after all atoms have decayed			
7	The model produces no or erroneous significance intervals around estimates			
8	The average time until the last decay is unspecified or incorrect			
9	The stochastic behaviour will strongly depend on the time-step used (even when DT is sufficiently small)			

The artefacts in this example may seem innocent, but when part of a larger model, they may generate severe consequences that may be hard to trace back to their root cause. Variations without appropriate reasons may excite other parts of the model. A negative number of entities may trigger various phenomena, e.g. driving other processes backwards. Furthermore, if e.g. the logarithm or the root of the number of remaining atoms is used, the model will crash when this number becomes negative. *Trying to make a deterministic CSS stochastic by adding or multiplying noise to the transitions is never a good idea, since it creates a number of artefacts.* ■

Exercise 3

After including transition stochasticity in an erroneous way in Model (b) and in a correct way in Model (c), we will now analyse the statistics from many replications (simulation runs) by using the StatRes tool.

Task: Reduce the simulation time, 'Length', to 20 time units so that there are usually some undecayed atoms left. Under the Tools menu, open StatRes. In its 'Result quantity' field, Enter and Add X, Y and Z. Set 'Requested Simulations' to 1000, and press Run. Fill in the results in the Table, below.

	Results from StatRes	a) Deterministic model	b) Added noise	c) Stochastic model
1	Average			
2	Standard deviation			
3	Min and Max	,	,	,
4	95% Confidence Intervals (In practice, |values| < 1E-10 means zero)	–	–	–

Task: Now you can also complete issues 7-9 in the 'Absurd Artefacts' table. ■

4. Models of three very different systems

Continuous modelling removes many of the characteristics of a system because transition stochasticities vanish. Here we will demonstrate that three models with fundamentally different structures can behave identically when modelled as continuous but very differently when modelled as discrete.

Exercise 4

a) First, consider a *discrete logistic growth model* with a population size *POP* that growth proportionally by *Births* and declines quadratically by *Deaths* because of competition (each individual competes with every other individual). Mathematically:

$POP(t+\Delta t) = POP(t) + \Delta t{\cdot}a{\cdot}POP(t) - \Delta t{\cdot}b{\cdot}POP(t)^2$; where $POP(0)=1$, $a=0.1$ and $b=0.01$.

This is an *open, first-order, non-linear model*. The model is shown in Figure 3a.

b) Second, consider an epidemic *SI model* where susceptible individuals (*S*) are infected by encounters with infectious individuals (*I*). The risk of infection per time unit is proportional to the number of susceptibles, to the number of infectious subjects and to a constant *c*. Mathematically:

$S(t+\Delta t)=S(t) - \Delta t \cdot Infect(t)$

$I(t+\Delta t)=I(t) + \Delta t \cdot Infect(t)$

$Infect(t) = c \cdot S(t) \cdot I(t)$

where $S(0)=9$, $I(0)=1$ *and* $c=0.01$.

This is a *closed, second-order, bilinear model (linear in both S and I)*, shown in Figure 3b.

c) Third, consider a discrete 'pruned' model where prey births and predator deaths are removed (why we calle the model 'pruned'). In a prey-predator a model, a fraction ($d \cdot x \cdot y$) of the prey (x) are killed each time unit in encounters with predators (y). These encounters give the predators capacity to breed according to ($e \cdot x \cdot y$). Mathematically:

$X(t+\Delta t) = X(t) - \Delta t \cdot Deaths\_X(t)$

$Y(t+\Delta t) = Y(t) + \Delta t{\cdot}Births\_Y(t)$

$Deaths\_X(t) = d \cdot X(t) \cdot Y(t)$

$Births\_Y(t) = e \cdot X(t) \cdot Y(t)$

where $X(0)=9$, $Y(0)=1$ and $d=e=0.01$.

This model consists of two *open and coupled first-order submodels*, shown in Figure 3c.

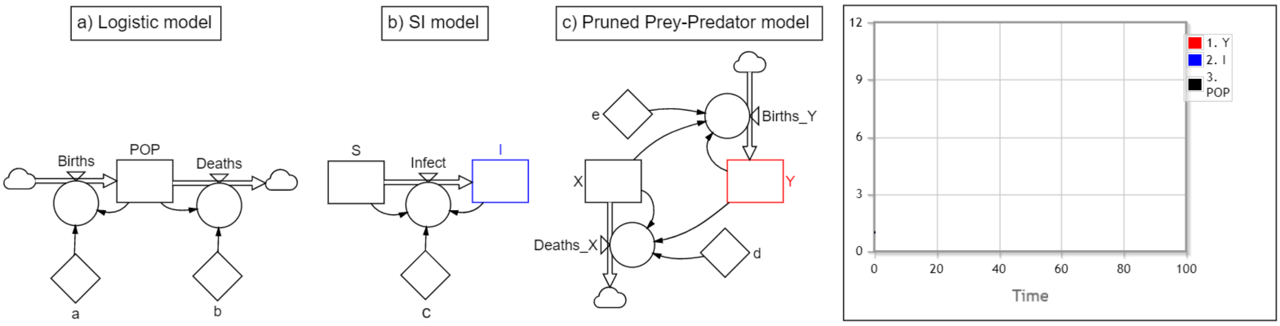


Figure 3. Forrester diagram of a *deterministic* Logistic model, an SI model and a Pruned prey-predator model.

Make a simulation of the three deterministic models during 100 time units with DT=0.1. Plot POP, I and Y in the same Time Plot and sketch the results in Figure 3. What dou you see?

Answer: ....

.... ■

Exercise 5

Now, add *transition stochasticity* to the three models by including the *PoFlow*( ) clauses in the flow equations, e.g. Infect=c*S*I → Infect=PoFlow(c*S*I). We then obtain a *corresponding* stochastic models with discrete individuals. In Figure 4, the three stochastic models are shown.

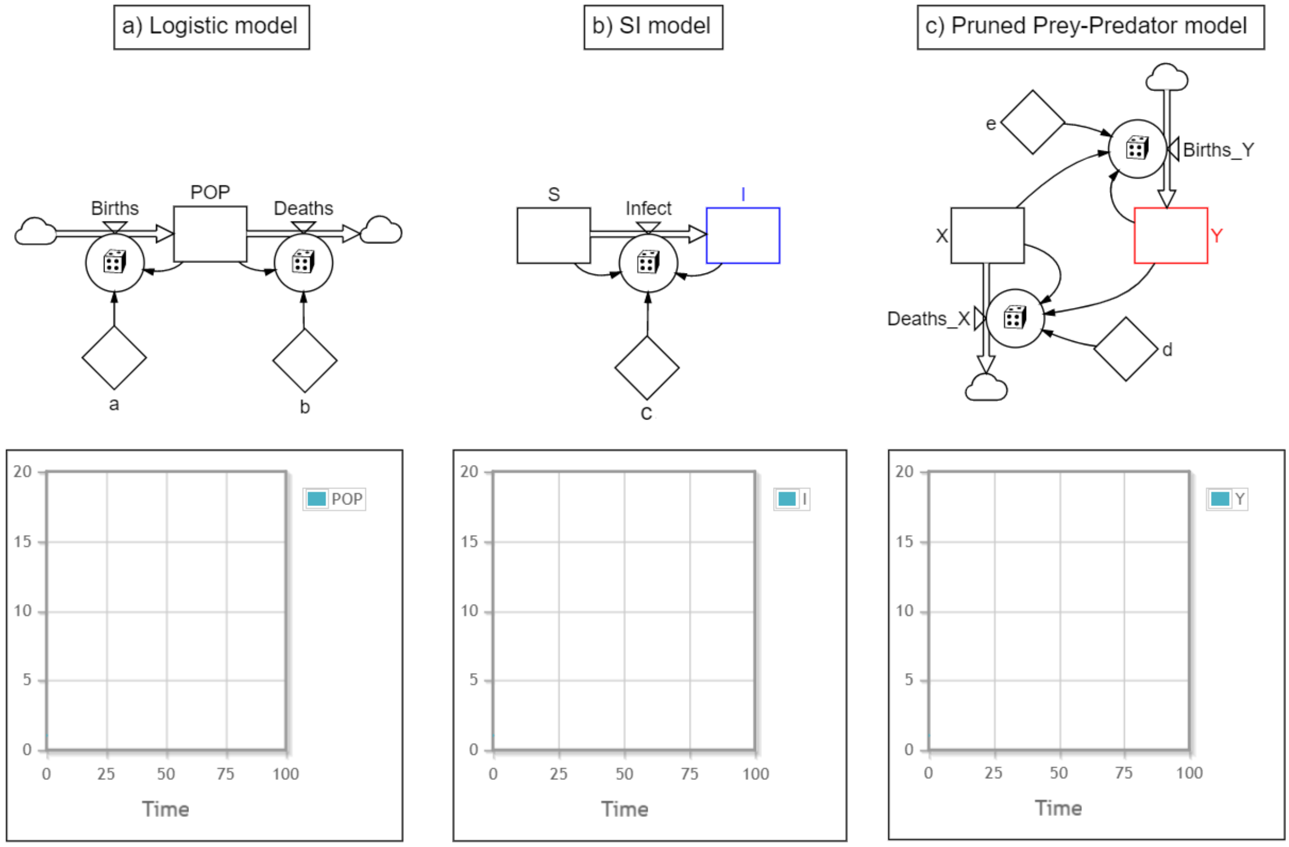


Figure 4. Forrester diagram of a *stochastic* Logistic model, an SI model and a Pruned prey-predator model.

This time, use separate **Compare Simulation Plots** for each model. (This type of plot lets you keep old replications if you check the ‘Keep Results’ box. Scale the Plot from 0 to 20. Make three to five replications of each model and sketch the results of the replications of each model.

Task: As seen from the replications, the stochastic approach brings forth the differences between the behaviours of the three models. So check (✕) the appropriate answer(s) in the table, below.

	Results from the stochastic models	**A) Stochastic Logistic model (POP)**	**B) Stochastic SI model (I)**	**C) Stochastic prey-pred. model (Y)**
1	Which model(s) will always end at the same level?			
2	Which model(s) can both increase and decrease?			
3	Which model(s) may extinct all individuals? (Although it happens seldom.)			

Finally, in one of the models, the stocastics preserves the risk of extinction. Explain how this can happen in that model.

Answer: ....

.... ■

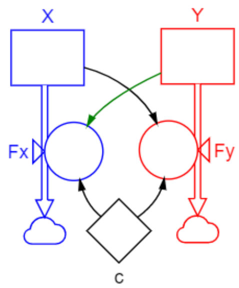
5. Lanchester's model of warfare

This example will illustrate that *linearity* alone does not ensure correct results for a deterministic model.

Exercise 6

Lanchester's model of warfare was first published in 1916. It is a simple model of a battle between two fighting forces (soldiers, aircrafts, battle ships, etc.). Force X is characterised by its initial number of entities, $X(0)$, and hitting power, a, and Force Y by its initial number, $Y(0)$, and hitting power, b. The hitting power is the number of eliminated entities per time unit that each member can inflict on the enemy. Here we let the hitting powers be the same, c (i.e. $c = a = b$).

The combat proceeds until the last entity of the losing force is eliminated. A deterministic model is shown in Figure 5.



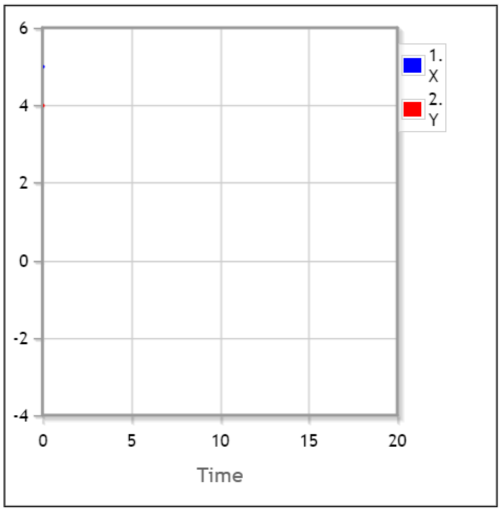


Figure 5. A deterministic Lanchester model where Force X and Force Y have the same hitting powers *c*.

Mathematically, the model is:

$X(0) = 5$ (Initial value of X.)
$Y(0) = 4$ (Initial value of Y.)
$Fx = c \cdot Y$
$Fy = c \cdot X$
$c = 0.1$

Task: Build the deterministic Lanchester model and simulate it for 20 time units with DT=0.1.

Perhaps you will be surprised over what you see. Describe the strange behaviour after all entities in the loosing force are eliminated!

Answer: ....

....

....

Yes, the model does what you have specified! A Stock can take both positive and negative values in analogue to e.g. a mathematical equation. This is necessary to be able to model all types of systems. For example, your bank account may be positive or negative. You can go over or under the sea level.

However, when the last Y is eliminated in this model, Force X will continue to drain the Stock holding Force Y so it gets a negative content. Further, since $Fx = c \cdot Y$ and $Y < 0$, it means that *Fx* will *become an inflow* that increases Force X. There are several ways to prevent these absurdities. But the simplest way is to terminate the simulation when force Y becomes zero.

Task: Therefore, insert an Auxiliary, open it and name it ‘StopIf’. Then, find the IfThenElse(Test Condition, Value if True, Value if False) function among ‘General Functions’. Rewrite it to: IfThenElse([Y]<0, Stop(), 0) and draw a link from the Stock Y to this Auxiliary. Stop() will promptly terminate the simulation (*without showing the last time-step*) and ‘0’ means ‘No action’.

To include the last time-step, replace [Y]<0 by Delay([Y], DT()), giving:
StopIf = IfThenElse(Delay([Y], DT())<0, Stop(), 0).

Also include a Parameter, name it ‘BattleTime’ (or ‘BT’) and include the Time function: T(). Also associate this parameter with a Number Box.

Now we are interested in the questions:

	Deterministic Lanchester model	**Answer:**
1	Which side will always win?	
2	What number of entities will remain after the battle? (For a more precise answer, set DT=0.01.)	
3	How long will the fight take? (For a more precise answer, set DT=0.01.)	

Comment: The number of remaining entities after the battle can be calculated from *Lanchester’s quadratic law* [1], which states that: $X^2(BT) - X^2(0) = Y^2(BT) - Y^2(0)$, where X(0) and Y(0) are the initial numbers and X(BT) and Y(BT) are the final numbers (i.e. Y(BT)=0). This gives $X^2(BT)=5^2-4^2=9$, i.e. X(BT)=3. Did you get (almost) that answer in the table above?

Answer: ....

From this law you can draw the conclusion that it is better to fight a smaller part of the enemy forces at a time than to meet them all in a single battle. ■

Exercise 7

Lanchester’s analysis is simple and beautiful, but is it correct? Does the stronger Force always win? Is the number of remaining entities in the winning Force a correct estimate? Is the estimated BattleTime correct? To answer these questions the assumptions of Lanchester’s model are kept, but the number of entities eliminated for each time-step is now to be *integer* and *stochastic*. Therefore, modify the two outflows according to:

Fx = c·Y → Fx = PoFlow(c·Y)
Fy = c·X → Fy = PoFlow(c·X)

[1] **For the mathematically interested**: The Lanchester model as differential equations is: $dX/dt = -c \cdot Y$, and $dY/dt = -c \cdot X$. Dividing the first equation by the second then gives $dX/dY = Y/X$ or $\int X\, dX = \int Y\, dY$. When integrated over the BattleTime of the fight from 0 to BT. This gives: $X^2(BT) - X^2(0) = Y^2(BT) - Y^2(0)$ if the forces are initially X(0) and Y(0).

Also modify the termination criterion to:

IfThenElse(Delay([X], DT()) < 0.5 OR Delay([Y], DT()) < 0.5, Stop(), 0),

which delays the termination one time-step to include the last time-step in a table or plot.

Also make it possible for StatRes to count the number of victories for X and Y by including two Auxiliaries Xwin and Ywin defined by:
IfThenElse([X]>0.5, 1, 0)   (Because X and Y stays integer, you can e.g. use >0.5 to see)
IfThenElse([Y]>0.5, 1, 0).   (whether there are any entities left – i.e. separating 0 from 1.)

Prolong the Length to a large value, so that you are sure all battles will end. (You can use a very large value because the simulation will stop when a battle is over.) Open StatRes and Add: X, Xwin, Y, Ywin, and BattleTime. Make 1000 replications and answer the questions in the Table below.

	Stochastic Lanchester model	**Answers from 1000 replications:**
1	Which side will win?	Average X_{win} = 95% C.I. = – Average Y_{win} = 95% C.I. = –
2	What number of entities will remain after the battle?	Expected number of survivors for X: Expected number of survivors for Y:
3	How long will the fight take?	Average BattleTime = 95% C.I. – Min = Max =

Finally, compare the results for the *stochastic* and *deterministic* models about X and Y winning the battle, the expected number of survivors for the winning side, and the battle time. What do you find? In particular, which of the differences are *significantly* different based on 1000 replications. (This is the case if the *deterministic result* lies outside the *confidence interval* of the *stochastic results*.)

Answer: ....

....

....

.... ■

Exercise 8

Hopefully, you found that Lanchester's quadratic law does not hold! However, it may still be better to fight a smaller part of the enemies at a time than to meet them all in a single battle. This question we will now investigate.

Therefore, we will test the strategy for Force X to use a single unit at a time when defeating Force Y. So, when one X-unit is defeated, a new is sent into the battle. Therefore, build the model shown in Figure 6. Initiate X_Reserve to 5 and X to 0. Set F_next = IfThenElse([X]<0.5, 1/DT(), 0). ([X]<0.5 works well since X is an integer). Because 1 unit at a time is sent to the battle during a time-step, the flow rate during this DT must be 1/DT() to obtain DT*1/DT=1 entity. Also modify StopIf to:

IfThenElse(Delay([X_All], DT()) < 0.5 OR Delay([Y], DT()) < 0.5, Stop(), 0),

where: ([X_All] = [X] + [X_Reserve].

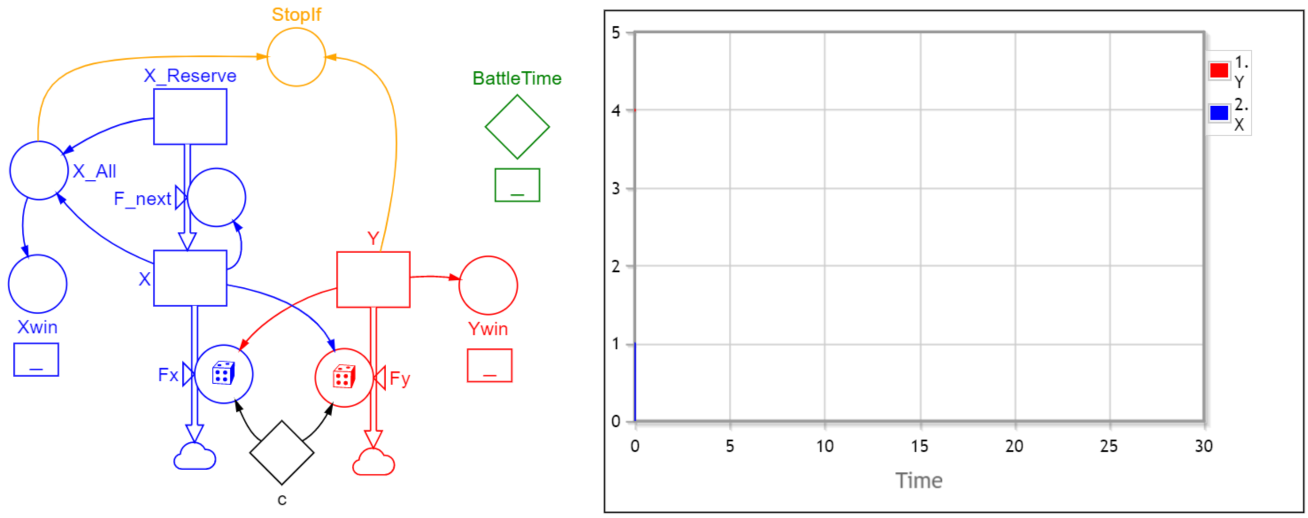


Figure 6. A stochastic Lanchester model where Force X tries to defeat Force Y by engaging one entity at a time.

Make sure that Length is enough. Use DT=0.1, adjust the Time Plot to show 30 time units, and make a some replication. Observe X and Y in a Time Plot and sketch what you see in Figure 6.

Finally, Prolong the Length to a large value (so you are sure all battles will end). Open StatRes and Add: X_All, Xwin, Y, Ywin, and BattleTime. Make 1000 replications and answer the questions in the Table, below.

	Stochastic Lanchester model where X uses one unit a time	**Answers from 1000 replications:**
1	Which side will win?	Average X_{win} = 95% C.I. = – Average Y_{win} = 95% C.I. = –
2	What number of entities will remain after the battle?	Expected number of survivors for X_All: Expected number of survivors for Y:
3	How long will the fight take on average?	Average BattleTime =

Is it the stronger Force X or the weaker Force Y that wins in this case? Also, draw a relevant conclusion about the strategy.

Answer: ....

....

.... ■

In this lab, we have mainly studied the effects of Transition stochasticity. How to handle other types of uncertainties will be the subject in Lab-4: "**Stochastic Modelling of Uncertainties**". *There, all types of uncertainties will be further discussed and exercised.*