Randomness and uncertainty in SBML

Darren Wilkinson
School of Mathematics & Statistics
and
Centre for Integrated Systems Biology of Ageing and Nutrition
Newcastle University, UK

2010 SBML Hackathon
Seattle, USA
1–4 May, 2010

distrib

- `distrib` proposal based around old proposal (on sbml.org wiki) from Newcastle, implemented in BASIS/gillespie2
- Adds random numbers from various distributions to MathML subset used by SBML
- Proposal just defined additional `csymbols`, but other ways to do it
- Random numbers are useful for assigning initial values for parameters and species amounts, in event assignments, event delays and potentially for timed events (though care must be taken)

Rationale

- Generally useful for describing uncertainty about parameters (especially reaction rate constants) and about initial amounts for species
- Actually a philosophical difference between a description of uncertainty and the mechanism of choosing a random number
- Old proposal didn't make that distinction — could be relevant for certain kinds of analysis (including parameter scans)
- `UniformRandom` also best way to describe ranges — defines the natural scale — eg. for parameter scans
- Must not use random numbers in rate laws or other structures that are evaluated continuously — leads to undefined behaviour of the model — but see later...

Questions

1. Make a distinction between describing uncertainty with a random quantity and picking one?
2. How many distributions? Trade-off between generality and implementation burden
3. Include PMFs/PDFs/CDFs etc, as well as actual distributions
4. How to encode in XML/SBML? To be nice to tools which don't understand...

Particle representations

- Describing uncertainty with standard distributions is insufficiently general and flexible for describing uncertainty in biological models — especially joint distributions and distributions arising from Monte Carlo algorithms — uncertainty often not in standard form
- Often better to represent joint distribution over uncertain parameters with a large number of realisations from the distribution
- Big table — columns represent variables (eg. parameters and species), and rows represent realisations
- Can sample from the joint distribution simply by picking a row
- Can also represent parameter designs and scans this way — again, possible distinction between uncertainty description and mechanism of sampling

Benefits

- Very natural for use in conjunction with Monte Carlo algorithms for both uncertainty analysis and parameter inference (Bayesian, MCMC, or otherwise)
- The output from MCMC algorithms, representing posterior uncertainty, has to be represented this way, and represents prior uncertainty for the next experiment...
- We have implemented this representation in CaliBayes (`www.calibayes.ncl.ac.uk`), using a simple XML format external to the SBML
- Semantics — every variable in the table has its SBML values over-ridden by values picked from rows of the table
- Better in SBML? Big files, often containing millions of values...

UncertML

- Systems biologists are obviously not the only people who worry about the representation of uncertainty in data, parameters and computer experiments...
- Draft specification for UncertML `www.uncertml.org`
- This covers essentially everything covered so far (and more), including representation of uncertainty using both algebraic distributions and particle representations
- Very heavy-weight specification — would seriously hinder implementation if we were to adopt it all
- More focused on representation of uncertainty than on sampling from random distributions

Extrinsic noise

- Discrete stochastic simulators are typically concerned with "intrinsic noise" in gene expression due to discreteness of biomolecules
- However, many other sources of noise and heterogeneity in biological systems
- "Extrinsic noise" is often modelled by allowing reaction rate constants to vary randomly and continuously in time according to a stochastic process governed by a stochastic differential equation (SDE)
- As mentioned previously, you can not just add random numbers into rate laws — the models will "blow up"
- Need a mechanism for defining SDEs within SBML models to make this work

Implications

- Actually very easy from the point of view of SBML and libSBML
- Essentially just need a new `stochasticRateRule` to replace the existing rate rule, that will allow the definition of SDEs
- However, there are very significant implications for tool support — will require simulator developers to be competent in numerical stochastic integration (not easy), and also users to be familiar with stochastic differential equation modelling
- Therefore important to keep this separate from other random package(s) — other stuff is easy to implement and very useful
- Actually possible to implement SDEs approximately and inefficiently using existing SBML and discrete stochastic simulators, but involves exceptionally ugly hacks...