DRAFT: XML description of models of character evolution in `CIPRES`

Mark Holder

June 26, 2006

Contents

1 Introduction

CIPRES needs a way to describe analysis assumptions in several contexts. For example, the current rec-I-DCM3 GUI is unsatisfactory because there is no user-interface for entering the desired style of analysis. The user must be sure that they select large and small tree inference engines that perform evaluate trees in the same way. Ideally, there would be a user interface for describing a type of analysis (e.g. a model of character evolution). Based on these selections, the GUIGen tool could filter the tree improver choices (if the service's UI-XML advertised which analysis styles they support).

A text format for describing models seems necessary because of the diversity of possible models and lack of an accepted standard – a data structure in IDL would have to be updated frequently as we find better solutions to initial hacks or as we realize that we need more flexibility. At least initially it seems likely that we will fall back on passing models around as strings. Because NEXUS does not provide an accepted solution to defining models (in a public block), it seems logical to devise and XML standard. Once we are happy with an XML specification, we can move to an IDL based struct if that seems helpful.

I'm unsure of the best answer to a few of high level questions:

- Should "model" just refer to a probabilistic model of character of evolution or should we be calling parsimony step-matrices "models"?
- Should we use separate structure for the specific instances of a model (e.g. HKY85 with $\kappa = 2.4, \pi_A = 0.3, \pi_C = 0.1, \pi_G = .1$) vs a description of a class of models (e.g. HKY85 with no parameters specified)?
- Should prior distributions be considered a part of the model or external to it?

In this draft, I'll assume that we use attributes of model to denote a probabilistic (vs parsimony or distance) model and to differentiate between discrete and continuous data. I'll pursue a generic model description that could be used to describe a class of models or a particular instance of a model. There will be a place for priors in the model itself. We'll see if we hate it.

1.1 Assignment of models to data

Conceptually, models are separate from the data. Clearly, we will need a mechanism for describe the scope "where" the user thinks a model should apply. A model must be assignable to:

1. a set of characters
2. a portion of the tree
3. a scope that depends on the state of some other character (possibly an inferred state or condition).

Presumably this information about where to apply the model should be separate from the model description. Case 3 is difficult – if the information about the "other" character is not present, then

specifying this type of scope essentially forces the model-fitting module to implement some kind of dependence model of the kind described by **?**. In this instance it would seem better to put that in the inter-character dependence in the model itself. On the other hand, in some cases the "other" character may have been inferred by another mechanism, thus this form of dependence seems most appropriate as an external description of model scope.

1.2 Probabilistic models of discrete characters

A statistical model is system for defining a probability distribution over all possible outcomes of a process. Usually the model has parameters that control the exact shape of the distribution. Thus, to describe a model we need to:

- provide a list of parameters,
- describe the type of data the model applies to, and
- convey how the parameters are used to make probability statements about the data.

The easiest way to describe how the parameters make probability statements would be to describe the likelihood equation in some mathematical description format like MathML. Unfortunately this would be extremely difficult to read, interpret, and it would make it very hard to provide an API that provides programmer answer to the common questions about a model.

It is certainly possible to envision model that depend on the tree and its parameter (e.g. branch lengths). Making the distinction between Character models and Tree-Character models seems valuable (since I'm not aware of any implementation of the latter category we can safely ignore them right now – and assume they will have their own XML description).

2 The `model` element

2.1 Attributes

- `id` unique identifier for an instance
- `type` ∈ { probabilistic | parsimony | distance | ... }
- `datatype` IDREF. A Reference to a standard model or previously described datatype. Examples of standard types (obviously we could shorten the id's I'll use descriptive strings):
  - "Aligned DNA nucleotide" – for a model does not accommodate the indel process, and it only assigns probabilities to the nucleotides (the gaps in the alignment are not treated as some kind of fifth base).
  - "Aligned DNA and gaps" – requires an alignment, but models the gaps that are in the data (without trying to realign them).
  - "Unaligned DNA" – models the indel process.
- `class-name` to display to the user to describe the model family (e.g Jukes-Cantor)

- `name` user-defined name for the model
- `time-reversible` boolean - True for time-reversible models
- `probability-source` { rates | invariant } Models that do not use branch lengths (constant probability of transitions or the invariant model) are possible. Constant probability models would contain a `p-matrix` instead of `r-matrix`, and the invariant model would not contain any matrix (p, q, or r)

2.2 Elements inside `model`

- `parameter` describes a parameter of the model (value may be unknown)
- `r-matrix` describes the symmetric component of the instantaneous rate matrix (time-reversible models only)
- `equil-state-frequencies` can be fixed values for any model and parameters only for time-reversible models.
- `q-matrix` describes the instantaneous rate-matrix (non-reversible models only)
- `rate` multiplier for the model used to scale a branch length, which is assumed to (defaults to 1.0)

2.2.1 `parameter`

attributes

- `name`
- `id`
- `range` ∈{ non-negative | positive | unbounded | zeroToOne | generic} generic bounds could be used for parameters that have flexible bounds that depend on other parameters (I can't think of any parameters like this, so I suppose that we can postpone developing the syntax for the generic system).

elements

- `value` can be empty if the value is unknown, otherwise will contain a real number (the `estimator-type` indicates where this value has been (or should be) estimated. Contains attributes that describe what aspect of the parameter is described by the value:
  -
  - `estimator-type` ∈ {ML | MAP | posterior mean | user-specified | random | quantile | . . . }
- `posterior` description of a probability density that approximates the posterior distribution of the parameter.
- `prior` description of a probability density to be used (or which was used to infer the parameter
-

2.2.2 r-matrix

The q-matrix is used for non-reversible models, but is other wise identical.

attributes

- parameterization ∈ {relative | sum-to-one }

elements

- cell contains attributes:
  - from and to attributes identify the cell of the matrix.
  - parameter IDREF. Should contain the id of a parameter
  - reference should be true for the rate that is the reference rate (when parameterization is relative, other rates are measures with respect to this rate). The reference rate is fixed at 1.0 (so this element should not have a parameter attribute).

3 Examples

It may be desirable to have some predefined models that are identified by names –like JC. I'll delay that and describe JC as if it were any old model.

3.1 Jukes-Cantor

```
<model type="probabilistic" datatype="Aligned␣DNA␣nucleotide"
   time-reversible="true">
    <equil-state-frequencies>
        <fixed>0.25</fixed>
        <fixed>0.25</fixed>
        <fixed>0.25</fixed>
        <fixed>0.25</fixed>
    </equil-state-frequencies>
</model>
```

We could make the last state frequency optional.

3.2 Kimura 1980

κ unknown (ml estimate is requested)

```
<model type="probabilistic" datatype="Aligned␣DNA␣nucleotide"
   time-reversible="true">
```

```
    <parameter name="kappa" range="nonnegative" id="315351"
       estimator-type="ML"  />
    <r-matrix parameterization="relative">
        <cell from="0" to="1" reference="true"/>
        <cell from="0" to="2" param="315351"/>
        <cell from="1" to="3" param="315351"/>
    </r-matrix>
    <equil-state-frequencies>
        <fixed>0.25</fixed>
        <fixed>0.25</fixed>
        <fixed>0.25</fixed>
        <fixed>0.25</fixed>
    </equil-state-frequencies>
</model>
```

Note that the from and to are denoted in terms of the index of the state

4 References