The Paraphrased, Human-Readable Adaptation of SED-ML

Lucian Smith, Kiri Choi, Kyle Medley, Herbert Sauro
University of Washington, NIH GM081070.

Purpose

- Allow fast creation of SED-ML documents
- Allow viewing of SED-ML documents in human-readable form
- Integrate with Python/Tellurium for model creation and simulation.
- SED-ML L1v2

Example

```
//Created by libphrasedml v0.4 beta
// Models
model_1 = model "00001-sbml-l3v1.xml"

// Simulations
simulation_1 = simulate uniform(0, 5, 51)

// Tasks
task_1 = run simulation_1 on model_1

// Outputs
plot time vs S1 / compartment, S2 / compartment
report time, S1 / compartment, S2 / compartment
```

General format:

[variable] = [keyword] [details]

model_1 = model "00001-sbml-l3v1.xml"
simulation_1 = simulate uniform(0, 5, 51)
task_1 = run simulation_1 on model_1

phraSED ML

Models:

[variable] = model “[filename]”

model_1 = model "00001-sbml-l3v1.xml"

Models:

[variable] = model “[filename]”

[variable] = model [othermodel]

model_1 = model "00001-sbml-l3v1.xml"

model_2 = model model_1

Models:

[variable] = model “[filename]”

[variable] = model [othermodel] with [changes]

model_1 = model "00001-sbml-l3v1.xml"

model_2 = model model_1 with S1=3

Models:

[variable] = model “[filename]”

[variable] = model [othermodel] with [changes]

model_1 = model "00001-sbml-l3v1.xml"

model_2 = model model_1 with S1=3, S2=S1+4

Models:

[variable] = model “[filename]”

[variable] = model [othermodel] with [changes]

model_1 = model "00001-sbml-l3v1.xml"

model_2 = model model_1 with S1=3, S2=S1+4

Translates to attribute change

phraSED ML

Models:

[variable] = model “[filename]”

[variable] = model [othermodel] with [changes]

model_1 = model "00001-sbml-l3v1.xml"

model_2 = model model_1 with S1=3, S2=S1+4

Translates to computeChange

phraSED ML

Models:

[variable] = model “[filename]”

[variable] = model [othermodel] with [changes]

model_1 = model "00001-sbml-l3v1.xml"

model_2 = model model_1 with x=3, S2=x^2

phraSED ML

Models:

[variable] = model “[filename]”

[variable] = model [othermodel] with [changes]

model_1 = model "00001-sbml-l3v1.xml"

model_2 = model model_1 with x=3, S2=x^2

No ‘x’ in model: becomes local parameter

phraSED ML

Models:

[variable] = model “[filename]”

[variable] = model [othermodel] with [changes]

model_1 = model "00001-sbml-l3v1.xml"

model_2 = model model_1 with x=3, S2=x^2

Still translates to computeChange

phraSED ML

Models (future work?):

[variable] = model “[filename]”

[variable] = model [othermodel] with [changes]

model_1 = model "00001-sbml-l3v1.xml"

model_2 = model model_1 with ~~add p1=3~~

model_2 = model model_1 with ~~remove J0~~

Simulations:

[variable] = simulate [simulation]

sim1 = simulate steadystate

Simulations:

[variable] = simulate [simulation]

sim1 = simulate steadystate

sim2 = simulate uniform(0,10,100)

sim3 = simulate uniform(0,5,10,50)

Simulations:

[variable] = simulate [simulation]

sim1 = simulate steadystate
sim2 = simulate uniform(0,10,100)
sim3 = simulate uniform(0,5,10,50)
sim4 = simulate uniform_stochastic(0,10,100)
sim5 = simulate uniform_stochastic(0,5,10,50)

Simulations:

[variable] = simulate [simulation]

sim1 = simulate steadystate

sim2 = simulate uniform(0,10,100)

sim3 = simulate uniform(0,5,10,50)

sim4 = simulate uniform_stochastic(0,10,100)

sim5 = simulate uniform_stochastic(0,5,10,50)

sim6 = simulate onestep(0.5)

Tasks:

[variable] = run [simulation] on [model]

task1 = run sim1 on model1

Repeated Tasks:

[variable] = repeat [task] for [loop]

task1 = run sim1 on model1

task2 = repeat task1 for S1 in uniform(1,10,50)

phraSED ML

Repeated Tasks:

[variable] = repeat [task] for [loop]

task1 = run sim1 on model1

task2 = repeat task1 for S1 in uniform(1,10,50)

task3 = repeat task1 for S1 in logUniform(1,10,20)

Repeated Tasks:

[variable] = repeat [task] for [loop]

task1 = run sim1 on model1
task2 = repeat task1 for S1 in uniform(1,10,50)
task3 = repeat task1 for S1 in logUniform(1,10,20)
task4 = repeat task1 for S1 in [1, 3, 6, 10]

Repeated Tasks:

[variable] = repeat [task] for [loop], reset=true

task1 = run sim1 on model1

task2 = repeat task1 for S1 in [1,3,6], reset=true

Repeated Tasks:

[variable] = repeat [task] for [loop], [changes]

task1 = run sim1 on model1

task2 = repeat task1 for S1 in [1,3,6], S2=3

Repeated Tasks:

[variable] = repeat [task] for [loop], [changes]

task1 = run sim1 on model1

task2 = repeat task1 for S1 in [1,3,6], S2 in [3,6,7]

phraSED ML

Repeated Tasks:

[variable] = repeat [task, task, …] for [loop]

task1 = run sim1 on model1

task2 = run sim1 on model2

task3 = repeat [task1, task2] for S1 in [1,3,6]

Repeated Tasks:

[variable] = repeat [task, task, …] for [loop]

task1 = run sim1 on model1

task2 = run sim1 on model2

task3 = repeat [task1, task2]
for model1.S1 in [1,3,6],
model2.S1 in [5,5,8]

Output:

plot [variable] vs [variable]

plot S1 vs S2

Output:

plot [variable] vs [variable]

plot S1 vs S2

plot task1.S1 vs task1.S2

Output:

plot [xvariable] vs [yvariable1], [yvariable2]

plot time vs S1, S2

Output:

plot [xvariable] vs [yvariable] vs [zvariable]

plot S1 vs S2 vs S3

Output:

plot [xvariable] vs [yvariable1] vs [zvariable1],
[yvariable2] vs [zvariable2]

plot time vs S1 vs S2, p1 vs p2

Output:

plot [formula] vs [formula]

plot time vs S1/compartment1

plot time vs p1, p1^2, p1^4

plot S1/compartment1 vs S2/compartment1

Output:

report [formula], [formula], [formula] […]

report time, S1, S2, S3/comp1

Examples: 1d parameter scan

```
model1 = model "oscli.xml"
timecourse1 = simulate uniform(0, 20, 1000)
task0 = run model1 with timecourse1
task1 = repeat task0 for J0_v0 in [8, 4, 0.4] with reset
plot task1.time vs task1.S1, task1.S2
```

Examples: 2d parameter scan

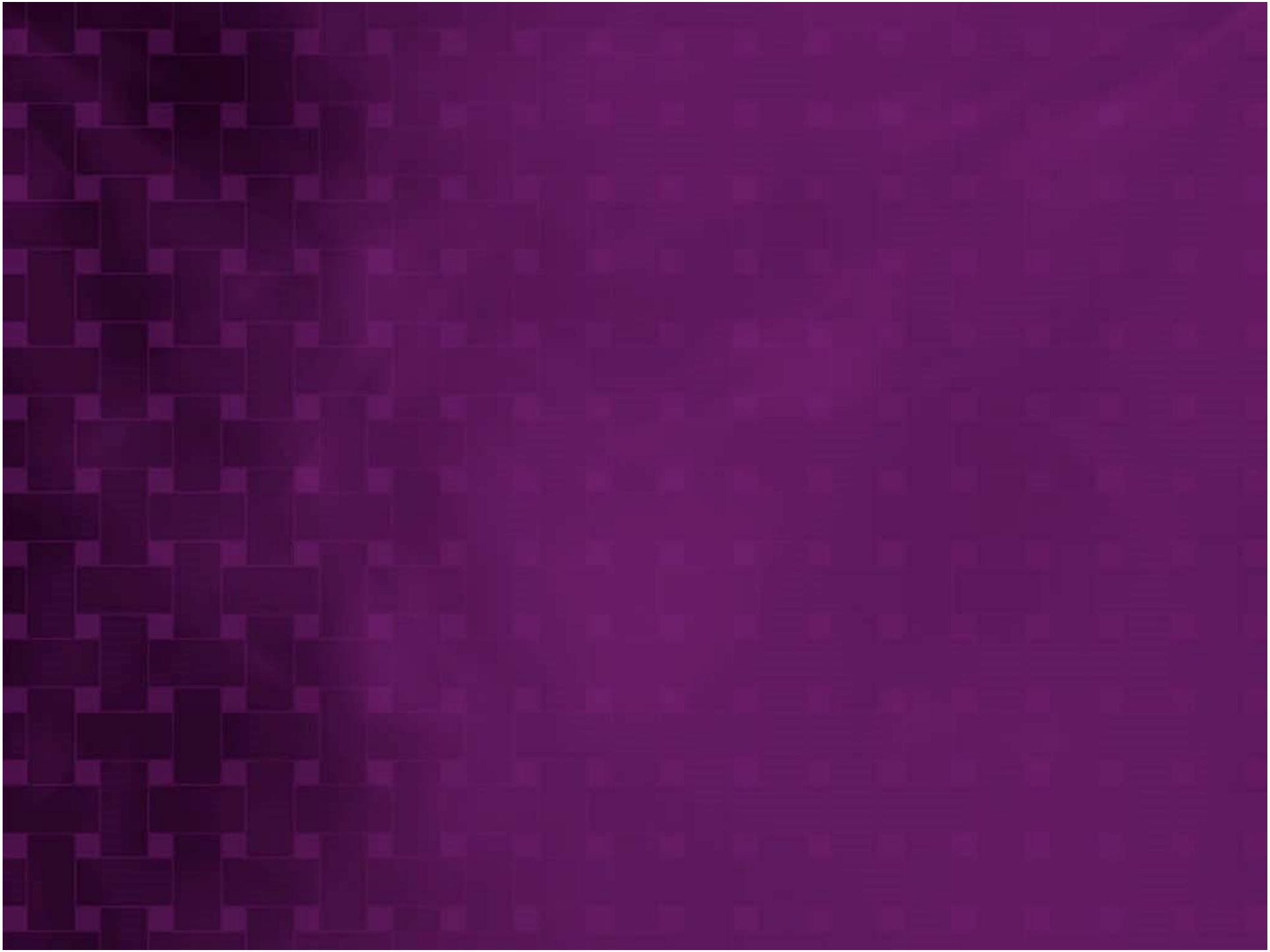
```
model_3 = model “borisejb.xml”
sim_repeat = simulate steadystate
task_1 = run model_3 with sim_repeat
repeatedtask_1 = repeat task_1 for J1_KK2
                 in [1, 5, 10, 50, 60, 70, 80, 90, 100]
repeatedtask_2 = repeat repeatedtask_1 for J4_KK5
                 in uniform(1, 40, 100)
plot repeatedtask_2.J4_KK1 vs repeatedtask_2.MKK_1,
                              repeatedtask_2.MKK_P_1
```

Integration with Python/Tellurium

- API:
  - convertFile(“filename”)
  - convertString(“model”)
  - getLastError()
  - setWorkingDirectory(“path/to/files”)

Integration with Python/Tellurium

```
import phrasedml as pml
phrasedmlstr = """
  model_1 = model "00001-sbml-l3v1.xml"
  simulation_1 = simulate uniform(0, 5, 51)
  task_1 = run simulation_1 on model_1
  plot time vs S1 / compartment, S2 / compartment
  report time, S1 / compartment, S2 / compartment
"""
sedml = pml.convertString(phrasedmlstr)
```



Integration with Python/Tellurium

```
import tellurium as te
antimonystr = """
model ex1
  S1 -> S2; k1*S1
  S1=10; S2=0; k1=0.1
end
"""
phrasedmlstr = """
  model_1 = model "ex1"
  simulation_1 = simulate uniform(0, 5, 50)
  task_1 = run simulation_1 on model_1
  plot time vs S1, S2
"""
exp1 = te.experiment(antimonystr, phrasedmlstr)
exp1.execute()
```

Repeated Tasks:

[variable] = repeat [task, task, …] for [loop]

task1 = run sim1 on model1

task2 = run sim1 on model2

task3 = repeat [task1, task2]
for model1.S1 in [1,3,6],
model2.S1 in [5,5,8]

Repeated Tasks:

[variable] = repeat [task, task, …] for [loop]

task1 = run sim1 on model1

task2 = run sim2 on **model1**

task3 = repeat [task1, task2]
for **task1**.S1 in [1,3,6],
task2.S1 in [5,5,8]

Technically illegal in SED-ML L1v2

phraSED ML

SIDEBAR: referencing nested model variables

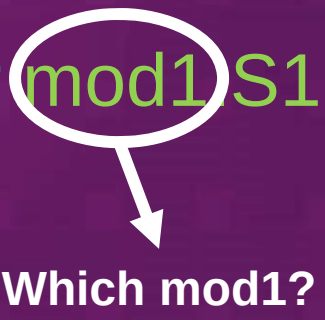
[variable] = repeat [task, task, …] for [loop]

t1 = run sim1 on mod1

t2 = run sim1 on mod2

t3 = run sim2 on mod1

t4 = repeat [t1, t2, t3] for mod1.S1 in [1,3,6]



phraSED ML

SIDEBAR: referencing nested model variables

[variable] = repeat [task, task, …] for [loop]

t1 = run sim1 on mod1

t2 = run sim1 on mod2

t3 = run sim2 on mod1

t4 = repeat [t1, t2, t3] for t1.mod1.S1 in [1,3,6]

Illegal in L1v2, but possible

phraSED ML

SIDEBAR: referencing nested model variables

[variable] = repeat [task, task, …] for [loop]

t1 = run sim1 on mod1

t2 = run sim1 on mod2

t3 = run sim2 on mod1

t4 = repeat [t1, t2, t3] for t1.mod1.S1 in [1,3,6]

t5 = repeat [t1, t2, t3] for t3.mod1.S1 in [3,6,10]

t6 = repeat [t4, t5] for t4.t2.mod2.S1 in [0,1,10]

Impossible in L1v2 (and not implemented in phraSED-ML)

Output:

plot [variable] vs [variable]

plot S1 vs S2

plot task1.S1 vs task1.S2

Output:

plot [variable] vs [variable]

plot S1 vs S2

plot task1.S1 vs task1.S2

plot task1.mod1.S1 vs task1.mod2.S2

Output:

plot [variable] vs [variable]

plot S1 vs S2
plot task1.S1 vs task1.S2
plot task1.mod1.S1 vs task1.mod2.S2
plot task4.task1.mod2.S1 vs task4.task2.mod1.S2

Output:

plot [variable] vs [variable]

plot S1 vs S2

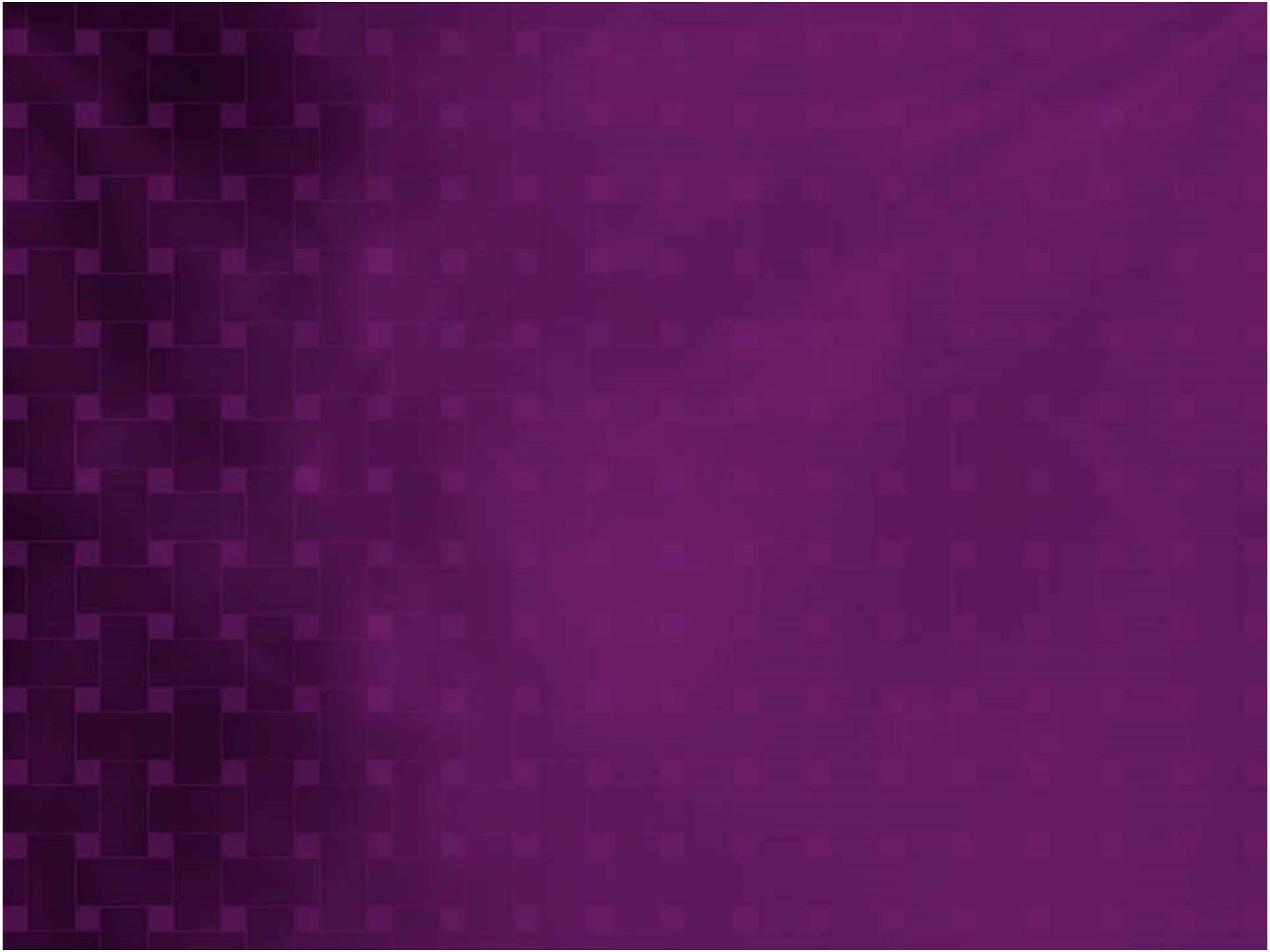
plot task1.S1 vs task1.S2

plot task1.mod1.S1 vs task1.mod2.S2

~~plot task4.task1.mod2.S1 vs task4.task2.mod1.S2~~

Issues

- ‘FunctionalRange’ seems to be useless
  - Repeats ‘computeChange’ functionality without adding anything
  - Does not actually allow dynamically-sized ranges (?)
- Impossible to reference deeply-nested models in repeatedTasks.



Examples: comparisons

```
model1 = model "BIOMD0000000021.xml"
model2 = model model1 with V_mT = 0.28, V_dT = 4.8
simulation1 = simulate uniform(0, 380, 1000)
task1 = run simulation1 on model1
task2 = run simulation1 on model2
plot task1.time vs task1.Mt
plot task2.time vs task2.Mt
plot task1.Cn vs task1.Mt
plot task2.Cn vs task2.Mt
```

Examples: repeated stochastic

```
model1 = model "C:\borisejb.xml"
timecourse1 = simulate uniform_stochastic(0, 4000, 1000)
task0 = run model1 with timecourse1
task1 = repeat task0 for uniform(0, 10, 10) with reset
plot task1.time vs task1.MAPK, task1.MAPK_P,
                   task1.MAPK_PP, task1.MKK,
                   task1.MKK_P, task1.MKKK, task1.MKKK_P
```